Semantically Secure Anonymity: Foundations of Re-Encryption

Adam L. Young[1] and Moti Yung[2]

[1] Cryptovirology Labs
[2] Snap Inc. and Dept. of Computer Science, Columbia University

Abstract. The notion of universal re-encryption is an established primitive used in the design of many anonymity protocols. It allows anyone to randomize a ciphertext without changing its size, without decrypting it, and without knowing the receiver's public key. By design it prevents the randomized ciphertext from being correlated with the original ciphertext. We revisit and analyze the security foundation of universal re-encryption and show that to date it has not had a satisfactory definition of security, in spite of its numerous uses. In modern times, a property of a cryptosystem cannot be just a side effect of the specific tools used in its first implementation; rather, a proper definition of correctness and security are needed when a new cryptographic primitive is defined. We demonstrate this by constructing a cryptosystem that satisfies the established definition of a universal cryptosystem but that has an encryption function that does not achieve anonymity; thereby forming a gap in the definition of security of universal re-encryption. We then use this gap to cryptanalyze and break the security definition of the mix application of Golle et al. To correct this, we introduce a new definition that includes the properties that are needed for a re-encryption cryptosystem to achieve key anonymity for *both* the encryption function and the re-encryption function building on "semantic security" and the original notion of "key anonymity." Our examples show that omitting any of our requirements introduces a security weakness. We then introduce a new (more flexible) generalization of the well-known Decision Diffie-Hellman (DDH) random self-reduction and use it, in turn, to prove that the original ElGamal-based universal cryptosystem of Golle et al is secure under our revised security definition. As a new application, we present a novel secure *Forward-Anonymous Batch Mix*.

1 Introduction

Nowadays, perhaps more then ever, anonymity tools are crucial for maintaining basic civil liberties. For example, as a result of the whistle-blowing by Edward Snowden, Americans and others have a better understanding of surveillance states and the privacy risks they pose. This reinforces the need for anonymity of communication, which, in fact, has been an active area of cryptographic research since the 1980s with numerous propositions and tools, suitable for various scenarios. Having a sound theoretical foundation for anonymity systems is a critical component in achieving privacy of users in the same way that message security is achieved by having a sound theoretical foundation for encryption. Camenisch and Lysyanskaya, for example, presented a formal treatment of onion routing [4] where prior work was comparatively informal with ad-hoc security justifications. Onion routing falls into a class of anonymity systems known as "decryption mixes", since layers of ciphertext are shed as the onion makes its way to the receiver.

In this work we present a formal treatment of a different fundamental class of anonymous communication protocols, namely, those based on universal re-encryption. This concept forms the basis of what has been called "re-encryption mixes". Golle, Jakobsson, Juels, and Syverson introduced the notion of universal re-encryption and presented a cryptosystem that implements it called `UCS` [10]. A ciphertext in a re-encryption mix has the property that it can be efficiently re-encrypted by anyone without knowledge of the receiver's public key. This is accomplished without ever exposing the underlying plaintext and without changing the size of the ciphertext. Using re-encryption randomness, the mapping is "lost" between the ciphertext that is supplied to the re-encryption operation and the resulting output ciphertext. Therefore, the notion of universal re-encryption propelled anonymity into the important area of "end-to-end encryption" systems that do not rely on servers for maintaining secrecy and that have the forward-secrecy property. Forward-secrecy and end-to-end encryption are becoming increasingly important in industrial systems in the post-Snowden era.

A number of anonymity protocols have been constructed that utilize universal re-encryption. Here, we show by way of analyzing the associated security definitions (i.e., a cryptanalysis methodology applied to provably secure systems) that the required level of security of this cryptographic primitive has not been properly defined. Specifically, we first show by way of concrete examples that the previous security definition of a universal cryptosystem does not sufficiently capture the key anonymity property. The definition does capture semantic security of messages. However, in this key anonymity setting where there is a complex goal that on top of semantic security requires additional properties, these needed properties and formal notions of security were neglected and were, in fact, never discussed. If we regard receiver anonymity, say, as having the same level of seriousness as message content security then they must be discussed!

As a result of this missing foundational step, previous claims involving re-encryption in anonymity systems have been based on heuristics. Indeed, there are no properties required, nor proofs that they are achieved, in the papers exploiting re-encryption for anonymity.

In contrast, what is needed is a formal foundation of the field (as was done in other areas such as message encryption). To this end, we put forth a model of what is required for re-encryption in the context systems that require key anonymity. In particular we show that the definition of security of a re-encryption mix from [10] does not require that the initial encryption function achieve key anonymity, thereby forming a gap in the definition. We exploit this gap to use an encryption system with basic security as defined in [10] and yet break the security definition of the mix application in [10], a break that results in the compromise of receiver anonymity. We re-emphasize that our investigation follows the traditional methodology of modern cryptography where similar corrections of definitions and proofs, and reductions have been a central theme aimed at establishing proper trust in security and privacy applications and protocols.

Maintaining message semantic security under the publicly performed re-encryption operation is relatively straightforward. But, this is misleading, since this characteristic does not hold for the entire set of properties needed for applications that require user anonymity. In particular, key anonymity of *both* the probabilistic encryption algorithm and the re-encryption algorithm were not properly defined or required in *any* prior work. As we know from other areas of cryptography, without foundations following a model, definitions, constructions and careful proofs, issues will surely arise (if not in the original implementation, perhaps in future ones).

We mention that aside from our analysis of the mix application in [10] and our batch mix application, this paper does not address re-encryption applications. Our primary focus is on the security foundation of universal re-encryption as a cryptographic *primitive*. We address the key anonymity of the encryption function and the key anonymity of the re-encryption function in a universal cryptosystem. Put another way, anonymous messaging, DC-nets, onion-routing, and e-voting applications are outside the scope of this paper. We therefore make no effort to frame our work on this cryptographic *primitive* relative to e-voting or re-encryption networks. We summarize our contributions as follows:

1. We show a gap in the definition of a universal re-encryption cryptosystem, namely, the missing requirement that the output of the encryption function achieve key anonymity.[3] Our cryptanalysis leverages this gap, *breaking* the security definition of the *mix application* in [10], thereby causing receiver anonymity to be compromised.
2. We present what we call *semantically secure anonymity* that defines the complete set of security properties that assure key anonymity. Existing protocols have taken the key anonymity of the encryption function for granted.
3. Construction: We generalize the well-known DDH random self-reduction and then use this generalization to prove that the original ElGamal-based universal cryptosystem of [10] is secure under DDH under our new model.[4] This is a new reduction technique that may have independent applications.
4. Example applications: We re-examine the original universal re-encryption protocol, present a new forward-anonymous batch mix, and show them to be secure (as modeled here) under DDH.

[3] Note we are talking about the "initial" encryption function, not the re-encryption function.

[4] i.e., that key-privacy holds for the encryption and re-encryption functions and that message indistinguishability holds for the encryption and re-encryption functions.

Due to its flexibility, we anticipate that our new reduction technique will aid in future concrete and workable designs that use number theoretic and elliptic curve groups where DDH holds, since anonymity of channels is a central issue in cryptography and privacy applications and since sound foundations and correct proofs are needed. In fact, our new application of a forward-anonymous batch mix is an example of such an application, giving an end-to-end secure anonymous communication system. One may argue that DDH based systems are an issue of the 1990s and are of no current interest. However, we claim that this work is evidence to the contrary: in cryptography subtleties do not disappear by themselves over time!

Organization: In Section 2 we present related work. Notation and definitions are covered in Section 3. In Section 4 we review the security definition of universal re-encryption due to [10] and show a gap in it's definition. We define semantically secure anonymity in Section 5. We present the universal re-encryption cryptosystem UCS of [10] in Section 6 with adjusted input/output specifications to accommodate our proofs of security. The new DDH reduction technique is covered in Section 7 and we use it to prove the security of UCS in Section 8 and Appendix A. The forward-secure batch mix is summarized in Section 9 and proven secure in Appendix B. We conclude in section 10.

2 Related Work

We first review the literature that leverages universal re-encryption as a primitive. Jakobsson et al presented a universal re-encryption cryptosystem that they referred to as UCS [10]. UCS is a 4-tuple of algorithms: a key generator, an encryption algorithm, a re-encryption algorithm, and a decryption algorithm. It is an extension of the ElGamal public key cryptosystem [6]. A ciphertext produced using this cryptosystem can be re-encrypted by anyone without first decrypting it. They present two applications that leverage a universal cryptosystem. In the first application an RFID tag is set to be a universal ciphertext that contains an underlying ID as the plaintext. The ciphertext is re-randomized periodically to prevent the tag from being tracked over time, e.g., as the object that contains the tag moves from place to place. With the private decryption key the ID can be obtained. Without the private key the ID in the ever changing RFID ciphertext cannot be obtained, making it difficult to track the object. They also apply universal re-encryption to construct a hybrid universal mix that leverages a public bulletin board. The mix is based on uploading and downloading ciphertexts to/from a bulletin board as opposed to leveraging a cascade of mix servers.

Fairbrother sought a more efficient hybrid universal cryptosystem based on `UCS` [7]. Universal re-encryption was used in a protocol to control anonymous information flow, e.g., to prevent spam from being injected into the anonymization network [14]. Onion-based routing and universal re-encryption were leveraged to form hybrid anonymous communication protocols [11, 15]. A circuit-based anonymity protocol was presented based on universal re-encryption [16]: in the first stage a channel is established through the network between Alice and Bob along with the keys needed for re-encryption and in the second stage Alice and Bob communicate with one another. Weakness in [14, 15, 11, 16] were presented in [5]. Golle presented a *reputable mix network* construction based on universal re-encryption [9]. A reputable mix has the property that the mix operator can prove that he or she did not author the content output by the mix.

Groth presented a re-randomizable and replayable cryptosystem based on DDH achieving adaptive chosen ciphertext security [12]. The construction and security arguments do not address key anonymity.

Prabhakaran and Rosulek presented a construction for a rerandomizable encryption scheme [19] that aims to be CCA-secure under DDH. It extends the Cramer-Shoup public key cryptosystem. They define RCCA receiver-anonymity in detail but state that their scheme does not achieve it and that it is an open problem. The approach was later extended to combine computability features with non-malleability of ciphertexts. The construction enables anyone to change an encryption of an unknown message m into an encryption of $T(m)$ (a feature), for a set of specific allowed functions T, but is non-malleable with respect to all other operations [20]. They indicate that their construction does not achieve HCCA-anonymity and leave the anonymity problem as open.

There has been recent work on proxy encryption [13]. In proxy encryption a ciphertext of a message m encrypted under Alice's public key is transformed (re-encrypted) into a ciphertext of m under Bob's public key. Note that our setting is different since the receiver's public key does not change in our re-encryption operation.

Re-encryption mix networks are utilized in electronic voting systems such as Helios [1]. They are also used in GR.NET's Zeus system.[5]

Having surveyed the literature it became apparent to us that numerous works have utilized universal re-encryption as a basic building block. This forms the motivation for a clean and correct foundation for this area. While we fully appreciate the pioneering work on this concept (a trailblazing step which is necessary), we believe that the time has come to treat anonymity with the same formal care and level of provability as, say, message security in public key cryptosystems. We believe that our work shows that identifying subtleties and producing necessary revisions is relevant, even for works that are older than 10 years, especially in areas that are becoming increasingly important to real-world applications.

The important notion of key privacy (also called key anonymity) was introduced by Boldyreva et al [2]. They formally defined public key cryptosystems that produce ciphertexts that do not reveal the receiver and showed that ElGamal and Cramer-Shoup achieve key privacy.

