Verifiable Encryption in Anonymous Ad Hoc Groups

Joseph K. Liu[1], Victor K. Wei[1], and Duncan S. Wong[2]

[1] Department of Information Engineering
The Chinese University of Hong Kong
Shatin, Hong Kong
{ksliu9,kwwei}@ie.cuhk.edu.hk

[2] Department of Computer Science
City University of Hong Kong
Kowloon, Hong Kong
duncan@cityu.edu.hk

Abstract. In a verifiable encryption, an asymmetrically encrypted ciphertext can be publicly verified to be decypherable by a designated receiver without revealing the plaintext. In this paper, we introduce publicly verifiable encryption that is intended for a single anonymous decypherer within an ad hoc group of n receivers. The verifier can ascertain that the ciphertext can be decrypted by one receiver, but it cannot find out the identity of the targeted decypherer among the n receivers in the ad hoc group. Our scheme is *spontaneous*: the prover does not need the collaboration from any exterior party such as TTP, group manager, or any member of the receiver group when forming this ad hoc group of n receivers. We also introduce two extensions. In the first extension a targeted subset of t receivers jointly recover the message. In the second extension, any member of a targeted subset of t receivers can recover the encrypted message. Both extensions preserve the anonymity of the targeted subset.

Feb 17, 2004.

1 Introduction

Consider the following scenario. Alice wants to send a public-key encrypted message to Bob, who works for ABC Company. The company security gateway does not allow the message in unless it is for a company employee. However, Bob does not wish to divulge his private key. Without knowing Bob's private key, how can the gateway ensure the message is intended for a company employee? Furthermore, Alice and Bob do not want the company gateway to know that Bob is the actual recipient. By knowing only the public information of the company employees, the verifier has to determine if the encrypted incoming data is for a company employee without being able to identify the actual recipient. In addition, other employees of the company should not be involved in the secret communication between Alice and Bob and should be totally unaware of the entire process. In this paper, we present a solution to this problem.

In a *verifiable encryption scheme for anonymous ad hoc groups*, a *Prover* wishes to send a public-key encrypted message to a *Receiver* through a *Verifier*. The Prover arbitrarily and spontaneously forms a group consisting of the targeted decypherer and $n-1$ diversion receivers; conducts a special public-key encryption for the group of n in such a way that the public verifier can ascertain the message can be decrypted by at least one of the group members. In addition, the verifier cannot identify the targeted decypherer from the group, and the verifier cannot read the message. We propose this scheme as a solution to the motivating application above. By spontaneously forming a group of decypherers, the Prover does not need collaboration of any exterior party such as TTP, group manager, or any member of the group. The group members are totally unaware of being enrolled in the group. We call this spontaneously formed group with the requirement of maintaining anonymity of the group members as an Anonymous Ad Hoc Group (AAHG).

We also introduce two extensions. In the first extension a *targeted* subset of t receivers in an AAHG jointly recover the message. In the second extension, any member of a *targeted* subset of t receivers in an AAHG can recover the encrypted message. Both extensions preserve the anonymity of the targeted subset.

These extensions can be useful in the following scenario. Bob belongs to a cluster of t members in an ad hoc group of n members. For example, the cluster can be a small unit in a temporarily formed task force for a special mission. Our extension schemes can be used to transmit confidential messages to the unit, or to unit members, while keeping non-unit members of the task force and the security gateway of the task force at bay.

In a verifiable encryption scheme [13, 1, 4, 2, 8], a prover is able to convince a verifier that a public-key encrypted message can be recovered by a designated receiver, whose identity is known to the verifier. Our scheme is about a group of possible receivers in which at least one designated receiver can recover the message. In a verifiable group encryption scheme [4], any subset of t members out of a group of n receivers can jointly recover the message. When $t = 1$, it ensures that *any* receiver can decrypt the message. Our scheme is about a single *targeted* but anonymous receiver (and targeted but anonymous receivers in the extension schemes), not a threshold of any t receivers.

1.1 Contributions

We introduce the notion of *verifiable encryption for anonymous ad hoc groups* (VE-AAHG). It allows a prover to arbitrarily and spontaneously form a group of n receivers, and prepare an encrypted message which can be recovered by one targeted group member, in a way such that a public verifier can make sure that the encrypted message can be recovered by at least one targeted group member, yet the verifier knows nothing about the identity of the targeted decypherer. The n receivers can be totally unaware of being included in the group and no TTP or group manager is involved.

We present a VE-AAHG scheme which is perfectly separable [10, 7]. A cryptographic protocol possesses separability if the participants can choose their keys independently of each other. In our scheme, the receivers can not only choose their keys independently of each other and even use different kinds of public-key encryption schemes. For spontaneity, the only assumption is that each receiver has a public key associated to and it is publicly known or under the existence of PKI.

We also make two extensions to the basic VE-AAHG. The first one requires a *targeted* subset of t receivers out of n receivers to work jointly in decrypting the message. The second extension allows any member of a *targeted* subset of t receivers to decrypt the message. Both extensions preserve the anonymity of the targeted decypherers.

This paper is organized as follows. We describe some related work in Sec. 2. This is followed by our security model specified in Sec. 3. Our VE-AAHG scheme is described in Sec. 4 and two extensions are presented in Sec. 5 and Sec. 6. The paper is concluded in Sec. 7.

