Semantically Secure Anonymity: Foundations of Re-Encryption

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

[1] Cryptovirology Labs
[2] 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 first decrypting it, and without knowing who the receiver is (i.e., not knowing the public key used to create it). 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 a subtlety in it, namely, that it does not require that the encryption function achieve key anonymity. Recall that the encryption function is different from the re-encryption function. We demonstrate this subtlety by constructing a cryptosystem that satisfies the established definition of a universal cryptosystem but that has an encryption function that does not achieve key anonymity, thereby instantiating the gap in the definition of security of universal re-encryption. We note that the gap in the definition carries over to a set of applications that rely on universal re-encryption, applications in the original paper on universal re-encryption and also follow-on work. This shows that the original definition needs to be corrected and it shows that it had a knock-on effect that negatively impacted security in later work. We then introduce a new definition that includes the properties that are needed for a re-encryption cryptosystem to achieve key anonymity in *both* the encryption function and the re-encryption function, building on Goldwasser and Micali's "semantic security" and the original "key anonymity" notion of Bellare, Boldyreva, Desai, and Pointcheval. Omitting any of the properties in our definition leads to a problem. We also introduce a new generalization of the 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. We apply our new DDH reduction technique to give the first proof in the standard model that ElGamal-based incomparable public keys achieve key anonymity under DDH. We present a novel secure *Forward-Anonymous Batch Mix* as a new application.

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 [5] 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 paper we present a formal treatment of a different fundamental class of anonymous communication protocols, namely, those based on universal re-encryption. A shortened version of this paper is in *Security and Cryptography for Networks—SCN 2018* [30]. Universal re-encryption forms the basis of what has been called "re-encryption mixes".

Golle et al presented the definition of a universal cryptosystem that permits re-encryption without knowledge of the public key. They called this definition `UCS` [11]. By extending the ElGamal public key cryptosystem [7], they instantiated a `UCS`, hereafter referred to as the `UCS` construction. They also defined what it means for a `UCS` to be secure. This they called universal semantic security under re-encryption, abbreviated `USS`. They used `UCS` as a basis to construct a re-encryption mix and an anonymized RFID tag application, hereafter referred to as GJJSMix and GJJSRFID, respectively.

A ciphertext of `UCS` has the property that it can be efficiently re-encrypted by anyone without knowledge of the receiver's public key. This re-encryption is accomplished without decrypting the ciphertext, without adding a new encryption layer, 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 anonymous communication protocols into the area of "end-to-end encryption" systems that do not rely on servers to maintain secret keys, thereby exhibiting the forward-secrecy property. Forward-secrecy and end-to-end encryption are becoming increasingly important in industrial systems in the post-Snowden era.

Whereas the `USS` definition has an anonymity test after the re-encryption operation, there is no anonymity test after the initial encryption operation. This means that `USS` does not require that the encryption function achieve key anonymity. This gap has had a knock-on effect on follow-on works, causing them to exhibit the same gap. We show that the key anonymity gap that is present in the definition of security of a universal cryptosystem (as defined by `USS`) is inherited by the security definitions of six applications that rely on universal re-encryption as a black-box. All six applications did not introduce new problems

per se, but inherently assume that the encryption function is key anonymous, thereby potentially exposing ciphertexts produced by the encryption function to the adversary.

However, the gap did not only affect applications in follow-on work to [11]. We show that the security definitions of the applications GJJSMix and GJJSRFID that appear in [11] exhibit this gap as well, allowing instantiations of the encryption function that compromise user anonymity. Since these two application security definitions are in [11], we show that `USS` does not sufficiently capture what is necessary for security.

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 of systems that require key anonymity. In particular, our new definition requires that the re-encryption function *and* the encryption function achieve key anonymity. Our definition requires that the re-encryption function and the encryption function achieve message indistinguishability. Our contributions are as follows:

1. We identify a gap in the definition of a universal re-encryption cryptosystem, namely, the missing requirement that the encryption function achieve key anonymity.
2. We cryptanalyze this gap and formally prove that it exists using a carefully constructed encryption function that achieves all that is required in the original work.[3]
3. Due to black-box use of the primitive, we point out that the gap applies to the following applications: GJJSMix, a mix network with defense against unwanted messages [16], GJJSRFID, Klein bottle routing protocol [21], the mobile private microblogging protocol [26], and an additional RFID protocol [25]. For all of these protocols: failure of the encryption function to achieve key anonymity (as not required by the original work) results in privacy loss/compromised receiver anonymity.
4. We then present what we call *semantically secure anonymity* that defines the complete set of security properties that assure key anonymity.
5. Construction: We generalize the well-known DDH random self-reduction and then use this generalization to prove that the `UCS` construction is secure under DDH in *our new model*.[4] The proof may point at how to correct the derived applications, while the new reduction technique may have independent applications.
6. Example application: We present a new forward-anonymous batch mix and prove that it is secure (as modeled here) under DDH.
7. A notion related to universal re-encryption is that of *incomparable public keys*. The proof of key anonymity of the ElGamal-based incomparable public

[3] See Theorem 1. The gap pertains to the "initial" encryption function, **not** the re-encryption function.

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

key cryptosystem [28] is in the random oracle model. We give the first proof of key anonymity for incomparable public keys in the standard model and it is a direct reduction with respect to DDH.

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.

Organization:

Related work is presented in Section 2. Notation and definitions are covered in Section 3. In Section 4 we review `UCS` and `USS` and show that there is a gap in `USS`. We use the gap in Section 5 to break the security definitions of six cryptographic applications. We define semantically secure anonymity in Section 6. We review the `UCS` construction in Section 7 with adjusted input/output specifications to accommodate our proofs of security. The new DDH reduction technique is given in Section 8 and we use it to prove the security of the `UCS` construction in Section 9. The forward-secure batch mix is presented and proven secure in Section 10. We conclude in Section 11. A security analysis of incomparable public keys is given in Appendix A.

