A new attack on Jakobsson Hybrid Mix-Net

Seyyed Amir Mortazavi
Tehran, Iran.
sa.mortezavi@gmail.com

Abstract

The Jakobsson hybrid Mix-net proposed by Jakobsson and Juels, is a very practical and efficient scheme for long input messages. But this hybrid Mix-net does not have public verifiable property.

In this paper a new attack to the Jakobsson hybrid Mix-net is introduced. This attack breaks the robustness of the hybrid Mix-net scheme, given that the corrupted first mix server and one of the senders collude with each other.

keyword:Mix-net,Hybrid Mix-net, Anonymity.

1. Introduction

Mix-net is a cryptographic tool for creating an anonymous channel between a group of senders and receivers. By using it the identity of senders to receivers is kept unknown. The most important security aspect of Mix-net design is maintaining the anonymity of senders. A properly designed Mix-net takes a group of encrypted messages as its input and generates a set of plaintexts and delivers them to receivers while keeping anonymity of senders. Ideas for Mix-net first appeared in Chaum's work [2] and since then it is used as a strong tool to design the anonymous channels. In [4] the different methods of design of anonymous channels are discussed. Any Mix-net consists of three participants; mix servers, senders and receivers. Senders encrypt their messages and deliver them to the first mix server. First mix server performs mixing operation over these input messages and produces its output.
Mixing operations include:
1- cryptographic operations (encryption or decryption algorithms) to make input messages anonymous.
2- random permutation on the input messages to eliminate any relationship between outputs and inputs of the mix server.

Outputs of each server are transferred to the next mix server, where the same mixing operations are carried out. The outputs of the last mix server are the plaintexts which are delivered to the receivers.

Mix-nets may be designed to have some features, such as correctness, privacy of senders, robustness and verifiability. Desirable goal in the design of Mix-net is achieving these features with high computational efficiency. Correctness means that the result is correct, if all the mix servers are honest. In Mix-net the privacy is dealt with the anonymity of senders. A Mix-net is considered to be robust, if mix servers could produce a correct output in presence of some faulty mix servers. In the verifiable Mix-net each mix server is required to prove that it performs its tasks correctly.

In 1981 Chaum introduces the Mix-net in [2] for the first time, that is now known as decryption Mix-net. In a decryption Mix-net the senders encrypt their messages by the public key of all servers in a reverse order. During the decryption

operation of each server, a layer of encryption is decrypted. Thus the output of the last mix server will be plaintext messages. Not being robust to a faulty mix server was the major drawback of the decryption Mix-net, It means that if a mix server refuses to decrypt its input messages , mixing operations of Mix-net will stop. To achieve robustness against faulty servers, Park et.al in [14] propose a new Mix-net, in which decryption is substituted by re-encryption. Re-encryption Mix-net formerly introduced in [14], later shown to have some weaknesses in [17]. A new re-encryption Mix-net is introduced in [12] that is robust to above attacks. Since then the main objective of Mix-net designers was reaching a Mix-net with public verifiability and high efficiency. The first publicly verifiable Mix-net is presented in [7] that uses the cut and choose method. But this scheme has low efficiency and high complexity. After that, a lot of publicly verifiable Mix-nets with high efficiency was introduced [1, 6, 10, 8]. Sako in [5] and Neff in [11] present two practical publicly verifiable Mix-nets with high efficiency.

Schemes that are introduced above achieve their best efficiency when length of input ciphertext is short. A method for mixing long messages with these Mix-nets, is to divide each message into blocks of the short bits, re-encrypt each block and concatenate them to form a string. This approach requires only public key operations and proving correctness of mixing operations for each block.