3 Notation and Definitions

If T is a finite set then $x \in_U T$ denotes sampling x uniformly at random from T. Define $\mathbb{Z}_p$ to be $\{0, 1, 2, ..., p-1\}$. Let $\mathbb{Z}_n^*$ be the set of integers from $\mathbb{Z}_n$ that

[5] github.com/grnet/zeus

are relatively prime to n. $[1,t]$ denotes the set of integers $\{1,2,...,t\}$. $|\mathbb{G}|$ denotes the size of the group $\mathbb{G}$, i.e., number of elements in $\mathbb{G}$. We may omit writing "mod p" when reduction modulo p is clear from the context. $\Pr[A]$ denotes the probability that A is true. Let $a \leftarrow b$ denote the assignment of b to a. For example, $a \leftarrow M(x)$ denotes the execution of Turing machine M on input x resulting in output a.

A function negl is *negligible* if for all polynomials $p(\cdot)$ there exists an α such that for all integers $n > \alpha$ it is the case that $\mathrm{negl}(n) < \frac{1}{p(n)}$. We use negl to denote a negligible function.

The following definition of DDH is directly from [3]. A group family $\mathbb{G}$ is a set of finite cyclic groups $\mathbb{G} = \{G_{\mathfrak{p}}\}$ where $\mathfrak{p}$ ranges over an infinite index set. We denote by $|\mathfrak{p}|$ the size of the binary representation of $\mathfrak{p}$. We assume that there is a polynomial time (in $|\mathfrak{p}|$) algorithm that given $\mathfrak{p}$ and two elements in $G_{\mathfrak{p}}$ outputs their sum. An instance generator, $\mathcal{IG}$, for $\mathbb{G}$ is a randomized algorithm that given an integer n (in unary), runs in time polynomial in n and outputs some random index $\mathfrak{p}$ and a generator g of $\mathbb{G}_{\mathfrak{p}}$. In particular, $(\mathfrak{p}, g) \leftarrow \mathcal{IG}(n)$. Note that for each n, the instance generator induces a distribution on the set of indices $\mathfrak{p}$. The index $\mathfrak{p}$ encodes the group parameters.

A DDH algorithm $\mathcal{A}$ for $\mathbb{G}$ is a probabilistic polynomial time Turing machine satisfying, for some fixed $\alpha > 0$ and sufficiently large n:

$$|\Pr[\mathcal{A}(\mathfrak{p}, g, g^a, g^b, g^{ab}) = \text{“true”}] - \Pr[\mathcal{A}(\mathfrak{p}, g, g^a, g^b, g^c) = \text{“true”}]| > \frac{1}{n^\alpha}$$

where g is a generator of $G_{\mathfrak{p}}$. The probability is over the random choice of $\langle \mathfrak{p}, g \rangle$ according to the distribution induced by $\mathcal{IG}(n)$, the random choice of a, b, and c in the range $[1, |G_{\mathfrak{p}}|]$ and the random bits used by $\mathcal{A}$. The group family $\mathbb{G}$ satisfies the DDH assumption if there is no DDH algorithm for $\mathbb{G}$.

We now review the well-known random-self reduction for DDH [3, 21, 18]. $\texttt{DDHRerand}((p, q), g, x, y, z)$ randomizes a DDH problem instance by choosing $u_1, u_2, v \in_U [1, q]$ and computing,

$$(x', y', z') \leftarrow (x^v g^{u_1}, y g^{u_2}, z^v y^{u_1} x^{v u_2} g^{u_1 u_2})$$

When (x, y, z) is a valid Diffie-Hellman 3-tuple then the output is a random Diffie-Hellman 3-tuple. When (x, y, z) is not a valid Diffie-Hellman 3-tuple then the output is a random 3-tuple.

4 Gap in Universal Re-encryption Definition

4.1 Review of the foundation of universal re-encryption

We review the definition of a universal cryptosystem exactly as defined in [10]. A universal cryptosystem (UCS) is a 4-tuple of algorithms defined by UCS = (UKG,UE,URe,UD), where UKG is the key generator, UE is the encryption algorithm, URe is the re-encryption algorithm, and UD is the decryption algorithm.

UKG outputs a public key PK (Golle et al do not have it return a key pair in the definition of their experiment). $\mathrm{UE}(m, r, PK)$ denotes the encryption of message m using public key PK and r is a re-encryption factor. It outputs a universal ciphertext C. $\mathrm{URe}(C, r)$ denotes the re-encryption of C using a re-encryption factor r. Golle et al assume an implicit parameterization of UCS under security parameter k. The decryption algorithm $\mathrm{UD}(SK, C)$ takes as input a private key SK and ciphertext C and returns the corresponding plaintext (or an indicator for failure).

Let $\mathbf{M}$ be a message space and let $\mathbf{R}$ be a set of encryption factors. Let $\mathcal{A}$ be a stateful adversarial algorithm. Below is the verbatim definition of *universal semantic security under re-encryption* (USS) from [10].

Experiment $\mathbf{Exp}_{\mathcal{A}}^{uss}(UCS, k)$
$PK_0 \leftarrow$ UKG;$PK_1 \leftarrow$ UKG;
$(m_0, m_1, r_0, r_1) \leftarrow \mathcal{A}(PK_0, PK_1,$"specify ciphertexts");
if $m_0, m_1 \notin \mathbf{M}$ or $r_0, r_1 \notin \mathbf{R}$ then output '0';
$C_0 \leftarrow \mathrm{UE}(m_0, r_0, PK_0)$;$C_1 \leftarrow \mathrm{UE}(m_1, r_1, PK_1)$;
$r'_0, r'_1 \in_U \mathbf{R}$;
$C'_0 \leftarrow \mathrm{URe}(C_0, r'_0)$;$C'_1 \leftarrow \mathrm{URe}(C_1, r'_1)$;
$b \in_U \{0, 1\}$;
$b' \leftarrow \mathcal{A}(C'_b, C'_{1-b},$"guess");
if $b = b'$ then output '1' else output '0';

An instantiation UCS is said to be *semantically secure under re-encryption* if for any adversary $\mathcal{A}$ with resources polynomial in K, the probability given by $\mathrm{pr}[\mathbf{Exp}_{\mathcal{A}}^{uss}(UCS, k) = \text{'1'}] - 1/2$ is negligible in k.

The UCS construction from [10] is as follows. let $\mathfrak{p} = (p, q)$ be a group family where p is prime and $p - 1$ is divisible by a large prime q. The group $G_{\mathfrak{p}}$ is the subgroup of $\mathbb{Z}_p^*$ having order q. The key generator outputs $(PK, SK) = (y, x)$ where y is the public key and $x \in_U \mathbb{Z}_q$

The encryption operation is denoted by $\mathrm{UE}(m, (k_0, k_1), y)$. It encrypts message $m \in G_{\mathfrak{p}}$ using y. $(k_0, k_1) \in_U \mathbb{Z}_q \times \mathbb{Z}_q$ are random encryption nonces. The encryption operation outputs the ciphertext $c \leftarrow ((\alpha_0, \beta_0), (\alpha_1, \beta_1)) \leftarrow ((my^{k_0} \bmod p, g^{k_0} \bmod p), (y^{k_1} \bmod p, g^{k_1} \bmod p))$.

The universal re-encryption algorithm $\mathrm{URe}(((\alpha_0, \beta_0), (\alpha_1, \beta_1)), (k'_0, k'_1))$ outputs a re-randomized ciphertext C'. $(k'_0, k'_1) \in_U \mathbb{Z}_q \times \mathbb{Z}_q$ is a random re-encryption factor. Generate $k'_0, k'_1 \in_U \mathbb{Z}_q$. The output C' is defined as,

$$C' = ((\alpha'_0, \beta'_0), (\alpha'_1, \beta'_1)) = ((\alpha_0 \alpha_1^{k'_0}, \beta_0 \beta_1^{k'_0}), (\alpha_1^{k'_1}, \beta_1^{k'_1}))$$

The decryption algorithm $\mathrm{UD}(x, ((\alpha_0, \beta_0), (\alpha_1, \beta_1)))$ takes as input the private key x followed by a universal ciphertext under public key y. First it verifies that all 4 values in the universal ciphertext are in $G_{\mathfrak{p}}$ and if not the special symbol $\perp$ is output. Compute $m_0 = \alpha_0/\beta_0^x$ and $m_1 = \alpha_1/\beta_1^x$. If $m_1 = 1$ then the output is $m = m_0$. Otherwise, output $\perp$ indicating decryption failure.

4.2 Missing key anonymity requirement for UE in USS definition

We now prove that USS as defined before (for which the security requirement is quoted above), accepts as valid cryptosystems that, in fact, contain encryption algorithms UE that do not produce key anonymous ciphertexts.

Cryptosystem A: Let UE be a universal encryption algorithm that outputs the ciphertext $((\alpha_0, \beta_0), (\alpha_1, \beta_1))$ computed according to UE in UCS except that k_0 is repeatedly generated until the least significant byte (lsb) of β_0 matches that of y. Suppose y_0 and y_1 have differing lsbs. When breaking the anonymity of UE, the adversary extracts the lsb of β_0 and correlates it with the public key with matching lsb.

This is "secure" under [10] (as per the definition copied above). USS, however, is devoid of a requirement that the output of UE be key anonymous. Put another way, it has a test of anonymity of URe *but there is no test of anonymity of UE*. Furthermore, this is the only definition of security spelled out for the universal re-encryption cryptosystem in [10]. The definition "accepts" as secure encryption algorithms UE that fail to achieve key anonymity as proven by the above example cryptosystem. There may exist other constructions in which the failure of UE to achieve key anonymity is subtle, yet like the above satisfy USS.

Practically, this means that cryptographers can construct universal cryptosystems that satisfy the USS definition but that have encryption algorithms that compromise the identity of the receiver without violating USS. This could potentially place the users of a universal cryptosystem in harms way.

Breaking the definition of security of the mix application: Let $\mathcal{U}$ be a universal cryptosystem that has an encryption algorithm UE that outputs ciphertexts that are not key anonymous. The universal mix network construction in Section 4 of [10] has, in the first step called "submission of inputs", users post ciphertexts produced by UE to a public bulletin board. When $\mathcal{U}$ is used in this universal mix network construction, the anonymity of receivers is compromised. We have therefore broken the security definition of the Golle et al mix application.

Consequently, defining security for universal re-encryption in a way that achieves key anonymity for ciphertexts output by UE has been left open. In addition, the properties of message indistinguishability for encryption and re-encryption were claimed to hold under DDH but no proof for this was given. In hindsight, we believe that the work in [10] was an *extremely* insightful step in the right direction of laying the foundation for universal re-encryption. In particular, we commend their approach of having the adversary fully specify the ciphertexts (messages and nonces) that are used in forming the re-encryption challenge ciphertexts. However, their definition is certainly insufficient as shown above.

5 Semantically Secure Anonymity

We now present the first definition of security for a universal cryptosystem that requires that the encryption algorithm provide key anonymity. We made slight adjustments to the input/output specifications of the algorithms in the universal re-encryption cryptosystem presented in [10]. For example, their key generator did not take a security parameter as input, ours does. We define the algorithms in the cryptosystem to take auxiliary information such as group parameters as input. We remark that the adjustments to the input/output specifications of the algorithms are superficial. We made them to support the full proofs of security that we provide.