2 Related Work

We first review the literature that leverages universal re-encryption as a primitive. `UCS` is a 4-tuple of algorithms: a key generator, an encryption algorithm, a re-encryption algorithm, and a decryption algorithm. A ciphertext produced using this cryptosystem can be re-encrypted by anyone without first decrypting it. In GJJSRFID, 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 is intractable to obtain, making it difficult to track the object. GJJSMix applies the `UCS` construction to produce 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` [8]. 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 [16]. Onion-based routing and universal re-encryption were leveraged to form hybrid anonymous communication protocols [12, 17]. A circuit-based anonymity protocol was presented based on universal re-encryption [18]: 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. Weaknesses in [16, 17, 12, 18] were presented in [6]. Golle presented a *reputable mix network* construction based on universal re-encryption [10]. 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.

Issues about key anonymity have been noticed or absent in other works on re-encryption: First, Groth presented a re-randomizable and replayable cryptosystem based on DDH achieving adaptive chosen ciphertext security [13]. The construction and security arguments do not address key anonymity. Secondly, Prabhakaran and Rosulek presented a construction for a rerandomizable encryption scheme [22] 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 [23]. They indicate that their construction does not achieve HCCA-anonymity and leave the anonymity problem as open.

There has been more recent work on proxy encryption [15]. 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 actual 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 (i.e., the same "respect") as, say, message security in public key cryptosystems. We believe that our work shows that identifying subtleties and producing necessary revisions of models is relevant, even for works that are older than 10 years, especially in areas that are becoming increasingly important to real-world applications and systems.

The important notion of key privacy (also called key anonymity) was introduced by Bellare, Boldyreva, Desai, and Pointcheval [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 anonymity. The folklore at the time was that Indeed ElGamal where all keys are drawn from the same group has a property which is similar to what was formally modeled and

[5] github.com/grnet/zeus

proved in that paper, yet the correct model and work as a whole is, nowadays, considered fundamental to privacy.

The present paper was published in 2016 on e-print [29]. It influenced the privacy-preserving user-auditable pseudonym system of Camenisch and Lehmann [4] who leverage our security definition for incomparable public keys and cite the applicability of our reduction technique from Section 8. The present paper was also mentioned as a needed building block for universal re-encryption for AppeCoin.[6]

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 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 $\text{negl}(n) < \frac{1}{p(n)}$. 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, 27, 20]. `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,

[6] blog.coinfabrik.com/review-appecoin-alternative-anonymous-cryptocurrency

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

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 UCS and USS

UCS is a 4-tuple of algorithms (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 USS:

Experiment $\mathbf{Exp}^{uss}_{\mathcal{A}}(UCS, k)$
$PK_0 \leftarrow$ 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 of UCS is said to be *semantically secure under re-encryption* (i.e., achieve USS) if for any adversary $\mathcal{A}$ with resources polynomial in K, the probability given by $\mathrm{pr}[\mathbf{Exp}^{uss}_{\mathcal{A}}(UCS, k) = \text{'1'}] - 1/2$ is negligible in k.

The UCS construction 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. Let g be a generator for $G_{\mathfrak{p}}$. The key generator outputs $(PK, SK) = (y, x)$ where $x \in_U \mathbb{Z}_q$ and $y = g^x \bmod p$.

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 $c \leftarrow ((\alpha_0, \beta_0), (\alpha_1, \beta_1)) \leftarrow ((m y^{k_0} \bmod p, g^{k_0} \bmod p), (y^{k_1} \bmod p, g^{k_1} \bmod p))$.

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 $((\alpha'_0, \beta'_0), (\alpha'_1, \beta'_1))$ which is equal to $((\alpha_0 \alpha_1^{k'_0}, \beta_0 \beta_1^{k'_0}), (\alpha_1^{k'_1}, \beta_1^{k'_1}))$.

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 by Golle et al accepts as valid cryptosystems that, in fact, contain encryption algorithms UE that do not produce key anonymous ciphertexts. Consider the following modification of UE called UE':

1. let b_1 be the least significant bit of y
2. generate random encryption nonces $(k_0, k_1) \in_U \mathbb{Z}_q \times \mathbb{Z}_q$
3. set $(\alpha_0, \beta_0) \leftarrow (my^{k_0} \bmod p, g^{k_0} \bmod p)$
4. set $(\alpha_1, \beta_1) \leftarrow (y^{k_1} \bmod p, g^{k_1} \bmod p)$
5. set $c \leftarrow ((\alpha_0, \beta_0), (\alpha_1, \beta_1))$
6. let b_2 be the least significant bit of β_0
7. if $b_1 \neq b_2$ then goto step 2 otherwise output c

Cryptosystem A: Cryptosystem A is the same as the `UCS` construction except that UE is replaced with UE'.

`DDHRerand5` in the proof of Theorem 1 is covered in Section 8.

Theorem 1. *If DDH is hard then Cryptosystem A is secure in the sense of* `USS`.

Proof. Suppose for the sake of contradiction that there exists a successful probabilistic polynomial time USS distinguishing adversary $\mathcal{A}$ for Cryptosystem A. Adversary $\mathcal{A}$ is **stateful**. Consider algorithm `AlgRA` that takes as input a Decision Diffie-Hellman problem instance $((p, q), g, a_0, b_0, c_0)$.

