Verifying a shuffle in zero-knowledge

Chris Patton

February 11, 2016

1 Introduction

A mixnet (sometimes called a "mix-network") is a service designed to facilitate anonymous communications over the internet [1]. Each of a set of users $\mathcal{Q} = \{P_1, \ldots, P_n\}$ wish to send a message to a user in $\mathcal{Q}$. Each encrypts their message M and the identity of the intended recipient P_i under the public key of the mixnet server (heereafter referred to as the "mix") and transmits the ciphertext. The mix server waits until it receives $m \leq n$ ciphertexts from m distinct users. It then decrypts each message and destination pair under its secret key and transmits each message to its intended destination. Crucially, the mix transmits the messages in a random order so as to hide the correspondence between senders and recipients. As long as the mix is trusted and the user's ciphertexts are faithfully transmitted to the mix, this simple approach suffices to preserve anonymity of its participants.

Normally we think of the mixnet being composed of a cascade of mix servers, each decrypting, shuffling, then transmitting to the next mix in the cascade until finally the last server outputs the users' messages. This is meant to distribute trust so that as long as the messages pass through at least one honest mix, anonymity is achieved. Still, the trust model here is rather strong. For example, the mix could send a fake message on behalf of one of the senders, or redirect it to a different recipient. We consider the problem of publically verifying that the outputs are indeed a permutation of the inputs. Moreover, we require that the correspondence not be revealed. Andrew Neff describes in [6] a method for proving a shuffle in zero-knowledge.

Neff's solution is based on the ElGamal public key encryption scheme, which is known to be semantically secure assuming the discrete log problem (decisional Diffie-Hellman, or DDH) is hard [2]. Based on this assumption, Neff proves his protocol achieves a property known as honest verifier zero-knowledge (HVZK) [5]. Informally speaking, this means that as long as the verifier faithfully executes the protocol, the mix can prove the outputs are a permutation of the inputs without revealing the permutation.

I formulate and define security for the problem and present Neff's protocol. For simplicity, I limit the discussion to a mixnet with a single mix server. In this scenario, users encrypt their messages under the public key of the mix. The mix waits until it receives a full batch of ciphertexts, decrypts them under its private key, and shuffles the output. In section 2, I specify ElGamal encryption, describe the DDH assumption, and

sketch the HVZK notion. In section 3, I present the protocol and sketch its security. I implement the general k-shuffle as described in this section in the Go programming language.[1].

2 Preliminaries

If X is a finite set, let $x \twoheadleftarrow X$ denote uniformly sampling an element x from the set. Let $y \leftarrow A(x)$ denote running A on input x and assigning its output to y. Let $y \leftarrow A(x;r)$ denote the execution of a probabilistic algorithm with the sequence of coins $r \in \{0,1\}^\infty$. Let $y \twoheadleftarrow A(x)$ denote choosing $r \twoheadleftarrow \{0,1\}^\infty$ and executing $y \leftarrow A(x;r)$. Let $[i..j]$ where $i \leq j$ denote the set of integers from i to j inclusive. Let $y \leftarrow \langle P, V\rangle(x)$ denote the execution of interactive Turing machines P and V on common input x. When the protocol finishes, the output of V is assigned to y. A function $\Delta(n)$ is negligible if for all positive polynomials $p(\cdot)$ and sufficiently large n, $\Delta(n) < \frac{1}{p(n)}$.

2.1 ElGamal encryption

Fix a prime number p and let $\mathbb{Z}_p$ denote the modular ring of order p. In what follows, we assume arithmetic is performed in the modular ring $\mathbb{Z}_p$. Let $g \in \mathbb{Z}_p$ such that $g^q = 1$ where q is prime. Then $\langle g\rangle = \mathbb{Z}_q^*$ is a subgroup of $\mathbb{Z}_p^*$ and $q|(p-1)$. We call (p,q,g) the *public parameters*. The *secret-key* is chosen uniformly from the set of powers of g mod p: let $k \twoheadleftarrow [1..q-1]$. Let $K = g^k$ be the *public-key*.

Encryption of $m \in \{0,1\}^*$ proceeds as follows: the plaintext is encoded as $M \in \mathbb{Z}_p^*$. (Of course, only a subset of $\{0,1\}^*$ may be encoded for a fixed modulus p.) Let $r \twoheadleftarrow [1..q-1]$, $R = g^r$, $S = K^r$, and $C = M \cdot S$. The ciphertext is the tuple (R, C). Decryption of a ciphertext $(R, C) \in \mathbb{Z}_p^* \times \mathbb{Z}_p^*$ proceeds as follows: let $S = R^k$ and $M = C \cdot S^{-1}$. The plaintext m is recovered by decoding M.