Definition 1. *A universal cryptosystem Π is a 4-tuple of probabilistic polynomial time algorithms (UKG, UE, URe, UD) together with auxiliary information λ (e.g., group parameters) such that:*

1. *The key generation algorithm $UKG(n, \lambda)$ takes as input a security parameter n (in unary) and λ and outputs (pk, sk) where pk is a public key and sk is the corresponding private key.*
2. *The encryption algorithm $UE_{pk}(m, k, \lambda)$ is deterministic and it takes as input a public key pk, a message m from the underlying plaintext space, an encryption nonce k, and λ. It outputs a ciphertext c. The operation is expressed as $c \leftarrow UE_{pk}(m, k, \lambda)$.*
3. *The re-encryption algorithm $URe(c, k, \lambda)$ is deterministic and it takes as input a ciphertext c, a re-encryption nonce k, and λ. It outputs a ciphertext c'. The operation is expressed as $c' \leftarrow URe(c, k, \lambda)$.*
4. *The decryption algorithm $UD_{sk}(c, \lambda)$ takes as input a private key sk, a ciphertext c, and λ. It outputs a message m and a Boolean s. s is true if and only if decryption succeeds. The operation is expressed as $(m, s) \leftarrow UD_{sk}(c)$.*

It is required that, for all m, the ordered execution of $c_0 \leftarrow UE_{pk}(m, k_0, \lambda)$, $c_{i+1} \leftarrow URe(c_i, k_i, \lambda)$ for $i = 0, 1, 2, ..., t-1$, $(m', s) \leftarrow UD_{sk}(c_t, \lambda)$ with $(m', s) = (m, true)$ except with possibly negligible probability over (pk, sk) that is output by $UKG(n, \lambda)$ and the randomness used by the nonces for UE and URe. Here t is bounded from above by u^α for some fixed $\alpha > 0$ and sufficiently large u.

Definition 2 for message indistinguishability has been adapted from [8, 17].

Definition 2. *The experiment for eavesdropping indistinguishability for the encryption operation is $PubKEnc^{eav}_{\mathcal{A},\Pi}(n, \lambda)$:*

1. *$UKG(n, \lambda)$ is executed to get (pk, sk).*
2. *Adversary $\mathcal{A}(n, \lambda, pk)$ outputs a pair of messages (m_0, m_1) where m_0 and m_1 have the same length. These messages must be in the plaintext space associated with pk.*
3. *A random bit $b \in_U \{0, 1\}$ and random nonce k are chosen. Then ciphertext $c \leftarrow UE_{pk}(m_b, k, \lambda)$ is computed and provided to $\mathcal{A}$. This is the challenge ciphertext.*

4. $\mathcal{A}(c)$ outputs a bit b'.
5. The output of the experiment is defined to be 1 if $b' = b$ and 0 otherwise.

Definition 3. *The experiment for eavesdropping indistinguishability for the re-encryption operation is* $PubKReEnc^{eav}_{\mathcal{A},\Pi}(n, \lambda)$*:*

1. $UKG(n, \lambda)$ is executed to get (pk, sk).
2. Adversary $\mathcal{A}(n, \lambda, pk)$ outputs $((m_0, k_0), (m_1, k_1))$ where (m_i, k_i) is a message/nonce pair for $i = 0, 1$. The messages must be of the same length. These messages must be in the plaintext space associated with pk.
3. A random bit $b \in_U \{0, 1\}$ and random nonce k are chosen. Then ciphertext $c \leftarrow UE_{pk}(m_b, k_b, \lambda)$ is computed. Then $c' \leftarrow URe(c, k, \lambda)$ is computed and provided to $\mathcal{A}$. This is the challenge ciphertext.
4. $\mathcal{A}(c')$ outputs a bit b'.
5. The output of the experiment is defined to be 1 if $b' = b$ and 0 otherwise.

Definition 4 is key anonymity [2]. Definition 5 is key anonymity adapted for re-encryption.

Definition 4. *The experiment for key anonymity of the encryption operation is denoted by* $AnonEnc^{eav}_{\mathcal{A},\Pi}(n, \lambda)$ *and is as follows:*

1. $UKG(n, \lambda)$ is executed twice to get (pk_0, sk_0) and (pk_1, sk_1).
2. Adversary $\mathcal{A}(n, \lambda, pk_0, pk_1)$ outputs a message m. This message must be in the plaintext space associated with pk_0 and pk_1.
3. A random bit $b \in_U \{0, 1\}$ and random nonce k are chosen. Then ciphertext $c \leftarrow UE_{pk_b}(m, k, \lambda)$ is computed and provided to $\mathcal{A}$. This is the challenge ciphertext.
4. $\mathcal{A}(c)$ outputs a bit b'.
5. The output of the experiment is defined to be 1 if $b' = b$ and 0 otherwise.

Definition 5. *The experiment for key anonymity of the re-encryption operation is denoted by* $AnonReEnc^{eav}_{\mathcal{A},\Pi}(n, \lambda)$ *and is as follows:*

1. $UKG(n, \lambda)$ is executed twice to get (pk_0, sk_0) and (pk_1, sk_1).
2. Adversary $\mathcal{A}(n, \lambda, pk_0, pk_1)$ outputs (m, k) where m is a message and k is an encryption nonce k. The message m must be in the plaintext space associated with pk_0 and pk_1.
3. A random bit $b \in_U \{0, 1\}$ and random nonce k' are chosen. Then $c \leftarrow UE_{pk_b}(m, k, \lambda)$ is computed. Then $c' \leftarrow URe(c, k', \lambda)$ is computed and provided to $\mathcal{A}$. This is the challenge ciphertext.
4. $\mathcal{A}(c')$ outputs a bit b'.
5. The output of the experiment is defined to be 1 if $b' = b$ and 0 otherwise.

Definition 6. *A universal cryptosystem Π is secure in the sense of* ***semantically secure anonymity*** *for security parameter n (in unary) and auxiliary information λ if it satisfies the following:*

1. $\Pr[PubKEnc^{eav}_{\mathcal{A},\Pi}(n,\lambda)=1] \leq \frac{1}{2} + \text{negl}(n)$
2. $\Pr[PubKReEnc^{eav}_{\mathcal{A},\Pi}(n,\lambda)=1] \leq \frac{1}{2} + \text{negl}(n)$
3. $\Pr[AnonEnc^{eav}_{\mathcal{A},\Pi}(n,\lambda)=1] \leq \frac{1}{2} + \text{negl}(n)$
4. $\Pr[AnonReEnc^{eav}_{\mathcal{A},\Pi}(n,\lambda)=1] \leq \frac{1}{2} + \text{negl}(n)$

We recap and say that correctness of decryption is obviously a must and we have demonstrated that message security must be required for the encryption and the re-encryption operations in order to maintain the security of the message throughout the system. Further, as the examples above demonstrated, key anonymity is required for these two operations as well. Intuitively, any violation of message security will render the encryption useless. Also, any tracing via the re-encryption operation due to message or key linkability will violate strict anonymity. Similarly, any tracing via the encryption operation due to message or key linkability will violate strict anonymity.

6 Universal Re-Encryption Cryptosystem

Given that we adjusted the input/output specifications of the algorithms in the universal re-encryption cryptosystem presented in [10] (see Section 5), we now present the cryptosystem in full. We stress that we are using the universal re-encryption cryptosystem in [10].

Let n be a security parameter (in unary) and let $\mathfrak{p}=(p,q)$ be a group family where p is prime and $p-1$ is divisible by a large prime q. The group $G_{\mathfrak{p}}$ is the subgroup of $\mathbb{Z}_p^*$ having order q. For key anonymity, the single group $((p,q),g)$ is generated once using $\mathcal{IG}(n)$ and is then used by all users. The auxiliary information λ is defined to be $((p,q),g)$. We define the following to be universal cryptosystem Ψ.

Key Generation: Key generation is denoted by $(y,x) \leftarrow \texttt{UKG}(n,\lambda)$. Here $y \leftarrow g^x \bmod p$ where $x \in_U [1,q]$. The public key is $pk=y$ and the private key is $sk=x$.

Encryption: Encryption is denoted by $\texttt{UE}_{pk}(m,(k_0,k_1),\lambda)$. It encrypts message $m \in G_{\mathfrak{p}}$ using y. $(k_0,k_1) \in_U [1,q]\times[1,q]$ is a random encryption nonce. The operation outputs the ciphertext $c \leftarrow ((\alpha_0,\beta_0),(\alpha_1,\beta_1)) \leftarrow ((my^{k_0} \bmod p),(g^{k_0} \bmod p),((y^{k_1} \bmod p),(g^{k_1} \bmod p))$.

Decryption: The following decryption operation is denoted by $\texttt{UD}_{sk}(c,\lambda)$. Here c is the ciphertext $((\alpha_0,\beta_0),(\alpha_1,\beta_1))$. Compute $m_1 \leftarrow \alpha_1/\beta_1^x \bmod p$. If $m_1=1$ then set $s=$ true else set $s=$ false. If $s=$ true set $m_0=\alpha_0/\beta_0^x \bmod p$ else set m_0 to be the empty string. $s=$ true indicates successful decryption. Return (m_0,s).

Universal Re-encryption: The universal re-encryption operation is denoted by $\texttt{URe}(((\alpha_0,\beta_0),(\alpha_1,\beta_1)),(\ell_0,\ell_1),\lambda)$. The pair $c=((\alpha_0,\beta_0),(\alpha_1,\beta_1))$ is a universal ciphertext and $(k_0',k_1') \in_U [1,q]\times[1,q]$ is a re-encryption nonce. Compute $(\alpha_0',\beta_0') \leftarrow (\alpha_0\alpha_1^{k_0'} \bmod p, \beta_0\beta_1^{k_0'} \bmod p)$ and compute $(\alpha_1',\beta_1') \leftarrow (\alpha_1^{k_1'} \bmod p, \beta_1^{k_1'} \bmod p)$. Output the ciphertext $c' \leftarrow ((\alpha_0',\beta_0'),(\alpha_1',\beta_1'))$.

7 The New Construction: Expanded DDH Self-Reduction

We now generalize the DDH random self-reduction to output five values instead of three. This allows us to transform a DDH problem instance into either two DH 3-tuples with a common "public key" or a random 5-tuple, depending on the input problem instance. We utilize this property in our proofs of security in Section 8. We define algorithm `DDHRerand5` as follows. $\texttt{DDHRerand5}((p,q),g,x,y,z)$ randomizes a DDH problem instance by choosing the values $u_1, u_2, v, v', u_1' \in_U [1,q]$ and computing,

$$(x'', x', y', z', z'') \leftarrow (x^{v'} g^{u_1'}, x^v g^{u_1}, y g^{u_2}, z^v y^{u_1} x^{v u_2} g^{u_1 u_2}, z^{v'} y^{u_1'} x^{v' u_2} g^{u_1' u_2})$$

Case 1. Suppose (x,y,z) is a valid Diffie-Hellman (DH) 3-tuple. Then $x = g^a$, $y = g^b$, $z = g^{ab}$ for some a, b. It follows that (x', y', z') is also a valid DH 3-tuple. It is straightforward to show that (x'', y', z'') is a valid DH 3-tuple as well.

Case 2. Suppose (x,y,z) is not a valid DH 3-tuple. Then $x = g^a$, $y = g^b$, $z = g^{ab+c}$ for some $c \neq 0$. In this case, $x' = g^{a'}$, $y' = g^{b'}$, $z' = g^{a'b'} g^{cv}$. Since $c \neq 0$ it follows that g^c is a generator of $G_{\mathtt{p}}$. Also, $x'' = g^{a''}$, $y' = g^{b'}$, $z'' = g^{a''b'} g^{cv'}$.

So, when (x,y,z) is a valid DH 3-tuple then (x',y',z') and (x'',y',z'') are random DH 3-tuples with y' in common and when (x,y,z) is not a valid DH 3-tuple then the output is a random 5-tuple.