`AlgRA`$((p, q), g, a_0, b_0, c_0)$:
1. $(\theta'_j, \theta_j, y_j, \mu_j, \mu'_j) \leftarrow$ `DDHRerand5`$((p, q), g, a_0, b_0, c_0)$ for $j = 0, 1$
2. $PK_0 \leftarrow y_0, PK_1 \leftarrow y_1$
3. $(m_0, m_1, r_0, r_1) \leftarrow \mathcal{A}(PK_0, PK_1, \text{“specify ciphertexts”})$;
4. if $m_0, m_1 \notin \mathbf{M}$ or $r_0, r_1 \notin \mathbf{R}$ then output ‘0’;
5. $C_0 \leftarrow ((\alpha_{0,0}, \beta_{0,0}), (\alpha_{0,1}, \beta_{0,1})) \leftarrow$ `UE'`(m_0, r_0, PK_0)
6. $C_1 \leftarrow ((\alpha_{1,0}, \beta_{1,0}), (\alpha_{1,1}, \beta_{1,1})) \leftarrow$ `UE'`(m_1, r_1, PK_1)
7. $C'_0 \leftarrow ((\alpha_{0,0}\mu_0, \beta_{0,0}\theta_0), (\mu'_0, \theta'_0))$
8. $C'_1 \leftarrow ((\alpha_{1,0}\mu_1, \beta_{1,0}\theta_1), (\mu'_1, \theta'_1))$
9. $b \in_U \{0, 1\}$
10. $b' \leftarrow \mathcal{A}(C'_b, C'_{1-b}, \text{“guess”})$
11. if $b = b'$ then output ‘1’ else output ‘0’;

Consider the case that the input is a DH 3-tuple. Clearly C_j is the ciphertext under public key PK_j as specified by $\mathcal{A}$ for $j = 0, 1$. It follows from the definition of `DDHRerand5` that C'_j is a re-encryption of C_j in accordance with `URe` for $j = 0, 1$. Therefore, the input to $\mathcal{A}$ is drawn from the same set and probability distribution as the input to $\mathcal{A}$ in USS. Since $\mathcal{A}$ distinguishes with non-negligible advantage, it follows that $b = b'$ with probability greater than or equal to $\frac{1}{2} + \gamma$ where γ is non-negligible in the security parameter.

Now consider the case that the input is not a DH 3-tuple. It follows from definition of `DDHRerand5` that the 5-tuple $(\theta'_j, \theta_j, y_j, \mu_j, \mu'_j)$ is uniformly distributed in $G_{\mathfrak{p}}^5$ for $j = 0, 1$. Therefore, C'_j is uniformly distributed in $G_{\mathfrak{p}}^2 \times G_{\mathfrak{p}}^2$ for $j = 0, 1$. Let p_1 be the probability that $\mathcal{A}$ responds with $b' = 0$. Then the probability that $b = b'$ is $\frac{1}{2}p_1 + \frac{1}{2}(1 - p_1) = \frac{1}{2}$. It follows that $\mathcal{A}$ has negligible advantage to distinguish in this case. □

We have therefore proven that Cryptosystem A is "secure" under `USS`. Let y_0 and y_1 be two public keys. Suppose that y_0 and y_1 have differing least significant bits. An adversary can break the anonymity of UE' by extracting the least significant bit of β_0 and correlating it with the public key with matching least significant bit. This proves that Cryptosystem A satisfies `USS` yet has an encryption algorithm that does not achieve key anonymity. This, in turn, proves that `USS` admits cryptosystems wherein URe is key anonymous but UE is not key anonymous.

`USS` is devoid of a requirement that the output of UE be key anonymous. It has a test of anonymity of URe *but there is no test of anonymity of UE.* This is the only definition of security spelled out for `UCS` in [11]. Therefore, the foundation put forth by Golle et al for universal re-encryption "accepts" as secure encryption algorithms UE that are not key anonymous as proven by Cryptosystem A. There may exist other constructions in which the failure of UE to achieve key anonymity is subtle, yet like Cryptosystem A 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.

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.

5 Systemic problem caused by the `USS` definition

An insufficient definition in security modeling may migrate to other constructions. Thus the risk of potentially getting an insecure system due to varying the underlying cryptographic tools is magnified. We have identified six cryptographic applications that leverage `UCS` and `USS` that, merely due to copying the component of the original work, have gaps in their security definitions. These

applications advocate the use of UCS to instantiate the applications. UCS is secure and does provide key anonymity of the encryption function and re-encryption function. However, the point we are making is that these six applications inherit USS as part of their security definitions, they rely on UE achieving key anonymity, but there is nothing in the security definitions that require UE to achieve key anonymity. From the perspective of having sound security definitions, the *definitions* of security for these six applications is broken. We emphasize the difference between a definition of security of a cryptosystem vs. an instantiation that must be proven to adhere to the definition. Our goal here is to provide a remedy to these applications that employ UE in a blackbox fashion. We elaborate on these important applications in details in order to demonstrate cases where anonymity of users is very crucial and must be modeled correctly.

The definition of security of GJJSMix is missing a crucial key anonymity requirement. Let $\mathcal{U}$ be a universal cryptosystem that has an encryption algorithm UE that outputs ciphertexts that are not key anonymous. GJJSMix 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. This places users of this mix (e.g., activists, journalists) in harms way.

We now analyze the mix network protocol that leverages signatures to protect against unwelcome messages such as spam [16]. Their solution leverages UCS and relies on USS. In the Admission Protocol, each message that enters the system is encrypted using the public key of the recipient. The resulting ciphertexts are received by a server from a pool of servers. This exposes ciphertexts produced by UE to the first server. This Admission Protocol application therefore assumes that UE provides key anonymity even though this security requirement is not captured anywhere.