Mix-nets usually use the public key cryptosystem (such as ElGamal or RSA cryptosystem) to re-encrypt or decrypt their input messages. Using the public key cryptosystem in designing a Mix-net has caused some problems like restriction of input message length and reduction of efficiency because of modular computations. Considering these statements, Mix-nets that use both symmetric and public keys were designed and known as hybrid Mix-nets. Accepting input messages with arbitrary length is the advantage of using symmetric key in the Mix-net design. Furthermore, encryption with symmetric key is more efficient than encryption with public key. Abe et.al in [13] have introduced first hybrid Mix-net that has robustness against $\sqrt{n}$ faulty mix servers from total of n mix servers. Jakobsson et.al in [9] proposes a hybrid Mix-net that has robustness against $\frac{n}{2}$ faulty mix servers. The Mix-net schemes in [13, 9], are able to mix long messages efficiently, but they do nor have the property of public verifiability.

In this paper a review of the Jakobsson hybrid Mix-net scheme carried out at section 2 is presented. The new attack against this hybrid Mix-net is introduced in section 3, this attack breaks robustness of this hybrid Mix-net. In section 4 the clue to a solution of modification this scheme against our new attack is presented. This proposed scheme is a public verifiable Mix-net.

2 Review of "Hybrid Mix-Net"

In this section the hybrid Mix-network protocol presented by Jakobsson et al is reviewed briefly. [9]. The description given here is as close as possible to the original paper, but we discard details irrelevant to the new attack. For details we refer the interested reader to [9].

2.1 Setup and building blocks

The participants of the this hybrid Mix-net protocol are; N senders that denoted by $P_1, ..., P_N$, and n mix servers that denoted by $S_1, ..., S_n$, and a bulletin board[1]. Each sender encrypts its message, and writes it on the bulletin board. The mix servers then run the hybrid Mix-net protocol.

In the remaining of this subsection, the notations and preliminary blocks for the hybrid Mix-net protocol are introduced.

[1]bulletin board is a publicly shared memory which all participants can read and write access to it with authentication but no participant can erase anything on the bulletin board [9, 18]

2.1.1 Scheme parameters

First the parameters used in this scheme are explained. Let p and q be two large primes that $q \mid p-1$ and Z_p^* is a multiplicative group module p and G_q is a subgroup of order q of Z_p^*. The generator of subgroup G_q is denoted with g.
Message authentication code (MAC) is a symmetric keyed function. MAC for message m with symmetric key z is denoted with $MAC_Z(m)$.
let $k \in G_q$ be a symmetric key shared between two entity, $E_k[m] = c$ and $D_k[c] = m$, are respectively the encryption and decryption function of the symmetric key algorithm.
Zero knowledge proof for proving quadruple $(a, b, y, z) \in G_q^4$ is in the form of $\log_a b = \log_y z = x$ and the prover knows the secret value of x is denoted with EQDL$[a, b, y, z]$.

2.1.2 Initialization

Each server S_i, $1 \leq i \leq n$, chooses three secure and random keys $\alpha_i, \beta_i, \gamma_i \in Z_q$, as its private keys, then each server computes and publishes triple (Y_i, K_i, Z_i) that $Y_i = {Y_{i-1}}^{\alpha_i}, K_i = {Y_{i-1}}^{\beta_i}, Z_i = {Y_{i-1}}^{\gamma_i}$ as its public key. In this computation, it is assumed that $Y_0 = g$.
In the initialization of this scheme each server must prove knowledge of its private keys by using a zero knowledge proof such as [16] or [3]. After that all the private keys of each server are shared between all the servers by using (t, n) verifiable secret sharing (VSS) technique [15].

2.1.3 Simulated server

By cooperations of all servers and using VSS technique the last server S_{n+1} with private keys $\beta_{n+1}, \gamma_{n+1}$ and public keys K_{n+1}, Z_{n+1} are simulated and these keys are shared between the other servers.

2.2 Encryption

In this section, encryption algorithm, that is used by senders to encrypt their messages is described. This encryption algorithm is based on the symmetric key encryption, that encrypts the message layer to a layer with all the public keys of the servers in a reverse order.

1. **Compressed key schedule Generation**
   Eeach sender, like P_j, selects randomly secret key $\rho_{(j)} \in Z_q$ then computes

$$\begin{cases} k_i^{(j)} = {K_i}^{\rho} & 0 \leq i \leq n+1 \\ z_i^{(j)} = {Z_i}^{\rho} & 1 \leq i \leq n+1 \\ y_0^{(j)} = {Y_0}^{\rho_{(j)}}. \end{cases} \tag{2.1}$$

   notice that the sender compressed key schedule is y_0 [9].