8 Security of Universal Cryptosystem Ψ

We now prove the security of our construction. These are the first proofs of security for universal re-encryption that constitute direct reductions with respect to DDH.

Theorem 1. *If DDH is hard then* $\Pr[\mathit{AnonEnc}^{eav}_{\mathcal{A},\Psi}(n,\lambda) = 1] \leq \frac{1}{2} + \mathrm{negl}(n)$.

Proof. Suppose there exists a probabilistic polynomial time adversary $\mathcal{A}$ for $\mathit{AnonEnc}^{eav}_{\mathcal{A},\Psi}$, an $\alpha > 0$, and a sufficiently large κ, such that $\mathcal{A}$ succeeds with probability greater than or equal to $\frac{1}{2} + \frac{1}{\kappa^\alpha}$. Consider algorithm `AlgR3` that takes as input a DDH problem instance $((p,q), g, a_0, b_0, c_0)$.

$\texttt{AlgR3}((p,q), g, a_0, b_0, c_0)$:
1. set $(\theta_j', \theta_j, y_j, \mu_j, \mu_j') \leftarrow \texttt{DDHRerand5}((p,q), g, a_0, b_0, c_0)$ for $j = 0, 1$
2. $m \leftarrow \mathcal{A}(n, \lambda, y_0, y_1)$
3. generate $u \in_U \{0,1\}$
4. set $c \leftarrow ((\alpha_0, \beta_0), (\alpha_1, \beta_1)) \leftarrow ((m\mu_u, \theta_u), (\mu_u', \theta_u'))$
5. $u' \leftarrow \mathcal{A}(c)$
6. if $u = u'$ then output "true" else output "false"

Consider the case that the input is a DH 3-tuple. It follows from the definition of `DDHRerand5` that c is an encryption of m in accordance with `UE` using y_u as the public key. Therefore, the input to $\mathcal{A}$ is drawn from the same set and probability distribution as the input to $\mathcal{A}$ in Definition 4. It follows that $u = u'$ with probability greater than or equal to $\frac{1}{2} + \frac{1}{\kappa^\alpha}$. So, for random exponents a and b in $[1, q]$, $\Pr[\mathtt{AlgR3}((p, q), g, g^a, g^b, g^{ab}) = \text{“true”}] \geq \frac{1}{2} + \frac{1}{\kappa^\alpha}$. Define $\psi = \Pr[\mathtt{AlgR3}((p, q), g, g^a, g^b, g^{ab}) = \text{“true”}]$.

Now consider the case that the input is not a DH 3-tuple. It follows from the definition of `DDHRerand5` that the 5-tuple $(\theta'_u, \theta_u, y_u, \mu_u, \mu'_u)$ is uniformly distributed in $G_{\mathtt{p}}^5$. Therefore, c is uniformly distributed in $G_{\mathtt{p}}^2 \times G_{\mathtt{p}}^2$. Let p_1 be the probability that $\mathcal{A}$ responds with $u' = 0$. Then the probability that $u = u'$ is $\frac{1}{2}p_1 + \frac{1}{2}(1 - p_1) = \frac{1}{2}$. So, for randomly chosen exponents a, b, and c in $[1, q]$, the probability $\Pr[\mathtt{AlgR3}((p, q), g, g^a, g^b, g^c) = \text{“true”}] = \frac{q^2}{q^3}\psi + (1 - \frac{q^2}{q^3})\frac{1}{2} = \frac{1}{2} + \frac{2\psi - 1}{2q}$ which is overwhelmingly close to $\frac{1}{2}$. □

Theorem 2. *If DDH is hard then* $\Pr[\mathit{AnonReEnc}^{eav}_{\mathcal{A},\Psi}(n, \lambda) = 1] \leq \frac{1}{2} + \mathrm{negl}(n)$.

Proof. Suppose there exists a probabilistic polynomial time adversary $\mathcal{A}$ for $\mathit{AnonReEnc}^{eav}_{\mathcal{A},\Psi}$, an $\alpha > 0$, and a sufficiently large κ such that $\mathcal{A}$ succeeds with probability greater than or equal to $\frac{1}{2} + \frac{1}{\kappa^\alpha}$. Consider algorithm `AlgR4` that takes as input a Decision Diffie-Hellman problem instance $((p, q), g, a_0, b_0, c_0)$.

$\mathtt{AlgR4}((p, q), g, a_0, b_0, c_0)$:
1. $(\theta'_j, \theta_j, y_j, \mu_j, \mu'_j) \leftarrow \mathtt{DDHRerand5}((p, q), g, a_0, b_0, c_0)$ for $j = 0, 1$
2. $(m, (k_0, k_1)) \leftarrow \mathcal{A}(n, \lambda, y_0, y_1)$
3. $u \in_U \{0, 1\}$
4. $((\alpha_0, \beta_0), (\alpha_1, \beta_1)) \leftarrow \mathtt{UE}_{y_u}(m, (k_0, k_1), \lambda)$
5. $c' \leftarrow ((\alpha'_0, \beta'_0), (\alpha'_1, \beta'_1)) \leftarrow ((\alpha_0\mu_u, \beta_0\theta_u), (\mu'_u, \theta'_u))$
6. $u' \leftarrow \mathcal{A}(c')$
7. if $u = u'$ then output “true” else output “false”

Consider the case that the input is a DH 3-tuple. Clearly $((\alpha_0, \beta_0), (\alpha_1, \beta_1))$ is the ciphertext under public key y_u as specified by $\mathcal{A}$. It follows from the definition of `DDHRerand5` that c' is a re-encryption of $((\alpha_0, \beta_0), (\alpha_1, \beta_1))$ in accordance with `URe`. Therefore, the input to $\mathcal{A}$ is drawn from the same set and probability distribution as the input to $\mathcal{A}$ in Definition 5. It follows that $u = u'$ with probability greater than or equal to $\frac{1}{2} + \frac{1}{\kappa^\alpha}$. So, for random exponents a and b in $[1, q]$, $\Pr[\mathtt{AlgR4}((p, q), g, g^a, g^b, g^{ab}) = \text{“true”}] \geq \frac{1}{2} + \frac{1}{\kappa^\alpha}$. Define the value ψ to be $\Pr[\mathtt{AlgR4}((p, q), g, g^a, g^b, g^{ab}) = \text{“true”}]$.

Now consider the case that the input is not a DH 3-tuple. It follows from definition of `DDHRerand5` that the 5-tuple $(\theta'_u, \theta_u, y_u, \mu_u, \mu'_u)$ is uniformly distributed in $G_{\mathtt{p}}^5$. Therefore, c' is uniformly distributed in $G_{\mathtt{p}}^2 \times G_{\mathtt{p}}^2$. Let p_1 be the probability that $\mathcal{A}$ responds with $u' = 0$. Then the probability that $u = u'$ is $\frac{1}{2}p_1 + \frac{1}{2}(1 - p_1) = \frac{1}{2}$. So, for randomly chosen exponents a, b, and c in $[1, q]$, the probability $\Pr[\mathtt{AlgR4}((p, q), g, g^a, g^b, g^c) = \text{“true”}] = \frac{1}{2} + \frac{2\psi - 1}{2q}$. □

The proofs of the below are in Appendix A.

Theorem 3. *If DDH is hard then* $\Pr[PubKEnc^{eav}_{\mathcal{A},\Psi}(n,\lambda)=1] \leq \frac{1}{2} + \mathrm{negl}(n)$.

Theorem 4. *If DDH is hard then* $\Pr[PubKReEnc^{eav}_{\mathcal{A},\Psi}(n,\lambda)=1] \leq \frac{1}{2} + \mathrm{negl}(n)$.

Theorems 1, and 2, 3, and 4 show the following.

Theorem 5. *If DDH is hard then Ψ is secure in the sense of semantically secure anonymity.*

9 Batch Mixing

A forward-anonymous mix protocol was presented in [10]. The definition of security for it is broken since it only requires that the universal cryptosystem satisfy USS, thereby allowing the use of a universal cryptosystem with a UE algorithm that does not produce key anonymous ciphertexts. When such an encryption algorithm is used, the anonymity of the receiver is compromised in step 1. We define a re-encryption batch mix protocol `FBMIX` in Appendix B together with a definition of security for it that assures anonymity of receivers. We prove that it is secure using direct reductions with respect to DDH. We claim the below. The proof is in Appendix B.

Theorem 6. *If DDH is hard then* `FBMIX` *is a forward-anonymous batch mix.*

10 Conclusion

We addressed basic definitions, constructions, and proofs related to anonymous end-to-end communication. We analyzed the security foundation of universal re-encryption and showed that the definition of security for it is missing the requirement that the encryption algorithm produce key anonymous ciphertexts, thereby forming a gap. We leveraged the gap to break the security definition of the mix application of Golle et al. We then presented a new definition of security for universal re-encryption that requires that message indistinguishability and key anonymity hold for both the encryption and re-encryption operations. We then proved that the original ElGamal-based universal cryptosystem of Golle et al is secure under our new definition of security. Finally, we presented a forward-anonymous batch mix that is secure under DDH.

References

1. B. Adida. Helios: Web-based Open-Audit Voting. In *Proceedings of the Seventeenth Usenix Security Symposium*, pages 335–348, 2008.
2. M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-Privacy in Public-Key Encryption. In *Asiacrypt '01*, pages 566–582, 2001.
3. D. Boneh. The Decision Diffie-Hellman Problem. In *Proceedings of the Third Algorithmic Number Theory Symposium*, pages 48–63, 1998.

4. J. Camenisch and A. Lysyanskaya. A Formal Treatment of Onion Routing. In *Advances in Cryptology—Crypto '05*, pages 169–187, 2005.
5. G. Danezis. Breaking Four Mix-related Schemes Based on Universal Re-encryption. *Int. J. Inf. Sec.*, 6(6):393–402, 2007.
6. T. ElGamal. A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. In *Advances in Cryptology—Crypto '84*, pages 10–18, 1985.
7. P. Fairbrother. An Improved Construction for Universal Re-encryption. In *Privacy Enhancing Technologies*, pages 79–87, 2004.
8. S. Goldwasser and S. Micali. Probabilistic Encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.
9. P. Golle. Reputable Mix Networks. In *Privacy Enhancing Technologies*, pages 51–62, 2004.
10. P. Golle, M. Jakobsson, A. Juels, and P. Syverson. Universal Re-encryption for Mixnets. In *CT-RSA 2004*, pages 163–178, 2004.
11. M. Gomułkiewicz, M. Klonowski, and M. Kutyłowski. Onions Based on Universal Re-encryption—Anonymous Communication Immune Against Repetitive Attack. In *WISA*, pages 400–410, 2004.
12. J. Groth. Rerandomizable and Replayable Adaptive Chosen Ciphertext Attack Secure Cryptosystems. In *Theory of Cryptography—TCC '04*, pages 152–170, 2004.
13. S. Hohenberger, G. N. Rothblum, A. Shelat, and V. Vaikuntanathan. Securely Obfuscating Re-encryption. *Journal of Cryptology*, 24(4):694–719, 2011.
14. M. Klonowski, M. Kutyłowski, A. Lauks, and F. Zagórski. Universal Re-encryption of Signatures and Controlling Anonymous Information Flow. In *WARTACRYPT*, pages 179–188, 2004.
15. M. Klonowski, M. Kutyłowski, and F. Zagórski. Anonymous Communication with On-line and Off-line Onion Encoding. In *SOFSEM*, pages 229–238, 2005.
16. T. Lu, B. Fang, Y. Sun, and L. Guo. Some Remarks on Universal Re-encryption and a Novel Practical Anonymous Tunnel. In *ICCNMC*, pages 853–862, 2005.
17. S. Micali, C. Rackoff, and B. Sloan. The Notion of Security for Probabilistic Cryptosystems. *SIAM Journal on Computing*, 17(2):412–426, 1988.
18. M. Naor and O. Reingold. Number-Theoretic Constructions of Efficient Pseudo-Random Functions. In *IEEE FOCS '97*, pages 458–467, 1997.
19. M. Prabhakaran and M. Rosulek. Rerandomizable RCCA Encryption. In *Advances in Cryptology—Crypto '07*, pages 517–534, 2007.
20. M. Prabhakaran and M. Rosulek. Homomorphic Encryption with CCA Security. In *Automata, Languages and Programming*, pages 667–678, 2008.
21. M. Stadler. Publicly Verifiable Secret Sharing. In *Advances in Cryptology—Eurocrypt '96*, pages 190–199, 1996.