In GJJSRFID, the data contained in RFID tags is encrypted using a UCS. An example is given that leverages a key pair owned by a transit agency and a key pair owned by a department store. The description of this application permits the initial RFID ciphertext to be the output of UE. When UE is not key anonymous it follows that the RFID tag can be correlated with the associated public key. This compromises privacy.

A protocol for RFID privacy that leverages a UCS and that relies on USS is given in [25]. Their protocol has the RFID tag regularly emit ID information in the form of a universal ciphertext C produced using UE. When UE does not achieve key anonymity, this means that receiver anonymity is compromised. This constitutes a *perpetual* window of attack.

The use of a UCS to achieve compliance with RFID privacy legislation, addressing the EU RFID Privacy and Data Protection working document in particular, is proposed in [24]. The proposal presumes that USS encapsulates the privacy assurances that are needed. We cannot overstate the importance of a proper definition of security for RFID applications. The correct frame of mind is not RFIDs in products in grocery stores. The correct frame of mind is RFID chips in people, e.g., VeriChip [14] that cites the use of UCS.

The Klein bottle routing protocol [21] leverages a UCS with a slight change. They add to a UCS n-out-of-n decryption. We will not reiterate the Klein bottle routing protocol but will provide enough points to show that the security definition of it is broken since it directly relies on USS. The protocol leverages a set of routers. Let the routers be labeled Alice, Bob, and Carol, each of whom has a key pair for the routing. There is a sender Sally and receiver Rick. Sally announces that she will create and send out Klein bottles. She states that she will only ever use two possible routes: Sally $\rightarrow$ Alice $\rightarrow$ Bob $\rightarrow$ Rick, or Sally $\rightarrow$ Alice $\rightarrow$ Carol $\rightarrow$ Rick. She further announces that she uses the following algorithm to decide which route to use for a given bottle: flip a fair coin. If the result is heads, use the route that goes through Bob. If the result is tails, use the route that goes through Carol. Let y_0 denote the public key that is the product of Alice's public key and Bob's public key (mod p). Let y_1 denote the public key that is the product of Alice's public key and Carol's public key (mod p). The second value in the encrypted route list is encrypted under either y_0 or y_1 using UE. Consider a distinguishing adversary that obtains the bottle right after it leaves Sally. When UE does not achieve key anonymity then the adversary has a non-negligible advantage in determining whether y_0 or y_1 was used to compute this second value. It follows that the adversary knows with non-negligible advantage whether the bottle will go to Bob or Carol before it even arrives at Alice.

A mobile private microblogging protocol MoP-2-MoP [26] that leverages a UCS and that relies on USS is another important application. The implementation as given exposes the ciphertexts produced by UE.[7] UE needs to achieve key anonymity for security to hold. When UE does not achieve key anonymity, receiver anonymity is compromised.

We have shown that these six applications all assume that UE achieves key anonymity yet nowhere is this cryptographic requirement asserted. We pointed out the above to demonstrate the harmful knock-on effect that the insufficient USS security definition has had. These multiple important examples show that this gap was and continues to be a systemic problem (risk) in the design of new application protocols since UE is an important building block. We have therefore further shown that the requirement for UE to achieve key anonymity is indeed **necessary**.

In hindsight, we believe that UCS and USS provide great insight into laying a proper foundation for universal re-encryption. In particular, we commend the approach of having the adversary fully specify the ciphertexts (messages and nonces) that are used in forming the re-encryption challenge ciphertexts. However, the USS definition is certainly not sufficient!

6 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

[7] Per Section 2.1 of [26].

adjustments to the input/output specifications of UCS. For example, the original UCS key generator does 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 [9, 19].

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. *Stateful 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. *Stateful 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. *Stateful 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. *Stateful 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.

7 Universal Re-Encryption Cryptosystem

We adjusted the input/output specifications of UCS (see Section 6) to facilitate our proofs. But, we preserved the original cryptosystem entirely. For clarity, we now present the cryptosystem in full.

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 $(y, x) \leftarrow \mathtt{UKG}(n, \lambda)$. 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 $\mathtt{UE}_{pk}(m, (k_0, k_1), \lambda)$. 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 $\mathtt{UD}_{sk}(c, \lambda)$. 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 $\mathtt{URe}(((\alpha_0, \beta_0), (\alpha_1, \beta_1)), (\ell_0, \ell_1), \lambda)$. 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 $c' \leftarrow ((\alpha_0', \beta_0'), (\alpha_1', \beta_1'))$.

8 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 9 (granted, this new reduction is given for pragmatic and proof simplicity reasons, and not as an essential issue as are the modeling issues and their correction presented above). We define algorithm `DDHRerand5` as follows. `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,

$$(x'', x', y', z', z'') \leftarrow (x^{v'} g^{u_1'}, x^v g^{u_1}, y g^{u_2}, z^v y^{u_1} x^{vu_2} g^{u_1 u_2}, z^{v'} y^{u_1'} x^{v'u_2} g^{u_1' u_2})$$

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 $G_{\mathfrak{p}}$. 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.

9 Security of Universal Cryptosystem Ψ

We now give the theorems for the proofs of security for our construction. These are the first proofs of security for universal re-encryption that constitute direct reductions with respect to DDH and prove all the properties that are necessary (in the sense of the fact that any missing property implies potential breaks).

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

Proof. Suppose there exists a probabilistic polynomial time adversary $\mathcal{A}$ for $AnonEnc^{eav}_{\mathcal{A},\Psi}$, 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 $((p,q),g,a_0,b_0,c_0)$.