This encryption scheme is known to be *semantically secure* under the decisional Diffie-Hellman assumption [4]. Informally speaking, this means that if given K, it is infeasible for any adversary to determine k, then given (C, R), it is infeasible for any adversary to learn anything meaningful about M (nor m). As we're not interested in the security of ElGamal from a privacy standpoint, we won't formalize semantic security. However, we define the DDH assumption.

2.2 Decisional Diffie-Hellman

Let (p,q,g) be the public parameters of the ElGamal cryptosystem. The game the adversary D plays is defined as follows: let $a, b, r \twoheadleftarrow [1..q-1]$ and let $A = g^a$, $B = g^b$, $Y_0 = g^r$, and $Y_1 = g^{ab}$. Choose a random bit $b \twoheadleftarrow \{0,1\}$. On input (A, B, Y_b), the adversary outputs a bit $b' \in \{0,1\}$ and wins if $b = b'$. The advantage of D is defined as

$$\Delta(q) = |\Pr[D(A,B,Y_1) = 1] - \Pr[D(A,B,Y_0) = 1]|$$

[1] Freely available at *github.com/cjpatton/shuffle*

The DDH assumption is that, for every polynomial-time adversary D, the function $\Delta(q)$ is negligible.

2.3 Honest verifier zero-knowledge

Zero-knowledge is defined in [5] with respect to the class of formal languages with interactive proof systems denoted **IP**. A language L has an interactive proof system if there exists a pair of probabilistic algorithms (P, V) such that for all $x \in L$, $\Pr[\langle P, V\rangle(x) = 1] = 1$, and for all $x \notin L$, $\Pr[\langle P, V\rangle(x) = 1] \leq 1/2$.

Let $L \in \mathbf{IP}$ exhibited by (P, V). We define security with respect to a game associated to adversary D, verifier V^*, and simulator M^*. Suppose that L is a finite set[2] and let $x \twoheadleftarrow L$. Let $y_1 \twoheadleftarrow \langle P, V^*\rangle(x)$ and $y_0 \twoheadleftarrow M^*(x)$ choose a bit $b \twoheadleftarrow \{0, 1\}$. On input (x, y_b), adversary D outputs a bit b' and wins if $b = b'$. Let $k = |L|$ and define the advantage of D as

$$\Delta(k) = |\Pr[D(x, y_1) = 1] - \Pr[D(x, y_0) = 1]|$$

We say that (P, V) is *computational zero-knowledge* (or just *zero-knowledge*) if for every probabilistic polynomial-time adversary D, for every probabilistic polynomial-time verifier V^*, there exists probabilistic polynomial-time simulator M^* such that $\Delta(k)$ is negligible. The security notion achieved by Neff's protocol is somewhat weaker: a proof system (P, V) is *honest verifier zero-knowledge* (HVZK) if for every probabilistic polynomial-time adversary D, there exists a probabilistic polynomial-time simulator M^* such that

$$|\Pr[D(x, \langle P, V\rangle(x)) = 1] - \Pr[D(x, M^*(x)) = 1]|$$

is a negligible function of k.

3 The Neff shuffle

Describe what the mix does with specified syntax.

3.1 The basic protocol and its security

Not useful yet ...

3.2 Simple k-shuffle

Not useful yet ...

3.3 General k-shuffle

This is what we actually implement.

[2] This is without loss of generality, since we consider polynomial-time adversaries.

References

[1] David Chaum, 1981. "Untraceable electronic mail, return addresses, and digital pseudonyms." `http://freehaven.net/anonbib/cache/chaum-mix.pdf`

[2] Taher Elgamal, "A public key cryptosystem and a signature scheme based on discrete logarithms." Appeared in *CRYPTO'84.*

[3] D. Chaum and T.P. Pedersen. "Wallet databases with observers." Appeared in *CRYPTO'92.*

[4] Yiannis Tsiounis and Moti Yung 1998. "On the security of ElGamal based encryption." Appeared in *PKC'98.*

[5] Oded Goldreich, 2001. *Foundations of Cryptography.*

[6] C. Andrew Neff, 2001. "A verifiable secret shuffle and its application to e-voting." `http://freehaven.net/anonbib/cache/shuffle:ccs01.pdf`

[7] Masayuki Abe and Hideki Imai, 2003. "Flaws in some robust optimistic mix-nets." Appeared in *ACISP'03.*