A Proofs of message indistinguishability

Below is proof of Theorem 3.

Proof. Suppose there exists a probabilistic polynomial time adversary $\mathcal{A}$ for $PubKEnc^{eav}_{\mathcal{A},\Psi}$, an $\alpha > 0$ and a sufficiently large κ, such that $\mathcal{A}$ succeeds with probability greater than or equal to $\frac{1}{2} + \frac{1}{\kappa^{\alpha}}$. Consider algorithm `AlgR1` that takes as input a DDH problem instance $((p, q), g, a_0, b_0, c_0)$.

$\texttt{AlgR1}((p,q),g,a_0,b_0,c_0)$:
1. set $(\theta',\theta,y,\mu,\mu') \leftarrow \texttt{DDHRerand5}((p,q),g,a_0,b_0,c_0)$
2. $(m_0,m_1) \leftarrow \mathcal{A}(n,\lambda,y)$
3. $b \in_U \{0,1\}$
4. $c \leftarrow ((\alpha_0,\beta_0),(\alpha_1,\beta_1)) \leftarrow ((m_b\mu,\theta),(\mu',\theta'))$
5. $b' \leftarrow \mathcal{A}(c)$
6. if $b = b'$ then output "true" else output "false"

Consider the case that the input is a DH 3-tuple. It follows from the definition of DDHRerand5 that c is an encryption of m_b according to UE using y as the public key. Therefore, the input to $\mathcal{A}$ is drawn from the same set and probability distribution as the input to $\mathcal{A}$ in Definition 2. It follows that $b = b'$ with probability greater than or equal to $\frac{1}{2} + \frac{1}{\kappa^\alpha}$. So, for random exponents a and b in $[1,q]$, $\Pr[\texttt{AlgR1}((p,q),g,g^a,g^b,g^{ab}) = \text{"true"}] \geq \frac{1}{2} + \frac{1}{\kappa^\alpha}$. Define $\psi = \Pr[\texttt{AlgR1}((p,q),g,g^a,g^b,g^{ab}) = \text{"true"}]$.

Now consider the case that the input is not a DH 3-tuple. It follows from the definition of DDHRerand5 that $(\theta',\theta,y,\mu,\mu')$ is uniformly distributed in $G_{\mathtt{p}}^5$. Therefore, c is uniformly distributed in $G_{\mathtt{p}}^2 \times G_{\mathtt{p}}^2$. Let p_1 be the probability that $\mathcal{A}$ responds with $b' = 0$. Then the probability that $b = b'$ is $\frac{1}{2}p_1 + \frac{1}{2}(1 - p_1) = \frac{1}{2}$. So, for randomly chosen exponents a, b, and c in $[1,q]$, the probability $\Pr[\texttt{AlgR1}((p,q),g,g^a,g^b,g^c) = \text{"true"}] = \frac{1}{2} + \frac{2\psi-1}{2q}$. □

Below is the proof of Theorem 4.

Proof. Suppose there exists a probabilistic polynomial time adversary $\mathcal{A}$ for $PubKReEnc^{eav}_{\mathcal{A},\Psi}$, an $\alpha > 0$, and a sufficiently large κ, such that $\mathcal{A}$ succeeds with probability greater than or equal to $\frac{1}{2} + \frac{1}{\kappa^\alpha}$. Consider algorithm AlgR2 that takes as input a DDH problem instance $((p,q),g,a_0,b_0,c_0)$.

$\texttt{AlgR2}((p,q),g,a_0,b_0,c_0)$:
1. set $(\theta',\theta,y,\mu,\mu') \leftarrow \texttt{DDHRerand5}((p,q),g,a_0,b_0,c_0)$
2. $((m_0,r_0),(m_1,r_1)) \leftarrow \mathcal{A}(n,\lambda,y)$
3. $b \in_U \{0,1\}$
4. $((\alpha_0,\beta_0),(\alpha_1,\beta_1)) \leftarrow \texttt{UE}_y(m_b,r_b,\lambda)$
5. $c' \leftarrow ((\alpha_0\mu,\beta_0\theta),(\mu',\theta'))$
6. $b' \leftarrow \mathcal{A}(c')$
7. if $b = b'$ then output "true" else output "false"

Consider the case that the input is a DH 3-tuple. Clearly $((\alpha_0,\beta_0),(\alpha_1,\beta_1))$ is the ciphertext of m_b as specified by adversary $\mathcal{A}$. It follows from the definition of DDHRerand5 that c' is a re-encryption of $((\alpha_0,\beta_0),(\alpha_1,\beta_1))$ according to URe. Therefore, the input to $\mathcal{A}$ is drawn from the same set and probability distribution as the input to $\mathcal{A}$ in Definition 3. It follows that $b = b'$ with probability greater than or equal to $\frac{1}{2} + \frac{1}{\kappa^\alpha}$. So, for random exponents a and b in $[1,q]$, $\Pr[\texttt{AlgR2}((p,q),g,g^a,g^b,g^{ab}) = \text{"true"}] \geq \frac{1}{2} + \frac{1}{\kappa^\alpha}$. Define the value ψ to be $\Pr[\texttt{AlgR2}((p,q),g,g^a,g^b,g^{ab}) = \text{"true"}]$.

Now consider the case that the input is not a DH 3-tuple. It follows from the definition of DDHRerand5 that $(\theta', \theta, y, \mu, \mu')$ is uniformly distributed in the set $G_{\mathtt{p}}^5$. Therefore, c' is uniformly distributed in $G_{\mathtt{p}}^2 \times G_{\mathtt{p}}^2$. Let p_1 be the probability that $\mathcal{A}$ responds with $b' = 0$. Then the probability that $b = b'$ is $\frac{1}{2}p_1 + \frac{1}{2}(1 - p_1) = \frac{1}{2}$. So, for randomly chosen exponents a, b, and c in $[1, q]$, the probability $\Pr[\mathtt{AlgR2}((p, q), g, g^a, g^b, g^c) = \text{"true"}] = \frac{1}{2} + \frac{2\psi - 1}{2q}$. □

B Forward-Anonymous Batch Mix

Golle et al used the universal cryptosystem to construct a forward anonymous mix protocol centered around the use of a bulletin board. The number of ciphertexts on the board can vary over time. Servers download the ciphertexts from the board, re-randomize them, and then upload them in permuted order. We instead chose to analyze a batch mix that mixes a fixed number of ciphertexts. We consider this case since: (1) it is concrete in the sense that a fixed size vector of ciphertexts needs to be anonymized and this gives a precise level of anonymity (fixed-size random permutation), and (2) we achieve low-latency since once the batch forms at the first mix the ciphertexts are pushed through the cascade of mixes rapidly.

We point out that the security arguments of Golle et al for their proposed mixes are flawed:

1. **Not tied to DDH**: None of the proofs in the paper take a DDH problem instance as input. It follows that they did not prove that security holds under DDH.
2. **Not randomized reductions**: None of the input problem instances in the paper are randomized. It is well-known that randomized reductions are stronger than non-randomized ones.

Consequently the security of their mixes were not tied to the DDH problem as claimed. This left as open the problem of proving the security of universal re-encryption batch mixing. We solve this problem in this section.

Informally, the problem we consider is to establish an externally anonymous communication channel. A set of w senders $s_1, s_2, ..., s_w$ want to send messages $m_1, m_2, ..., m_w$ respectively, to a target set of w receivers $r_1, r_2, ..., r_w$. Consider the case that s_i sends a message to s_j where $i, j \in \{1, 2, ..., w\}$. We want an eavesdropper to have negligible advantage in correlating the initial ciphertext that s_i sends out with the public key of r_j. In other words, the eavesdropper has negligible advantage over guessing the receiver.

The solution must be forward-anonymous: an adversary that compromises a mix server cannot break the anonymity of previously transmitted ciphertexts. The solution must be robust in that anonymity holds as long as there is at least one mix server not compromised by the adversary.

Note that a receiver of a message can determine who the sender of the message is. The receiver is able to decipher the ciphertext right when the sender transmits it to the first mix. Anonymity is against external adversaries.

B.1 Definition of security

Definition 7. *A **forward-anonymous batch mix** protocol, denoted by* FBMIX*, is a 4-tuple of algorithms* FBGEN*,* FBENCR*,* FBMIXER*, and* FBDECR *where* FBGEN *generates a key pair for each receiver, where* FBENCR *encrypts the messages of the senders, where the* FBMIXER *servers are connected in series and they mix received ciphertexts and forward them on, that satisfies the following properties for all probabilistic polynomial-time passive adversaries $\mathcal{A}$:*

1. FBENCR *Confidentiality: The ciphertexts output by algorithm* FBENCR *satisfy the message indistinguishability property with respect to $\mathcal{A}$ (Definition 10).*
2. FBMIXER *Confidentiality: The ciphertexts output by* FBMIXER *satisfy message indistinguishability with respect to $\mathcal{A}$ (Definition 11)*
3. FBENCR *Anonymity: The ciphertexts output by* FBENCR *satisfy key anonymity with respect to $\mathcal{A}$ (Definition 8).*
4. FBMIXER *Anonymity: The ciphertexts output by algorithm* FBMIXER *satisfy anonymity with respect to $\mathcal{A}$ (Definition 9).*
5. *Forward-Anonymity: The* FBMIXER *servers have no secret key material.*
6. *Robustness: Anonymity of* FBMIX *holds provided at least one* FBMIXER *server is not compromised by $\mathcal{A}$.*
7. *Completeness: $\forall\ i \in \{1, 2, ..., w\}$, when sender s_i sends m_i to r_j where $j \in \{1, 2, ..., w\}$ then r_j receives m_i.*
8. *Low-Latency: Once w ciphertexts arrive at the first* FBMIXER *server, the batch moves through the mix at a speed limited only by the time to re-encrypt, permute, and forward.*

B.2 Forward-anonymous batch mix construction

We instantiate the mix using security parameter n as follows.

FBGEN$(n, ((p, q), g))$:
1. $(y_i, x_i) \leftarrow$ UKG$(n, ((p, q), g))$ for $i = 1, 2, ..., w$
2. output $((y_1, x_1), (y_2, x_2), ..., (y_w, x_w))$

Let σ_i be the index of the receiver of the message of sender i. For example, if s_1 sends to s_3 then $\sigma_1 = 3$.