`AlgR3`$((p,q),g,a_0,b_0,c_0)$:
1. set $(\theta_j', \theta_j, y_j, \mu_j, \mu_j') \leftarrow$ `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 in Section 8 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 $G^5_{\mathfrak{p}}$. Therefore, c is uniformly distributed in $G^2_{\mathfrak{p}} \times G^2_{\mathfrak{p}}$. 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 3. *If DDH is hard then* $\Pr[AnonReEnc^{eav}_{\mathcal{A},\Psi}(n, \lambda) = 1] \leq \frac{1}{2} + \mathrm{negl}(n)$.

Proof. Suppose there exists a probabilistic polynomial time adversary $\mathcal{A}$ for $AnonReEnc^{eav}_{\mathcal{A},\Psi}$, 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 in Section 8 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 $G^5_{\mathfrak{p}}$. Therefore, c' is uniformly distributed in $G^2_{\mathfrak{p}} \times G^2_{\mathfrak{p}}$. 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 $\Pr[\mathtt{AlgR4}((p, q), g, g^a, g^b, g^c) = \text{“true”}] = \frac{1}{2} + \frac{2\psi - 1}{2q}$. □

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

Proof. Suppose there exists a probabilistic polynomial time adversary $\mathcal{A}$ for $PubKEnc^{eav}_{\mathcal{A},\Psi}$, 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 $((p,q),g,a_0,b_0,c_0)$.

`AlgR1`$((p,q),g,a_0,b_0,c_0)$:
1. set $(\theta',\theta,y,\mu,\mu') \leftarrow$ `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` in Section 8 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 $G^5_{\mathfrak{p}}$. Therefore, c is uniformly distributed in $G^2_{\mathfrak{p}} \times G^2_{\mathfrak{p}}$. 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 $\Pr[\texttt{AlgR1}((p,q),g,g^a,g^b,g^c) = \text{"true"}] = \frac{1}{2}+\frac{2\psi-1}{2q}$. □

Theorem 5. *If DDH is hard then* $\Pr[PubKReEnc^{eav}_{\mathcal{A},\Psi}(n,\lambda)=1] \leq \frac{1}{2}+\mathrm{negl}(n)$.

Proof. Suppose there exists a probabilistic polynomial time adversary $\mathcal{A}$ for $PubKReEnc^{eav}_{\mathcal{A},\Psi}$, 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 $((p,q),g,a_0,b_0,c_0)$.

`AlgR2`$((p,q),g,a_0,b_0,c_0)$:
1. set $(\theta',\theta,y,\mu,\mu') \leftarrow$ `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` in Section 8 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 $G_{\mathfrak{p}}^5$. Therefore, c' is uniformly distributed in $G_{\mathfrak{p}}^2 \times G_{\mathfrak{p}}^2$. 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 $\Pr[\texttt{AlgR2}((p, q), g, g^a, g^b, g^c) = \text{“true”}] = \frac{1}{2} + \frac{2\psi - 1}{2q}$. □

Theorems 2, 3, 4, and 5 show that Theorem 6 holds.

Theorem 6. *If DDH is hard then Ψ is secure in the sense of semantically secure anonymity.*

10 Batch Mixing

GJJSMix uses a universal cryptosystem to form a forward anonymous mix 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 GJJSMix 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. In particular, we define a re-encryption batch mix protocol `FBMIX` 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.

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.

10.1 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.*

10.2 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. $c_i \leftarrow \mathrm{UE}_{y_{\sigma_i}}(m_i, (k_{i,0}, k_{i,1}), ((p,q),g))$ 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 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 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. $c'_i \leftarrow$ URe$(c_i, (\ell_{i,0}, \ell_{i,1}), ((p,q),g))$ for $i = 1, 2, ..., w$
2. output 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$(c_1, c_2, ..., c_w, x_1, x_2, ..., x_w, ((p,q),g))$:
1. let L be the empty list
2. `for` i in 1 to w:
3. `for` j in 1 to w:
4. $(m, s) \leftarrow \mathrm{UD}_{x_j}(c_i, ((p,q),g))$
5. if $s =$ true
6. append (m, j) to L
7. `break`
8. 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 $c_i \leftarrow \mathrm{UE}_{y_{\sigma_i}}(m_i, (k_{i,0}, k_{i,1}), ((p,q),g))$. 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,

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

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.

10.3 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 8. 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 $\texttt{GetMiddle}(T)$ 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. $((y_1, x_1), (y_2, x_2), ..., (y_w, x_w)) \leftarrow \texttt{FBGEN}(n, ((p, q), g))$
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* $m_j \notin G_{\mathfrak{p}}$ *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. $(c_1, c_2, ..., c_w) \leftarrow \texttt{FBENCR}((m_1, (k_{1,0}, k_{1,1}), \sigma_1), ...,$
   $(m_w, (k_{w,0}, k_{w,1}), \sigma_w), y_1, y_2, ..., y_w, ((p, q), g))$
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. $T_j \leftarrow \texttt{DDHRerandN}((p, q), g, a_0, b_0, c_0, 2w)$ for $j = 1, 2, ..., w$
2. set $y_j = \texttt{GetMiddle}(T_j)$ for $j = 1, 2, ..., w$
3. $(m_1, m_2, ..., m_w) \leftarrow \mathcal{A}(((p, q), g), y_1, y_2, ..., y_w,$ "specify messages"$)$
4. if $\exists\ j \in \{1, 2, ..., w\}$ such that $m_j \notin G_{\mathfrak{p}}$ 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 $[1, q]$, $\Pr[\texttt{AlgR9}((p, q), g, g^a, g^b, g^{ab}) = \text{“true”}] \geq \frac{1}{w} + \frac{1}{n^\alpha}$. Define $\psi = \Pr[\texttt{AlgR9}((p, q), g, g^a, g^b, g^{ab}) = \text{“true”}]$.