2. **Message Encryption**
   Each sender encrypts his message in this stage. For example the jth sender encrypts the m_j by computing:

$$\begin{cases} c_n = E_{k_{n+1}^{(j)}}[m_j] \\ c_j = E_{k_{i+1}^{(j)}}[c_{i+1} \parallel \mu_{i+1}] & 0 \leq i \leq n-1 \\ \mu_j = MAC_{z_{i+1}^{(j)}}[c_i \parallel I] & 0 \leq i \leq n \end{cases} \tag{2.2}$$

   where I is a publicly random string.
   the ciphertext of the jth sender is $\{{c_0}^{(j)}, {\mu_0}^{(j)}, {y_0}^{(j)}\}$ and other senders similarly construct their ciphertexts. For the security of the scheme, the length of all the ciphertexts must be equal. The set of ciphertext sent to first mix server is shown by $\{{c_0}^{(j)}, {\mu_0}^{(j)}, {y_0}^{(j)}\}_{j=1}^{N}$.

2.3 Mixing operation

In this hybrid Mix-net, all of the transactions and interactions between Mix-net participants are public and performed by means of bulletin board.

1. Each server takes the set $\{{c_{i-1}}^{(j)}, {\mu_{i-1}}^{(j)}, {y_{i-1}}^{(j)}\}_{j=1}^{N}$ as its input and performs following steps:

   (a) key generation
   Server S_i by using its privet keys computes decryption keys as follows for $1 \leq j \leq N$.

$$\begin{cases} \widetilde{y_i}^{(j)} = \left({y_{i-1}}^{(j)}\right)^{\alpha_i} \\ \widetilde{k_i}^{(j)} = \left({y_{i-1}}^{(j)}\right)^{\beta_i} \\ \widetilde{z_i}^{(j)} = \left({y_{i-1}}^{(j)}\right)^{\gamma_i} \end{cases} \quad (2.3)$$

   (b) MAC verification
   Server S_i verifies the MAC validity by:

$$\mu_{i-1}^{(j)} \stackrel{?}{=} \text{MAC}_{\widetilde{z}_i^{(j)}}[c_{i-1}^{(j)} \parallel I] \quad (2.4)$$

   for $1 \leq j \leq N$. If the MAC verification is wrong, i.e above equation is not satisfied, the server S_i performs Verify Complaint (i, j) (section 2.4).

   (c) Message Decryption
   Server S_i decrypts the ciphertext as followes:

$$D_{\widetilde{k}_i^{(j)}}[c_{i-1}^{(j)}] = (\widetilde{c}_i^{(j)} \parallel \widetilde{\mu}_i^{(j)}) \quad (2.5)$$

   (d) Permutation
   Server S_i chooses randomely permutation π_i and rearranges set $\{{c_i}^{(j)}, {\mu_i}^{(j)}, {y_i}^{(j)}\}_{j=1}^{N}$ and the result is the output of server S_i.

   (e) proof of correctness of output keys
   Server S_i for proving correctness of output keys $\{y_i^{(j)}\}_{j=1}^{N}$ uses zero knowledge proofs as described in section 2.1.1. Server S_i must prove that $P_i = P_{i-1}^{\alpha_i}$ and $Y_i = Y_{i-1}^{\alpha_i}$ by using EQDL$[P_{i-1}, P_i, Y_{i-1}, Y_i]$, where $P_i = \prod_{j=1}^{N} y_i^{(j)}$. S_i must send the required information for prooving the EQDL equations to server S_{i+1} and if S_{i+1} determines that the proof is incorrect it performs Verify Complaint (section 2.4).