FBENCR$((m_1, (k_{1,0}, k_{1,1}), \sigma_1), ..., (m_w, (k_{w,0}, k_{w,1}), \sigma_w), y_1, y_2, ..., y_w, ((p, q), g))$:
1. $c_i \leftarrow \mathtt{UE}_{y_{\sigma_i}}(m_i, (k_{i,0}, k_{i,1}), ((p, q), g))$ for $i = 1, 2, ..., w$
2. output $(c_1, c_2, ..., c_w)$

Define set S to be $\{1, 2, ..., w\}$. Let π be a permutation from S onto S. Define $\mathtt{fp}(\pi, c_1, c_2, ..., c_w)$ to be a function that outputs $(c_{\pi(1)}, c_{\pi(2)}, ..., c_{\pi(w)})$. Let the algorithm $\mathtt{fpinv}(\pi, c_{\pi(1)}, c_{\pi(2)}, ..., c_{\pi(w)})$ be a function that uses π^{-1} to output the tuple $(c_1, c_2, ..., c_w)$.

FBMIXER$(\pi, (c_1,(\ell_{1,0}, \ell_{1,1})), (c_2, (\ell_{2,0}, \ell_{2,1})), ..., (c_w, (\ell_{w,0}, \ell_{w,1})), ((p, q), g))$:
1. $c'_i \leftarrow \mathtt{URe}(c_i, (\ell_{i,0}, \ell_{i,1}), ((p, q), g))$ for $i = 1, 2, ..., w$
2. output $\mathtt{fp}(\pi, c'_1, c'_2, ..., c'_w)$

The `break` statement terminates the execution of the nearest enclosing `for` loop in which `break` appears.

`FBDECR`$(c_1, c_2, ..., c_w, x_1, x_2, ..., x_w, ((p, q), g))$:
let L be the empty list
`for` i in 1 to w:
  `for` j in 1 to w:
    $(m, s) \leftarrow \texttt{UD}_{x_j}(c_i, ((p, q), g))$
    if $s =$ true
      append (m, j) to L
      `break`
output L

There are four stages in the mix protocol. The mix protocol leverages N mix servers labeled $1, 2, ..., N$ and they are connected in series.

Stage 1: r_j generates a key pair (y_j, x_j) using `UKG` and publishes y_j for $j = 1, 2, ..., w$. This stage is effectively `FBGEN`.

Stage 2: Sender s_i formulates a message m_i to send to receiver r_j. s_i generates $(k_{i,0}, k_{i,1}) \in_U [1, q] \times [1, q]$ and computes $c_i \leftarrow \texttt{UE}_{y_{\sigma_i}}(m_i, (k_{i,0}, k_{i,1}), ((p, q), g))$. s_i sends c_i to Mix 1 for $i = 1, 2, ..., w$. This stage is effectively `FBENCR`.

Stage 3: Mix k where $1 \leq k \leq N$ operates as follows. It waits until a full batch of w ciphertexts $c_1, c_2, ..., c_w$ arrive. It then generates $(\ell_{i,0}, \ell_{i,1}) \in_U [1, q] \times [1, q]$ for $i = 1, 2, ..., w$. It generates a permutation π from S onto S uniformly at random. It then computes,

$$(c'_{\pi(1)}, c'_{\pi(2)}, ..., c'_{\pi(w)}) \leftarrow \texttt{FBMIXER}(\pi, \\ (c_1, (\ell_{1,0}, \ell_{1,1})), (c_2, (\ell_{2,0}, \ell_{2,1})), ..., (c_w, (\ell_{w,0}, \ell_{w,1})), ((p, q), g))$$

If $k < N$ then $(c'_{\pi(1)}, c'_{\pi(2)}, ..., c'_{\pi(w)})$ is sent to mix $k + 1$. If $k = N$ then $(c'_{\pi(1)}, c'_{\pi(2)}, ..., c'_{\pi(w)})$ is posted to a public bulletin board. Each of these mixes is effectively `FBMIXER`.

Stage 4: r_j for $j = 1, 2, ..., w$ downloads all w ciphertexts from the bulletin board. r_j attempts decryption of every single one of the ciphertexts using x_j. In so doing, r_j receives zero or more messages. If there is no i for which $\sigma_i = j$ then r_j receives no messages. This stage is effectively `FBDECR`.

We can improve the performance of Stage 4 in the case that every receiver gets only one message from a sender. In this scenario, a receiver can pull down the ciphertexts from the bulletin board one by one and then stop when a ciphertext is received that properly decrypts. The batch mix provides external anonymity thereby breaking the link between senders and receivers. This use case would fail completely were the senders to post their key anonymous ciphertexts directly to the bulletin board. To see this, note that a passive eavesdropper would know the sender of each ciphertext on the bulletin board. The eavesdropper would then know who the receiver is of a given ciphertext based on when the receiver stops pulling down ciphertexts.

B.3 Security of FBMIX

Where possible we allow the adversary to choose the receivers of messages in FBMIX. For example, the adversary can have Alice and Bob send messages to the same receiver, Carol. Consequently, many senders can send messages to the same receiver. As a result we need to generalize DDHRerand5 from Section 7. It generalizes to produce more DH 3-tuples with a common "public key" in the same way that the DDH random self-reduction generalized to form DDHRerand5.

To make the pattern clear we define DDHRerand7 as follows. The algorithm DDHRerand7$((p,q),g,x,y,z)$ randomizes a DDH problem instance by choosing the exponents $u_1, u_2, v, v', v'', u_1', u_1'' \in_U [1,q]$ and computing,

$$(x''', x'', x', y', z', z'', z''') \leftarrow (x^{v''} g^{u_1''}, x^{v'} g^{u_1'}, x^{v} g^{u_1}, y g^{u_2},\\ z^v y^{u_1} x^{v u_2} g^{u_1 u_2}, z^{v'} y^{u_1'} x^{v' u_2} g^{u_1' u_2}, z^{v''} y^{u_1''} x^{v'' u_2} g^{u_1'' u_2})$$

and so on for ever more "v primes" and "u_1 primes".

For ease of use we parameterize this DDH generalization as follows. Let DDHRerandN$((p,q),g,x,y,z,t)$ be a DDH self-reduction algorithm that outputs a set T containing t 3-tuples. Define, the set $T = \{(A_1,B_1,R_1),(A_2,B_2,R_2), ...,(A_t,B_t,R_t)\}$.

The algorithm has these properties: (1) when the input (x,y,z) is a DH 3-tuple then all t output 3-tuples are random DH 3-tuples but with the middle term in common, and (2) when the input (x,y,z) is not a DH 3-tuple then A_1,A_2,...,A_t,B_1,R_1,R_2,...,$R_t \in_U G_{\mathtt{p}}$ and $B_1 = B_2 = ... = B_t$.

DDHRerandN$((p,q),g,x,y,z,2)$ is logically equivalent to DDHRerand5. To see this, note that the algorithm DDHRerandN$((p,q),g,x,y,z,2)$ outputs the set of tuples $T = \{(A_1,B_1,R_1),(A_2,B_2,R_2)\}$ which, rearranging and dropping the B_2 yields the 5-tuple (A_1,A_2,B_1,R_2,R_1). Observe that $B_1 = B_2$.

Let GetMiddle(T) to be a function that on input a set T that is output by DDHRerandN, selects a tuple in T and returns the middle value in it. All middle values are the same so it doesn't matter which tuple is selected. We now address key anonymity for FBENCR.

Definition 8. *If $\forall$ probabilistic polynomial time adversaries $\mathcal{A}$, $\forall$ $\alpha > 0$, $\forall$ $i \in \{1,2,...,w\}$, and $\forall$ sufficiently large n, after the following,*

1. *generate* $((p,q),g) \leftarrow \mathcal{IG}(n)$
2. $((y_1,x_1),(y_2,x_2),...,(y_w,x_w)) \leftarrow$ FBGEN$(n,((p,q),g))$
3. $(m_1,m_2,...,m_w) \leftarrow \mathcal{A}(((p,q),g),y_1,y_2,...,y_w,$ *"specify messages"*$)$
4. *if* $\exists$ $j \in \{1,2,...,w\}$ *such that* $m_j \notin G_{\mathtt{p}}$ *then output "false" and halt*
5. $(k_{j,0},k_{j,1}) \in_U [1,q] \times [1,q]$ *for* $j = 1,2,...,w$
6. $\sigma_j \in_U \{1,2,...,w\}$ *for* $j = 1,2,...,w$
7. $(c_1,c_2,...,c_w) \leftarrow$ FBENCR$((m_1,(k_{1,0},k_{1,1}),\sigma_1),...,$
   $(m_w,(k_{w,0},k_{w,1}),\sigma_w),y_1,y_2,...,y_w,((p,q),g))$
8. $(\sigma_1',\sigma_2',...,\sigma_w') \leftarrow \mathcal{A}(c_1,c_2,...,c_w,$ *"guess"*$)$
9. *if* $\sigma_i = \sigma_i'$ *then output "true" else output "false"*

the output of the experiment is "true" with probability less than $\frac{1}{w} + \frac{1}{n^\alpha}$ *then* `FBENCR` *is secure in the sense of key anonymity.*

Theorem 7. *If DDH is hard then algorithm* `FBENCR` *is secure in the sense of key anonymity.*

Proof. Suppose there exists a probabilistic polynomial time adversary $\mathcal{A}$, an $\alpha > 0$, an $i \in \{1, 2, ..., w\}$, and a sufficiently large n, such that $\mathcal{A}$ succeeds with probability greater than or equal to $\frac{1}{w} + \frac{1}{n^\alpha}$. Consider algorithm `AlgR9` that takes as input a DDH problem instance $((p, q), g, a_0, b_0, c_0)$.

`AlgR9`$((p, q), g, a_0, b_0, c_0)$:
1. $T_j \leftarrow$ `DDHRerandN`$((p, q), g, a_0, b_0, c_0, 2w)$ for $j = 1, 2, ..., w$
2. set $y_j =$ `GetMiddle`(T_j) for $j = 1, 2, ..., w$
3. $(m_1, m_2, ..., m_w) \leftarrow \mathcal{A}(((p, q), g), y_1, y_2, ..., y_w,$ "specify messages"$)$
4. if $\exists\ j \in \{1, 2, ..., w\}$ such that $m_j \notin G_{\mathtt{p}}$ then output "false" and halt
5. $\sigma_j \in_U \{1, 2, ..., w\}$ for $j = 1, 2, ..., w$
6. for j in $1..w$ do:
7.   extract a tuple (A_0, B_0, R_0) without replacement from T_{σ_j}
8.   extract a tuple (A_1, B_1, R_1) without replacement from T_{σ_j}
9. $c_j \leftarrow ((m_j R_0, A_0), (R_1, A_1))$
10. $(\sigma'_1, \sigma'_2, ..., \sigma'_w) \leftarrow \mathcal{A}(c_1, c_2, ..., c_w,$"guess"$)$
11. if $\sigma_i = \sigma'_i$ then output "true" else output "false"'

Consider the case that the input is a DH 3-tuple. It follows from the definition of `DDHRerandN` that c_j is a proper encryption of m_j using public key y_{σ_j} for $j = 1, 2, ..., w$ under `FBENCR`. Therefore, the input to $\mathcal{A}$ is drawn from the same set and probability distribution as the input to $\mathcal{A}$ in Definition 8. It follows that $\sigma_i = \sigma'_i$ with probability greater than or equal to $\frac{1}{w} + \frac{1}{n^\alpha}$. So, for random exponents a and b in $[1, q]$, $\Pr[$`AlgR9`$((p, q), g, g^a, g^b, g^{ab}) =$ "true"$] \geq \frac{1}{w} + \frac{1}{n^\alpha}$. Define $\psi = \Pr[$`AlgR9`$((p, q), g, g^a, g^b, g^{ab}) =$ "true"$]$.