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_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 $[1, q]$, $\Pr[\texttt{AlgR9}((p, q), g, g^a, g^b, g^c) = \text{“true”}] = \frac{q^2}{q^3}\psi + (1 - \frac{q^2}{q^3})\frac{1}{w}$ which is overwhelmingly close to $\frac{1}{w}$. □

We now address 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. $((y_1, x_1), (y_2, x_2), ..., (y_w, x_w)) \leftarrow \texttt{FBGEN}(n, ((p, q), g))$
3. $((m_1, r_1, \sigma_1), (m_2, r_2, \sigma_2), ..., (m_w, r_w, \sigma_w)) \leftarrow \mathcal{A}((p, q), g),$
   $y_1, y_2, ..., y_w, \text{“specify ciphertexts and receivers”})$
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. $(c_1, c_2, ..., c_w) \leftarrow \texttt{FBENCR}((m_1, r_1, \sigma_1), ..., (m_w, r_w, \sigma_w), y_1, y_2, ..., y_w, ((p, q), g))$
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. $(c'_{\pi(1)}, c'_{\pi(2)}, ..., c'_{\pi(w)}) \leftarrow \texttt{FBMIXER}(\pi, (c_1, \mu_1), (c_2, \mu_2), ..., (c_w, \mu_w), ((p, q), g))$
11. $\pi' \leftarrow \mathcal{A}(c'_{\pi(1)}, c'_{\pi(2)}, ..., c'_{\pi(w)}, \text{“guess”})$
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 $((p, q), g, a_0, b_0, c_0)$.

$\texttt{AlgR10}((p,q),g,a_0,b_0,c_0)$:
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. $((m_1,r_1,\sigma_1),(m_2,r_2,\sigma_2),...,(m_w,r_w,\sigma_w)) \leftarrow \mathcal{A}(((p,q),g),y_1,y_2,...,y_w,$ "specify ciphertexts and receivers")
4. if $\exists\ j \in \{1,2,...,w\}$ such that $m_j \notin G_{\mathfrak{p}}$ 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. $(c_1,c_2,...,c_w) \leftarrow \texttt{FBENCR}((m_1,r_1,\sigma_1),...,(m_w,r_w,\sigma_w),y_1,y_2,...,y_w,((p,q),g))$
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. $(c'_{\pi(1)},c'_{\pi(2)},...,c'_{\pi(w)}) \leftarrow \texttt{fp}(\pi,c'_1,c'_2,...,c'_w)$
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 $G_{\mathfrak{p}}$ for $j = 1,2,...,w$ and that c'_j is uniformly distributed in $G_{\mathfrak{p}}^2 \times G_{\mathfrak{p}}^2$ 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* $(m_{j,0} \notin G_{\mathfrak{p}}$ *or* $m_{j,1} \notin G_{\mathfrak{p}})$ *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. $(c_1, c_2, ..., c_w) \leftarrow$ `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 $((p, q), g, a_0, b_0, c_0)$.

`AlgR7`$((p, q), g, a_0, b_0, c_0)$:
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. $((m_{1,0}, m_{1,1}, \sigma_1), (m_{2,0}, m_{2,1}, \sigma_2), ..., (m_{w,0}, m_{w,1}, \sigma_w))$
$\leftarrow \mathcal{A}(((p, q), g), y_1, y_2, ..., y_w,$ "specify messages and receivers")
4. if $\exists\ j \in \{1, 2, ..., w\}$ such that $(m_{j,0} \notin G_{\mathfrak{p}}$ or
$m_{j,1} \notin G_{\mathfrak{p}})$ 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 $[1, q]$, $\Pr[$`AlgR7`$((p, q), g, g^a, g^b, g^{ab}) =$ "true"$] \geq \frac{1}{2} + \frac{1}{n^\alpha}$. Define $\psi = \Pr[$`AlgR7`$((p, q), g, g^a, g^b, g^{ab}) =$ "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 $G_{\mathfrak{p}}^2 \times G_{\mathfrak{p}}^2$ and

y_j is uniformly distributed in $G_{\mathfrak{p}}$ 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 $\Pr[\texttt{AlgR7}((p, q), g, g^a, g^b, g^c) = \text{“true”}] = \frac{q^2}{q^3}\psi + (1 - \frac{q^2}{q^3})\frac{1}{2}$ 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. $((y_1, x_1), (y_2, x_2), ..., (y_w, x_w)) \leftarrow \texttt{FBGEN}(n, ((p, q), g))$
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 ($m_{j,0} \notin G_{\mathfrak{p}}$ or $m_{j,1} \notin G_{\mathfrak{p}}$)*
   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. *$r_j \in_U [1, q] \times [1, q]$ 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 ($m_{j,0} \notin G_{\mathfrak{p}}$ or
   $m_{j,1} \notin G_{\mathfrak{p}}$) 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. $(c_1, c_2, ..., c_w) \leftarrow$ 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 $[1, q]$, $\Pr[\text{AlgR8}((p,q), g, g^a, g^b, g^{ab}) = \text{"true"}] \geq \frac{1}{2} + \frac{1}{n^\alpha}$. Define the value $\psi = \Pr[\text{AlgR8}((p,q), g, g^a, g^b, g^{ab}) = \text{"true"}]$.

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 $G_{\mathfrak{p}}$ for $j = 1, 2, ..., w$ and that c'_j is uniformly distributed in $G_{\mathfrak{p}}^2 \times G_{\mathfrak{p}}^2$ 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 $\Pr[\text{AlgR8}((p,q), g, g^a, g^b, g^c) = \text{"true"}] = \frac{q^2}{q^3}\psi + (1 - \frac{q^2}{q^3})\frac{1}{2}$ which is overwhelmingly close to $\frac{1}{2}$. □

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 11 therefore holds.