2. Output
   the output of S_n is set $\{{c_n}^{(j)}, {\mu_n}^{(j)}, {y_n}^{(j)}\}_{j=1}^{N}$ and this set is the input of the simulated server, S_{n+1}. With help of the Mix-net servers and using VSS techniques, the privite keys of the $(n+1)$th server $z_{n+1}^{(j)}, K_{(n+1)}^{(j)}$, are computed. By using these keys, the mixing operations are performed and Mix-net outputs are obtained with decryption $\{D_{k_{n+1}^{(j)}}[c_n^{(j)}] = m^{(j)}\}_{j=1}^{N}$.

2.4 Verify Complaint (i, j)

Verify Complaint (i, j) are executed when the mix server S_i complains that its jth input message $(c_{i-1}^{(j)}, \mu_{i-1}^{(j)}, y_{i-1}^{(j)})$ has invalid form or the $(i-1)$th server is faulty. If the input message of the mix server S_i is not satisfied, the MAC verification (equations (2.4)) or EQDL proof, this algorithm with complaint of S_i is run.
With cooperation of all the servers and using secret sharing method, first the truth of complaint is verified and then the faulty server S_i or invalid form of the sender's message [9] is inspected.

3 A practical attack on Jakobsson hybrid Mix-net

In this section, a new attack against Jakobsson hybrid Mix-net [9] is introduced by the authors. This attack breaks the robustness of the hybrid Mix-net. In this attack, the corrupted first mix server and one of the senders, cooperate together to produce arbitrary outputs, without detection by the other servers.

3.1 Corrupt mix server and corrupt sender

We assume that the first mix server S_1 and one of the senders,P_N, are corrupted and cooperate

together to break the correctness and robustness of hybrid Mix-net. In this attack the first mix server could remove its input messages and replace them with arbitrary messages without detection by other participants. This attack is the general form of similar attacks in [18, 17].

Input messages to the first mix server are $\{{c_0}^{(j)}, {\mu_0}^{(j)}, {y_0}^{(j)}\}_{j=1}^{N}$, that $y_0^{(j)} = Y_0^{\rho_{(j)}}$. In this statement, $\rho_{(j)}$ is secret for the senders.
Is assumed that the Nth sender is corrupted and makes its message as follows:

$$y_0^{(N)} = \Pi_{j=1}^{N-1} y_0^{-(j)}\, Y_0^{\rho_{(N)}} \tag{3.1}$$

Then formes $\{{c_0}^{(N)}, {\mu_0}^{(N)}, {y_0}^{(N)}\}$ and post it to the first server.

Since the messages are public in the bulletin board, P_N could construct $y_0^{(N)}$ in the form of (3.1) equations. In the above equations, ${c_0}^{(N)}$ and ${\mu_0}^{(N)}$ are random numbers, in a way that the structure of the message is correct and $\rho_{(N)}$ is random and common between the first mix server and Nth sender. $y_0^{-(j)}$ is the inverse of a $y_0^{(j)}$ in module p.
The first mix server could modify messages to pass the verification processes.
The first mix server chooses arbitrarily and uniformly $m'_1, m'_2, ..., m'_N$ and $\rho'_{(j)} \in Z_q$ for $1 \le j \le N$, that $\sum_{j=1}^{N} \rho'_{(j)} = \rho_{(N)} \bmod q$. The First mix server computes for $1 \le j \le N$:(corresponding (2.1) equations)

$$\begin{cases} \rho''_{(j)} = \rho'_{(j)}\alpha_1 \\ k_i^{(j)} = {K_i^{(j)}}^{\rho''_{(j)}} & 1 \le i \le n+1 \\ z_i^{(j)} = {Z_i^{(j)}}^{\rho''_{(j)}} & 2 \le i \le n+1 \\ y_1^{(j)} = {Y_0^{(j)}}^{\rho''_{(j)}}. \end{cases} \tag{3.2}$$

and it could encrypt messages m'_j, as follows: (corresponding (2.2) equations):

$$\begin{cases} c_n = E_{k_{n+1}^{(j)}}[m'_i] \\ c_i = E_{k_{i+1}^{(j)}}[c_{i+1} \parallel \mu_{i+1}] & 1 \le i \le n-1 \\ \mu_i = MAC_{z_{i+1}^{(j)}}[c_i \parallel I] & 1 \le i \le n \end{cases} \tag{3.3}$$