Now consider the case that the input is not a DH 3-tuple. It follows from the definition of `DDHRerandN` that c_j is uniformly distributed in $G_{\mathtt{p}}^2 \times G_{\mathtt{p}}^2$ and y_j is uniformly distributed in $G_{\mathtt{p}}$ for $j = 1, 2, ..., w$. Let p_j be the probability that $\mathcal{A}$ responds with $\sigma'_i = j$ for $j = 1, 2, ..., w$. Then the probability that $\sigma_i = \sigma'_i$ is $\frac{1}{w}p_1 + \frac{1}{w}p_2 + ... + \frac{1}{w}p_w = \frac{1}{w}$. So, for randomly chosen exponents a, b, and c in $[1, q]$, $\Pr[$`AlgR9`$((p, q), g, g^a, g^b, g^c) =$ "true"$] = \frac{q^2}{q^3}\psi + (1 - \frac{q^2}{q^3})\frac{1}{w}$ which is overwhelmingly close to $\frac{1}{w}$. $\square$

We now address key anonymity for `FBMIXER`.

Definition 9. *If* $\forall$ *probabilistic polynomial time adversaries* $\mathcal{A}$*,* $\forall\ \alpha > 0$*,* $\forall\ i \in \{1, 2, ..., w\}$*, and* $\forall$ *sufficiently large* n*, after the following,*

1. *generate* $((p, q), g) \leftarrow \mathcal{IG}(n)$
2. $((y_1, x_1), (y_2, x_2), ..., (y_w, x_w)) \leftarrow$ `FBGEN`$(n, ((p, q), g))$
3. $((m_1, r_1, \sigma_1), (m_2, r_2, \sigma_2), ..., (m_w, r_w, \sigma_w)) \leftarrow \mathcal{A}(((p, q), g),$

$y_1, y_2, ..., y_w$, "specify ciphertexts and receivers")
4. if $\exists\ j \in \{1, 2, ..., w\}$ such that $m_j \notin G_{\mathtt{p}}$ then output "false" and halt
5. if $\exists\ j \in \{1, 2, ..., w\}$ such that $r_j \notin [1, q] \times [1, q]$ then output "false" and halt
6. if $\exists\ j \in \{1, 2, ..., w\}$ such that $\sigma_j \notin S$ then output "false" and halt
7. $(c_1, c_2, ..., c_w) \leftarrow \mathtt{FBENCR}((m_1, r_1, \sigma_1), ..., (m_w, r_w, \sigma_w), y_1, y_2, ..., y_w, ((p, q), g))$
8. $\mu_j \in_U [1, q] \times [1, q]$ for $j = 1, 2, ..., w$
9. select a permutation π from S onto S uniformly at random
10. $(c'_{\pi(1)}, c'_{\pi(2)}, ..., c'_{\pi(w)}) \leftarrow \mathtt{FBMIXER}(\pi, (c_1, \mu_1), (c_2, \mu_2), ..., (c_w, \mu_w), ((p, q), g))$
11. $\pi' \leftarrow \mathcal{A}(c'_{\pi(1)}, c'_{\pi(2)}, ..., c'_{\pi(w)}$, "guess")
12. if $\pi'(i) = \pi(i)$ then output "true" else output "false"

the output of the experiment is "true" with probability less than $\frac{1}{w} + \frac{1}{n^\alpha}$ then FBMIXER *is secure in the sense of anonymity.*

Theorem 8. *If DDH is hard then* FBMIXER *is secure in the sense of anonymity.*

Proof. Suppose there exists a probabilistic polynomial time adversary $\mathcal{A}$, an $\alpha > 0$, an $i \in \{1, 2, ..., w\}$, and a sufficiently large n, such that $\mathcal{A}$ succeeds with probability greater than or equal to $\frac{1}{w} + \frac{1}{n^\alpha}$. Consider algorithm AlgR10 that takes as input a DDH problem instance $((p, q), g, a_0, b_0, c_0)$.

AlgR10$((p, q), g, a_0, b_0, c_0)$:
1. $T_j \leftarrow \mathtt{DDHRerandN}((p, q), g, a_0, b_0, c_0, 2w)$ for $j = 1, 2, ..., w$
2. set $y_j = \mathtt{GetMiddle}(T_j)$ for $j = 1, 2, ..., w$
3. $((m_1, r_1, \sigma_1), (m_2, r_2, \sigma_2), ..., (m_w, r_w, \sigma_w)) \leftarrow \mathcal{A}(((p, q), g), y_1, y_2, ..., y_w,$ "specify ciphertexts and receivers")
4. if $\exists\ j \in \{1, 2, ..., w\}$ such that $m_j \notin G_{\mathtt{p}}$ then output "false" and halt
5. if $\exists\ j \in \{1, 2, ..., w\}$ such that $r_j \notin [1, q] \times [1, q]$ then output "false" and halt
6. if $\exists\ j \in \{1, 2, ..., w\}$ such that $\sigma_j \notin S$ then output "false" and halt
7. $(c_1, c_2, ..., c_w) \leftarrow \mathtt{FBENCR}((m_1, r_1, \sigma_1), ..., (m_w, r_w, \sigma_w), y_1, y_2, ..., y_w, ((p, q), g))$
8. for j in $1..w$ do:
9. extract a tuple (A_0, B_0, R_0) without replacement from T_{σ_j}
10. extract a tuple (A_1, B_1, R_1) without replacement from T_{σ_j}
11. $((\alpha_0, \beta_0), (\alpha_1, \beta_1)) \leftarrow c_j$
12. $c'_j \leftarrow ((\alpha_0 R_0, \beta_0 A_0), (\alpha_1 R_1, \beta_1 A_1))$
13. select a permutation π from S onto S uniformly at random
14. $(c'_{\pi(1)}, c'_{\pi(2)}, ..., c'_{\pi(w)}) \leftarrow \mathtt{fp}(\pi, c'_1, c'_2, ..., c'_w)$
15. $\pi' \leftarrow \mathcal{A}(c'_{\pi(1)}, c'_{\pi(2)}, ..., c'_{\pi(w)}$, "guess")
16. if $\pi'(i) = \pi(i)$ then output "true" else output "false"

Consider the case that the input is a DH 3-tuple. Clearly the ciphertexts $c_1, c_2, ..., c_w$ are as specified by $\mathcal{A}$. It follows from the definition of DDHRerandN that c'_j is a proper re-encryption of c_j under FBMIXER for $j = 1, 2, ..., w$. Therefore, the input to $\mathcal{A}$ is drawn from the same set and probability distribution as the input to $\mathcal{A}$ in Definition 9. It follows that $\pi'(i) = \pi(i)$ with probability greater than or equal to $\frac{1}{w} + \frac{1}{n^\alpha}$. So, for random exponents a and b in

$[1, q]$, $\Pr[\texttt{AlgR10}((p, q), g, g^a, g^b, g^{ab}) = \text{“true”}] \geq \frac{1}{w} + \frac{1}{n^\alpha}$. Define the value $\psi = \Pr[\texttt{AlgR10}((p, q), g, g^a, g^b, g^{ab}) = \text{“true”}]$.

Now consider the case that the input is not a DH 3-tuple. It follows from the definition of DDHRerandN that y_j is uniformly distributed in $G_{\mathtt{p}}$ for $j = 1, 2, ..., w$ and that c'_j is uniformly distributed in $G_{\mathtt{p}}^2 \times G_{\mathtt{p}}^2$ for $j = 1, 2, ..., w$. Let p_j be the probability that $\mathcal{A}$ responds with $\pi'(i) = j$ for $j = 1, 2, ..., w$. Then the probability that $\pi'(i) = \pi(i)$ is $\frac{1}{w}p_1 + \frac{1}{w}p_2 + ... + \frac{1}{w}p_w = \frac{1}{w}$. So, for randomly chosen exponents a, b, and c in $[1, q]$, $\Pr[\texttt{AlgR10}((p, q), g, g^a, g^b, g^c) = \text{“true”}] = \frac{q^2}{q^3}\psi + (1 - \frac{q^2}{q^3})\frac{1}{w}$ which is overwhelmingly close to $\frac{1}{w}$. □

We now address message indistinguishability.

Definition 10. *If $\forall$ probabilistic polynomial time adversaries $\mathcal{A}$, $\forall\ \alpha > 0$, $\forall\ i \in \{1, 2, ..., w\}$, and $\forall$ sufficiently large n, after the following,*

1. *generate $((p, q), g) \leftarrow \mathcal{IG}(n)$*
2. *$((y_1, x_1), (y_2, x_2), ..., (y_w, x_w)) \leftarrow \texttt{FBGEN}(n, ((p, q), g))$*
3. *$((m_{1,0}, m_{1,1}, \sigma_1), (m_{2,0}, m_{2,1}, \sigma_2), ..., (m_{w,0}, m_{w,1}, \sigma_w))$
   $\leftarrow \mathcal{A}(((p, q), g), y_1, y_2, ..., y_w,$ “specify messages and receivers”$)$*
4. *if $\exists\ j \in \{1, 2, ..., w\}$ such that $(m_{j,0} \notin G_{\mathtt{p}}$ or $m_{j,1} \notin G_{\mathtt{p}})$
   then output “false” and halt*
5. *if $\exists\ j \in \{1, 2, ..., w\}$ such that $m_{j,0} = m_{j,1}$ then output “false” and halt*
6. *if $\exists\ j \in \{1, 2, ..., w\}$ such that $\sigma_j \notin \{1, 2, ..., w\}$ then output “false” and halt*
7. *$(k_{j,0}, k_{j,1}) \in_U [1, q] \times [1, q]$ for $j = 1, 2, ..., w$*
8. *$b_j \in_U \{0, 1\}$ for $j = 1, 2, ..., w$*
9. *$(c_1, c_2, ..., c_w) \leftarrow \texttt{FBENCR}((m_{1,b_1}, (k_{1,0}, k_{1,1}), \sigma_1), ...,$
   $(m_{w,b_w}, (k_{w,0}, k_{w,1}), \sigma_w), y_1, y_2, ..., y_w, ((p, q), g))$*
10. *$(b'_1, b'_2, ..., b'_w) \leftarrow \mathcal{A}(c_1, c_2, ..., c_w,$ “guess”$)$*
11. *if $b_i = b'_i$ then output “true” else output “false”*

the output of the experiment is “true” with probability less than $\frac{1}{2} + \frac{1}{n^\alpha}$ then FBENCR *is secure in the sense of message indistinguishability.*

Theorem 9. *If DDH is hard then* FBENCR *is secure in the sense of message indistinguishability.*

Proof. Suppose there exists a probabilistic polynomial time adversary $\mathcal{A}$, an $\alpha > 0$, an $i \in \{1, 2, ..., w\}$, and a sufficiently large n, such that $\mathcal{A}$ succeeds with probability greater than or equal to $\frac{1}{2} + \frac{1}{n^\alpha}$. Consider algorithm AlgR7 that takes as input a DDH problem instance $((p, q), g, a_0, b_0, c_0)$.