Theorem 11. *If DDH is hard then* FBMIX *is a forward-anonymous batch mix.*

11 Conclusion

We showed that the definition of security of universal re-encryption, USS, is missing the requirement that the encryption algorithm produce key anonymous ciphertexts, thereby forming a gap. We leveraged this gap to show that the security definitions of multiple applications of universal re-encryption contain the gap as well, breaking anonymity. Two of these applications are in the original paper on universal re-encryption by Golle et al, showing that the original security definition of re-encryption, namely, USS, is in err. 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 proved that the original ElGamal-based universal cryptosystem is secure under our new definition of security. We presented a forward-anonymous batch mix that is secure under DDH. Finally, we applied our new DDH reduction technique to give the first proof in the standard model that ElGamal-based incomparable public keys achieve key anonymity 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. Lehmann. Privacy-Preserving User-Auditable Pseudonym Systems. In *IEEE European Symposium on Security and Privacy*, 2017.
5. J. Camenisch and A. Lysyanskaya. A Formal Treatment of Onion Routing. In *Advances in Cryptology—Crypto '05*, pages 169–187, 2005.
6. G. Danezis. Breaking Four Mix-related Schemes Based on Universal Re-encryption. *Int. J. Inf. Sec.*, 6(6):393–402, 2007.
7. T. ElGamal. A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. In *Advances in Cryptology—Crypto '84*, pages 10–18, 1985.
8. P. Fairbrother. An Improved Construction for Universal Re-encryption. In *Privacy Enhancing Technologies*, pages 79–87, 2004.
9. S. Goldwasser and S. Micali. Probabilistic Encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.
10. P. Golle. Reputable Mix Networks. In *Privacy Enhancing Technologies*, pages 51–62, 2004.
11. P. Golle, M. Jakobsson, A. Juels, and P. Syverson. Universal Re-encryption for Mixnets. In *CT-RSA 2004*, pages 163–178, 2004.
12. 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.
13. J. Groth. Rerandomizable and Replayable Adaptive Chosen Ciphertext Attack Secure Cryptosystems. In *Theory of Cryptography—TCC '04*, pages 152–170, 2004.
14. J. Halamka, MD, A. Juels, A. Stubblefield, MD, and J. Westhues, MD. The Security Implications of Verichip Cloning. *J. Am. Med. Inform. Assoc.*, 13(6):384–396, 2006.

15. S. Hohenberger, G. N. Rothblum, A. Shelat, and V. Vaikuntanathan. Securely Obfuscating Re-encryption. *Journal of Cryptology*, 24(4):694–719, 2011.
16. 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.
17. M. Klonowski, M. Kutyłowski, and F. Zagórski. Anonymous Communication with On-line and Off-line Onion Encoding. In *SOFSEM*, pages 229–238, 2005.
18. 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.
19. S. Micali, C. Rackoff, and B. Sloan. The Notion of Security for Probabilistic Cryptosystems. *SIAM Journal on Computing*, 17(2):412–426, 1988.
20. M. Naor and O. Reingold. Number-Theoretic Constructions of Efficient Pseudo-Random Functions. In *IEEE FOCS '97*, pages 458–467, 1997.
21. K. Peng, J. M. Nieto, Y. Desmedt, and E. Dawson. Klein bottle routing: An alternative to onion routing and mix network. In *ICISC '06*, pages 296–309, 2006.
22. M. Prabhakaran and M. Rosulek. Rerandomizable RCCA Encryption. In *Advances in Cryptology—Crypto '07*, pages 517–534, 2007.
23. M. Prabhakaran and M. Rosulek. Homomorphic Encryption with CCA Security. In *Automata, Languages and Programming*, pages 667–678, 2008.
24. M. R. Rieback, B. Crispo, and A. S. Tanenbaum. Uniting Legislation with RFID Privacy-Enhancing Technologies. In *Proceedings of the 3rd Conference on Security and Protection of Information—SPI '05*, pages 15–23, 2005.
25. J. Saito, J.-C. Ryou, and K. Sakurai. Enhancing privacy of Universal Re-encryption scheme for RFID tags. In *Embedded and Ubiquitous Computing (EUC)*, pages 879–890, 2004.
26. M. Senftleben, M. Bucicoiu, E. Tews, F. Armknecht, S. Katzenbeisser, and A.-R. Sadeghi. MoP-2-MoP - Mobile private microblogging. In *Financial Cryptography and Data Security—FC '14*, pages 384–396, 2014.
27. M. Stadler. Publicly Verifiable Secret Sharing. In *Advances in Cryptology—Eurocrypt '96*, pages 190–199, 1996.
28. B. R. Waters, E. W. Felten, and A. Sahai. Receiver Anonymity via Incomparable Public Keys. In *ACM CCS*, pages 112–121, 2003.
29. A. L. Young and M. Yung. Semantically Secure Anonymity: Foundations of Re-encryption. Cryptology ePrint Archive, Report 2016/341. March 29, 2016. http://eprint.iacr.org/2016/341.
30. A. L. Young and M. Yung. Semantically Secure Anonymity: Foundations of Re-Encryption. In *Security and Cryptography for Networks—SCN '18*, 2018.

A Security of Incomparable Public Keys

We now review incomparable public keys from [28] using their definitions. G is a common key generator that given a security parameter k will produce the common key I. I serves as a global parameter for key generation. In the construction the receivers share the global parameters p and q. p and q are both primes and $q = (p-1)/2$. K is the private key generation algorithm that on input I outputs a random private key. L is an algorithm that on input the private key will output a random corresponding incomparable public key. In the construction the private key is the randomly generated exponent a and the corresponding public key is (g, g^a) where g is a randomly chosen quadratic residue in $\mathbb{Z}_p^*$.