The first mix server could construct the set $\{{c_1}^{(j)}, {\mu_1}^{(j)}, {y_1}^{(j)}\}_{j=1}^{N}$. The server S_1 rearrangees this set and adds the knowledge for proving EQDL proof to this ordered set, then posts it to the second server. Second server follow usual manner of protocol corresponding section 2.3:

1. Key generation
   S_2 computes $k_2^{(j)}$ and $z_2^{(j)}$ for $1 \le j \le N$ from equations (2.3).

2. Verify MAC equations
   S_2 with using equations (2.4), verifies MAC equations for all messages. Since the first mix server formes these messages properly, all the messages satisfy the MAC equations.

3. Message decryption: Server S_2 by using the equations (2.5) decrypts all the messages.

4. EQDL equations
   The server S_2 must check the EQDL proof, EQDL$[P_0, P_1, Y_0, Y_1]$, corresponding with section 2.3. Since:
   $P_0 = \Pi_{j=1}^{N} y_0^{(j)} = Y_0^{\rho_{(N)}}$ and $P_1 = \Pi_{j=1}^{N} y_1^{(j)} = Y_0^{\rho_N \alpha_1}$ $Y_1 = Y_0^{\alpha_1}$ and server S_1 knows the value of α_1, consequently it could create EQDL zero knowledge proof correctly.

The server S_1 could replace the messages of senders with other arbitrary messages and send new messages to the second server without detection by other participants. Similarly the messages are sent to the last mix server and the outputs of Mix-net are formed.

4 Modified and publicly verifiable hybrid Mix-net

In the hybrid Mix-net, senders must post $y_0^{(j)}$ to the first mix server, and the senders don't need to prove that they know the exponents of the $y_0^{(j)}$,i.e, $\rho_{(j)}$. by the use this weakness, the new attack on the hybrid Mix-net is introduced. For modifying the scheme against this attack, a clue for the solutions of scheme is proposed.

a new method for reaching public verifiability in the hybrid Mix-networks is introduced. Verification of MAC function in Jakobsson hybrid Mix-net is performed only by next server, since in this scheme the MAC function is a keyed function and only the next server has this key. The digital signature is used for reaching the public verifiability.

For using ElGamal digital signature, firstly the public and private keys of ElGamal signature are introduced as follows:
(keys of the jth sender for signing the message of the ith server)
public key:$(Y_i, y_i^{(j)}, p, q)$
private key: $\rho_{(j)}$
notice that $y_i^{(j)} = Y_i^{\rho_{(j)}}$ and Y_i (the public key of ith server) is a generator in group G_q. (q is a prime number and each exponent of g is a generator)
ElGamal signature with generator Y_i and private key $\rho_{(j)}$ over message m is denoted by $d = \text{SIG}_{\rho_j, Y_i}(H(m))$ and the signature verification by $H(m) = \text{VER-SIG}_{(Y_i, y_i)}(d)$. In the above equations H is a secure hash function.
The message encryption algorithms in section 2.2 are changed with new algorithms as following:

$$\begin{cases} c_n = E_{k_{n+1}^{(j)}}[m_i] \\ c_i = E_{k_{i+1}^{(j)}}[c_{i+1} \parallel \mu_{j+1}] & 0 \le i \le n-1 \\ \mu_i = SIG_{(\rho_{(j)}), Y_i)}[c_i] & 0 \le i \le n \end{cases} \tag{4.1}$$

and the MAC verification in section 2.3 is replaced with digital signature verification as following:

$$c_i^{(j)} = \text{VER-SIG}_{(Y_i, y_i^{(j)})}(\mu_i^{(j)}) \tag{4.2}$$

for $1 \le j \le N$ If the digital signature is used instead of MAC, the efficiency of the scheme is reduced, since the complexity of digital signature is higher than MAC verification.
In the below table the complexity of both schemes are compared. The comparison is based on the number of exponentiation for both schemes. n is the number of servers and N is the number of senders.