$\texttt{AlgR7}((p, q), g, a_0, b_0, c_0)$:
1. $T_j \leftarrow \texttt{DDHRerandN}((p, q), g, a_0, b_0, c_0, 2w)$ for $j = 1, 2, ..., w$
2. set $y_j = \texttt{GetMiddle}(T_j)$ for $j = 1, 2, ..., w$
3. $((m_{1,0}, m_{1,1}, \sigma_1), (m_{2,0}, m_{2,1}, \sigma_2), ..., (m_{w,0}, m_{w,1}, \sigma_w))$

$\leftarrow \mathcal{A}((((p,q),g),y_1,y_2,...,y_w,$ "specify messages and receivers")

4. if $\exists\ j \in \{1,2,...,w\}$ such that $(m_{j,0} \notin G_{\mathtt{p}}$ or
   $m_{j,1} \notin G_{\mathtt{p}})$ then output "false" and halt
5. if $\exists\ j \in \{1,2,...,w\}$ such that $m_{j,0} = m_{j,1}$ then output "false" and halt
6. if $\exists\ j \in \{1,2,...,w\}$ such that $\sigma_j \notin \{1,2,...,w\}$ then output "false" and halt
7. $b_j \in_U \{0,1\}$ for $j = 1,2,...,w$
8. for j in $1..w$ do:
9. extract a tuple (A_0, B_0, R_0) without replacement from T_{σ_j}
10. extract a tuple (A_1, B_1, R_1) without replacement from T_{σ_j}
11. $c_j \leftarrow ((m_{j,b_j} R_0, A_0), (R_1, A_1))$
12. $(b'_1, b'_2, ..., b'_w) \leftarrow \mathcal{A}(c_1, c_2, ..., c_w,$"guess")
13. if $b_i = b'_i$ then output "true" else output "false"'

Consider the case that the input is a DH 3-tuple. It follows from the definition of `DDHRerandN` that c_j is a proper encryption of m_{j,b_j} using public key y_{σ_j} for $j = 1,2,...,w$ under `FBENCR`. Therefore, the input to $\mathcal{A}$ is drawn from the same set and probability distribution as the input to $\mathcal{A}$ in Definition 10. It follows that $b_i = b'_i$ with probability greater than or equal to $\frac{1}{2} + \frac{1}{n^\alpha}$. So, for random exponents a and b in $[1,q]$, $\Pr[\mathtt{AlgR7}((p,q),g,g^a,g^b,g^{ab}) = \text{"true"}] \geq \frac{1}{2} + \frac{1}{n^\alpha}$. Define $\psi = \Pr[\mathtt{AlgR7}((p,q),g,g^a,g^b,g^{ab}) = \text{"true"}]$.

Now consider the case that the input is not a DH 3-tuple. It follows from the definition of `DDHRerandN` that c_j is uniformly distributed in $G_{\mathtt{p}}^2 \times G_{\mathtt{p}}^2$ and y_j is uniformly distributed in $G_{\mathtt{p}}$ for $j = 1,2,...,w$. Let p_1 be the probability that $\mathcal{A}$ responds with $b'_i = 0$. Then the probability that $b_i = b'_i$ is $\frac{1}{2}p_1 + \frac{1}{2}(1 - p_1) = \frac{1}{2}$. So, for randomly chosen exponents a, b, and c in $[1,q]$, the probability $\Pr[\mathtt{AlgR7}((p,q),g,g^a,g^b,g^c) = \text{"true"}] = \frac{q^2}{q^3}\psi + (1 - \frac{q^2}{q^3})\frac{1}{2}$ which is overwhelmingly close to $\frac{1}{2}$. □

Definition 11. *If $\forall$ probabilistic polynomial time adversaries $\mathcal{A}$, $\forall\ \alpha > 0$, $\forall\ i \in \{1,2,...,w\}$, and $\forall$ sufficiently large n, after the following,*

1. *generate* $((p,q),g) \leftarrow \mathcal{IG}(n)$
2. $((y_1,x_1),(y_2,x_2),...,(y_w,x_w)) \leftarrow \mathtt{FBGEN}(n,((p,q),g))$
3. $(\pi,(m_{1,0},m_{1,1},r_{1,0},r_{1,1},\sigma_1),(m_{2,0},m_{2,1},r_{2,0},r_{2,1},\sigma_2),$
   $...,(m_{w,0},m_{w,1},r_{w,0},r_{w,1},\sigma_w)) \leftarrow \mathcal{A}(((p,q),g),y_1,y_2,$
   $...,y_w,$*"specify ciphertexts, receivers, and π"*)
4. *if π is not a permutation from S onto S then output "false" and halt*
5. *if $\exists\ j \in \{1,2,...,w\}$ such that $(m_{j,0} \notin G_{\mathtt{p}}$ or $m_{j,1} \notin G_{\mathtt{p}})$*
   then output "false" and halt
6. *if $\exists\ j \in \{1,2,...,w\}$ such that $m_{j,0} = m_{j,1}$ then output "false" and halt*
7. *if $\exists\ j \in \{1,2,...,w\}$ such that $(r_{j,0} \notin [1,q] \times [1,q]$ or*
   $r_{j,1} \notin [1,q] \times [1,q])$ then output "false" and halt
8. *if $\exists\ j \in \{1,2,...,w\}$ such that $\sigma_j \notin \{1,2,...,w\}$ then output "false" and halt*
9. *$b_j \in_U \{0,1\}$ for $j = 1,2,...,w$*
10. $(c_1,c_2,...,c_w) \leftarrow \mathtt{FBENCR}((m_{1,b_1},r_{1,b_1},\sigma_1),...,$

$(m_{w,b_w}, r_{w,b_w}, \sigma_w), y_1, y_2, ..., y_w, ((p,q), g))$
11. $r_j \in_U [1,q] \times [1,q]$ *for* $j = 1, 2, ..., w$
12. $(c'_{\pi(1)}, c'_{\pi(2)}, ..., c'_{\pi(w)}) \leftarrow \texttt{FBMIXER}(\pi, (c_1, r_1), (c_2, r_2), ..., (c_w, r_w), ((p,q), g))$
13. $(c'_1, c'_2, ..., c'_w) \leftarrow \texttt{fpinv}(\pi, c'_{\pi(1)}, c'_{\pi(2)}, ..., c'_{\pi(w)})$
14. $(b'_1, b'_2, ..., b'_w) \leftarrow \mathcal{A}(c'_1, c'_2, ..., c'_w,$ *"guess"*$)$
15. if $b_i = b'_i$ *then output "true" else output "false"*

the output of the experiment is "true" with probability less than $\frac{1}{2} + \frac{1}{n^\alpha}$ *then* `FBMIXER` *is secure in the sense of message indistinguishability.*

Theorem 10. *If DDH is hard then* `FBMIXER` *is secure in the sense of message indistinguishability.*

Proof. Suppose there exists a probabilistic polynomial time adversary $\mathcal{A}$, an $\alpha > 0$, an $i \in \{1, 2, ..., w\}$, and a sufficiently large n, such that $\mathcal{A}$ succeeds with probability greater than or equal to $\frac{1}{2} + \frac{1}{n^\alpha}$. Consider algorithm `AlgR8` that takes as input a DDH problem instance $((p,q), g, a_0, b_0, c_0)$.

`AlgR8`$((p,q), g, a_0, b_0, c_0)$:
1. $T_j \leftarrow \texttt{DDHRerandN}((p,q), g, a_0, b_0, c_0, 2w)$ for $j = 1, 2, ..., w$
2. set $y_j = \texttt{GetMiddle}(T_j)$ for $j = 1, 2, ..., w$
3. $(\pi, (m_{1,0}, m_{1,1}, r_{1,0}, r_{1,1}, \sigma_1), (m_{2,0}, m_{2,1}, r_{2,0}, r_{2,1}, \sigma_2),$
   $..., (m_{w,0}, m_{w,1}, r_{w,0}, r_{w,1}, \sigma_w)) \leftarrow \mathcal{A}(((p,q), g), y_1, y_2,$
   $..., y_w,$"specify ciphertexts, receivers, and π")
4. if π is not a permutation from S onto S then output "false" and halt
5. if $\exists\ j \in \{1, 2, ..., w\}$ such that $(m_{j,0} \notin G_{\mathtt{p}}$ or
   $m_{j,1} \notin G_{\mathtt{p}})$ then output "false" and halt
6. if $\exists\ j \in \{1, 2, ..., w\}$ such that $m_{j,0} = m_{j,1}$ then output "false" and halt
7. if $\exists\ j \in \{1, 2, ..., w\}$ such that $(r_{j,0} \notin [1,q] \times [1,q]$ or
   $r_{j,1} \notin [1,q] \times [1,q])$ then output "false" and halt
8. if $\exists\ j \in \{1, 2, ..., w\}$ such that $\sigma_j \notin \{1, 2, ..., w\}$ then
   output "false" and halt
9. $b_j \in_U \{0, 1\}$ for $j = 1, 2, ..., w$
10. $(c_1, c_2, ..., c_w) \leftarrow \texttt{FBENCR}((m_{1,b_1}, r_{1,b_1}, \sigma_1), ...,$
    $(m_{w,b_w}, r_{w,b_w}, \sigma_w), y_1, y_2, ..., y_w, ((p,q), g))$
11. for j in 1..w do:
12. extract a tuple (A_0, B_0, R_0) without replacement from T_{σ_j}
13. extract a tuple (A_1, B_1, R_1) without replacement from T_{σ_j}
14. $((\alpha_0, \beta_0), (\alpha_1, \beta_1)) \leftarrow c_j$
15. $c'_j \leftarrow ((\alpha_0 R_0, \beta_0 A_0), (\alpha_1 R_1, \beta_1 A_1))$
16. $(b'_1, b'_2, ..., b'_w) \leftarrow \mathcal{A}(c'_1, c'_2, ..., c'_w,$"guess"$)$
17. if $b_i = b'_i$ then output "true" else output "false"

Consider the case that the input is a DH 3-tuple. Clearly the ciphertexts $c_1, c_2, ..., c_w$ are as specified by $\mathcal{A}$. It follows from the definition of `DDHRerandN` that c'_j is a proper re-encryption of c_j under `FBMIXER` for $j = 1, 2, ..., w$. Therefore, the input to adversary $\mathcal{A}$ is drawn from the same set and probability distribution as the input to $\mathcal{A}$ in Definition 11. It follows that $b_i = b'_i$ with probability greater than or equal to $\frac{1}{2} + \frac{1}{n^\alpha}$. So, for random exponents a and b in $[1, q]$, $\Pr[\texttt{AlgR8}((p, q), g, g^a, g^b, g^{ab}) = \text{“true”}] \geq \frac{1}{2} + \frac{1}{n^\alpha}$. Define the value $\psi = \Pr[\texttt{AlgR8}((p, q), g, g^a, g^b, g^{ab}) = \text{“true”}]$.

Now consider the case that the input is not a DH 3-tuple. It follows from the definition of `DDHRerandN` that y_j is uniformly distributed in $G_{\mathtt{p}}$ for $j = 1, 2, ..., w$ and that c'_j is uniformly distributed in $G_{\mathtt{p}}^2 \times G_{\mathtt{p}}^2$ for $j = 1, 2, ..., w$. Let p_1 be the probability that $\mathcal{A}$ responds with $b'_i = 0$. Then the probability that $b_i = b'_i$ is $\frac{1}{2}p_1 + \frac{1}{2}(1 - p_1) = \frac{1}{2}$. So, for randomly chosen exponents a, b, and c in $[1, q]$, the probability $\Pr[\texttt{AlgR8}((p, q), g, g^a, g^b, g^c) = \text{“true”}] = \frac{q^2}{q^3}\psi + (1 - \frac{q^2}{q^3})\frac{1}{2}$ which is overwhelmingly close to $\frac{1}{2}$. $\square$

Theorems 7, 8, 9, and 10 show that properties 1, 2, 3, and 4 of a forward-anonymous batch mix hold, respectively. The `FBMIXER` servers store no keys at all so the forward-anonymity property holds (property 5). Theorem 8 proves that anonymity holds from a single honest mix. Therefore, the robustness property holds (property 6). Completeness and low-latency are straightforward to show (properties 7 and 8). Theorem 6 therefore holds.