The key privacy property of the ElGamal-based incomparable public key cryptosystem relies on the random oracle model. The random oracle model proof is given in Appendix B of [28]. We improve upon this by proving that key anonymity of ElGamal-based incomparable public keys holds in the standard model under DDH.

We remark that Waters et al give a proof of security for a two-key implementation in Appendix C. Whereas the proof does not use the random oracle model, the underlying incomparable public key cryptosystem relies on a symmetric cipher that if broken compromises key anonymity. The symmetric cipher is not specified (i.e., it is generic). So, the anonymity of the construction in Appendix C relies not only on DDH but the security of an unspecified symmetric cipher.

A.1 Defining the Security of Incomparable Public Keys

The definition of incomparability is tied to the public key cryptosystem that uses the incomparable public key. Although it is a minor point, this dependency is not necessary. In practice the adversary may only have an incomparable public key of Alice, and incomparable public key of Bob, and a challenge incomparable public key. There are no messages and no ciphertexts here. Also, in a given protocol one may want to produce a new incomparable public key given not the corresponding private key but a given instance of the incomparable public key. We use these small observations to stream-line the definition of incomparability.

The key generator is $\texttt{IGEN}((p,q),g)$ that outputs the tuple $((g^k, y^k), x)$ where $y = g^x \bmod p$ and $k, x \in_U [1,q]$. Let (a,b) denote the incomparable public key (g^k, y^k). The corresponding private key is x. The public key re-randomization algorithm for incomparable public keys is $\texttt{IRR}((p,q),(a,b))$ and it outputs $(a^r \bmod p, b^r \bmod p)$ where $r \in_U [1,q]$.

Definition 12. *If $\forall$ probabilistic polytime adversaries $\mathcal{A}$, $\forall\ \alpha > 0$, and $\forall$ sufficiently large κ, after the following,*

1. *generate* $((p,q),g) \leftarrow \mathcal{IG}(\kappa)$
2. $((a_i,b_i),x_i) \leftarrow \texttt{IGEN}((p,q),g)$ *for* $i = 0,1$
3. $t \in_U \{0,1\}$
4. $(a_2,b_2) \leftarrow \texttt{IRR}((p,q),(a_t,b_t))$
5. $t' \leftarrow \mathcal{A}((p,q),g,(a_0,b_0),(a_1,b_1),(a_2,b_2))$
6. *if* $t = t'$ *then output "true" else output "false"*

the output of the experiment is "true" with probability less than $\frac{1}{2} + \frac{1}{\kappa^\alpha}$ *then* $(\texttt{IGEN}, \texttt{IRR})$ *is secure in the sense of incomparability.*

A.2 Proving the Security of Incomparable Public Keys

We now give the first proof in the standard model that incomparability holds under DDH.[8]

[8] This was covered in the March 2016 and May 2016 versions of this ePrint.

Theorem 12. *If DDH is hard then* $(\texttt{IGEN}, \texttt{IRR})$ *is secure in the sense of incomparability.*

The below is the proof of Theorem 12.

Proof. Suppose there exists a probabilistic polynomial time adversary $\mathcal{A}$, 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 $\texttt{AlgB}$ that takes as input a DDH problem instance $((p,q), g, a_0, b_0, c_0)$.

$\texttt{AlgB}((p,q), g, a_0, b_0, c_0)$:
1. $(A'_0, A_0, y_0, R_0, R'_0) \leftarrow \texttt{DDHRerand5}((p,q), g, a_0, b_0, c_0)$
2. $(A'_1, A_1, y_1, R_1, R'_1) \leftarrow \texttt{DDHRerand5}((p,q), g, a_0, b_0, c_0)$
3. $t \in_U \{0,1\}$
4. $t' \leftarrow \mathcal{A}((p,q), g, (A_0, R_0), (A_1, R_1), (A'_t, R'_t))$
5. if $t = t'$ then output "true" else output "false"

Consider the case that the input is a DH 3-tuple. It follows from the definition of $\texttt{DDHRerand5}$ that (A_i, R_i) is a proper incomparable public key with corresponding ElGamal public key y_i under $\texttt{IGEN}$ for $i = 0, 1$.

It also follows that (A'_t, R'_t) is a proper re-randomization of (A_t, R_t) under $\texttt{IRR}$. Therefore, the input to $\mathcal{A}$ is drawn from the same set and probability distribution as the input to $\mathcal{A}$ in Definition 12. It follows that $t = t'$ with probability greater than or equal to $\frac{1}{2} + \frac{1}{\kappa^\alpha}$. So, for random exponents a and b in $[1, q]$, $\Pr[\texttt{AlgB}((p,q), g, g^a, g^b, g^{ab}) = \text{"true"}] \geq \frac{1}{2} + \frac{1}{\kappa^\alpha}$. Define $\psi = \Pr[\texttt{AlgB}((p,q), g, g^a, g^b, g^{ab}) = \text{"true"}]$.

Now consider the case that the input is not a DH 3-tuple. It follows from the definition of $\texttt{DDHRerand5}$ that $(A_0, R_0, A_1, R_1, A'_t, R'_t)$ is uniformly distributed in $G_{\mathfrak{p}}^6$. Let p_1 be the probability that $\mathcal{A}$ responds with $t' = 0$. Then the probability that $t = t'$ 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 $\Pr[\texttt{AlgB}((p,q), g, g^a, g^b, g^c) = \text{"true"}] = \frac{1}{2} + \frac{2\psi - 1}{2q}$. $\square$