Mix-net	initializing	Encryption	Mixing
Jakobsson	$3n+9$	$2n+3$	$3N+6$
Our scheme	$2n+6$	$2n+3$	$5N+6$

5 Conclusion

Hybrid Mix-nets are efficient Mix-nets for long input messages. Hybrid Mix-nets are constructed by use of both symmetric and public key encryption. In this paper a new attack on Jakobsson hybrid Mix-net is introduced, that breaks the robustness of the scheme. A new scheme that is resistent against the new attack is proposed. The proposed solution is a public verifiable hybrid Mix-net that is based on Jakobsson hybrid Mix-net.

References

[1] Masayuki Abe, *Universally verifiable mix-net with verification work independent of the number of mix servers*, In Advances in Cryptology (EUROCRYPT 98), Springer-Verlag, LNCS 1403, 1998, pp. 437–447.

[2] David Chaum, *Untraceable electronic mail, return addresses, and digital pseudonyms*, Communications of the ACM **24** (1981), no. 2.

[3] David Chaum and Torben P. Pedersen, *Wallet databases with observers*, CRYPTO, 1992, pp. 89–105.

[4] George Danezis and Claudia Diaz, *A survey of anonymous communication channels*, Tech. Report MSR-TR-2008-35, Microsoft Research, January 2008.

[5] Jun Furukawa and Kazue Sako, *An efficient scheme for proving a shuffle*, Proceedings of CRYPTO 2001 (Joe Kilian, ed.), Springer-Verlag, LNCS 2139, 2001.

[6] Markus Jakobsson, *A practical mix*, Advances in Cryptology - EuroCrypt '98 (London, UK), Springer-Verlag, 1998, LNCS 1403, pp. 448–461.

[7] Joe Kilian and Kazue Sako, *Receipt-free MIX-type voting scheme - a practical solution to the implementation of a voting booth*, Proceedings of EUROCRYPT 1995, Springer-Verlag, 1995.

[8] Jakobsson Markus, *Flash Mixing*, Proceedings of Principles of Distributed Computing - PODC '99, ACM Press, 1999.

[9] Jakobsson Markus and Juels Ari, *An optimally robust hybrid mix network (extended abstract)*, Proceedings of Principles of Distributed Computing - PODC '01, ACM Press, 2001.

[10] Abe Masayuki, *Mix-networks on permutation networks*, ASIACRYPT, 1999, pp. 258–273.

[11] C. Andrew Neff, *A verifiable secret shuffle and its application to e-voting*, ACM Conference on Computer and Communications Security, 2001, pp. 116–125.

[12] Wakaha Ogata, Kaoru Kurosawa, Kazue Sako, and Kazunori Takatani, *Fault tolerant anonymous channel*, Information and Communications Security - ICICS 97 ,LNCS 1334, 1997, pp. 440–444.

[13] Miyako Ohkubo and Masayuki Abe, *A Length-Invariant Hybrid MIX*, Proceedings of ASIACRYPT 2000, Springer-Verlag, LNCS 1976, 2000.

[14] Choonsik Park, Kazutomo Itoh, and Kaoru Kurosawa, *Efficient anonymous channel and all/nothing election scheme*, EUROCRYPT '93: Workshop on the theory and application of cryptographic techniques on Advances in cryptology, 1994.

[15] Gennaro Rosario, Jarecki Stanislaw, Krawczyk Hugo, and Rabin Tal, *Secure distributed key generation for discrete-log based cryptosystems*, J. Cryptology **20** (2007), no. 1, 51–83.

[16] Claus-Peter Schnorr, *Efficient signature generation by smart cards*, J. Cryptology **4** (1991), no. 3, 161–174.

[17] Birgit Pfitzmann Universitiit and Birgit Pfitzmann, *Breaking an efficient anonymous channel*, In EUROCRYPT, Springer-Verlag, 1995, pp. 332–340.

[18] Douglas Wikström, *Five practical attacks for "optimistic mixing for exit-polls"*, Selected Areas in Cryptography (10th Annual International Workshop, SAC 2003), 2004, pp. 160–175.