2 Related Work

Stadler [13] introduced the concept of verifiable encryption scheme (VES) in the context of publicly verifiable secret sharing schemes (PVSS) [9] and presented two schemes. One scheme allows a public verifier to determine if a ciphertext contains the discrete logarithm of a given value without decrypting it. The scheme uses the cut-and-choose methodology. Another scheme is for a ciphertext containing the eth-root of a given value. The scheme is very efficient and does not use the cut-and-choose methodology. Later, Asokan, et al. [1] presented a very general VES for encryption of pre-image of any homomorphic one-way function. Their scheme also provides *perfect separability* in such a way that the scheme can take any type of encryption algorithm and encryption key associated to the receiver. To improve the efficiency of VES for the encryption of discrete logarithms, Bao, et al. [2] proposed the first one of this type without using the cut-and-choose methodology. Recently, several other VES's [4, 6, 8] have been proposed and achieved provable security under various security models.

In all the above schemes the verifier knows the identity of the receiver. An anonymous verifiable encryption scheme proposed by Camenisch, et al. [5] hides the identity of the receiver from the verifier. However, their scheme requires the prover to know the private key of the receiver. In another approach, Camenisch, et al. [4] extended the concept of VES to threshold VES (in [4], the authors call it a verifiable group encryption scheme). The threshold VES allows the verifier to determine if a ciphertext can only be decrypted when at least t arbitrary receivers are working jointly. Hence our notion of VES in anonymous ad hoc groups is very different from theirs in the way that we require the ciphertext to be recovered by one particular receiver in a group of possible receivers without revealing who this receiver is.

3 Security Model

In this section we define the security model to be used. Let $\mathsf{Domain}(f)$ and $\mathsf{Range}(f)$ specify the domain and range of a function f. Let $k \in \mathbb{N}$ be a system-wide security parameter. For $i = 1, \cdots, n$, let E_i and D_i denote the probabilistic polynomial time (PPT) public-key encryption and decryption functions of receiver i, respectively. We adopt the most primitive security requirement for asymmetric encryption scheme. No special assumptions are needed on the encryption scheme. A triple (G, E, D) of PPTs is a secure public key encryption scheme if for any $(E, D) \in G(1^k)$ such that $D(E(m)) = m$ for any $m \in \mathsf{Domain}(E)$, any PPT algorithm A and all sufficiently large k, we have

$$Pr[A(1^k, E, c) = m : m \in_R \mathsf{Domain}(E), c \leftarrow E(m)] \leq \Delta(k)$$

where Δ is some negligible function. A real-valued function $\Delta(k)$ is *negligible* if for every $c > 0$, there exists a $k_c > 0$ such that $\Delta(k) < k^{-c}$ for all $k > k_c$. $x \in_R X$ denotes an element x chosen randomly from X.

Verifiable Encryption Scheme for Anonymous Ad Hoc Groups. A verifiable encryption scheme for anonymous ad hoc groups (VE-AAHG) consists of a two-party PPT protocol between P (Prover) and V (Verifier), a PPT algorithm R (Recovery Algorithm), also known as the Decryption Algorithm, and a one-way function f. P accepts as inputs k, some appropriate binary string m (a *message*), n public-key encryption functions $\{E_i\}_{1 \leq i \leq n}$, and an integer $\pi \in \{1, \cdots, n\}$ (the index of the targeted decypherer). V accepts as inputs k, and $\{E_i\}_{1 \leq i \leq n}$ only. If the protocol completes successfully without early termination, V outputs two finite binary strings $d = f(m)$ (commitment) and $\mathcal{C}$ (ciphertext). The Recovery Algorithm $R : (d, \mathcal{C}, D_i) \mapsto \{m', NULL\}$ accepts as inputs the commitment, the ciphertext and one of the n decryption functions $\{D_i\}_{1 \leq i \leq n}$ corresponding to $\{E_i\}_{1 \leq i \leq n}$ and outputs either a finite string m' or the $NULL$ string.

On the security of the VE-AAHG scheme, we require that a secure scheme satisfies the following requirements when k is sufficiently large.

– *Targeted Decypherability*:
  (*Completeness*) If both P and V are honest, then at the end of the protocol, V outputs $(d, \mathcal{C})$ such that $R(\mathcal{C}, D_\pi) = m$ and $d = f(m)$ for some $1 \leq \pi \leq n$. This holds for all m, d and $\mathcal{C}$ with $d = f(m)$. Honest P and V are defined as PPT algorithms which behave exactly as the Prover and Verifier described in the scheme, respectively.
  (*Soundness*) If V completes a protocol run without early termination and outputs d and $\mathcal{C}$, then V is sure that $\mathcal{C}$ is the encrypted value of d's inverse. Formally, we require that an honest V completes a protocol run without early termination and outputs a pair $(d, \mathcal{C})$ such that

$$Pr[V(1^k, E_1, \cdots, E_n) \rightarrow (d, \mathcal{C}) \ : \ f(R(\mathcal{C}, D_\pi)) \neq d] \leq \Delta(k)$$

for some $1 \leq \pi \leq n$ and all sufficiently large k. The probability is taken over all possible values of d, coin flips of the public key encryption and decryption functions, coin flips of V and R, and the values of π.

– *Anonymity*: The probability that V can determine π is negligibly higher than $1/n$. In general, we require that for any PPT algorithm, $\mathcal{V}$, and all sufficiently large k,

$$Pr[\mathcal{V}(1^k, E_1, \cdots, E_n, d, Trans, D_{i_1}, \cdots, D_{i_t}) \rightarrow \pi : f(R(\mathcal{C}, D_\pi)) = d] \leq \frac{1}{n-t} + \Delta(k)$$

where $Trans$ is the set of transcripts of an honest P, and $\{D_{i_1}, \cdots, D_{i_t}\}$ is an arbitrary set of t decryption functions in which each of them is corresponding to one in $\{E_1, \cdots, E_n\} \setminus \{E_\pi\}$. The probability is taken over the values of d, transcripts of an honest P, t-element sets of decryption functions, and coin tosses of P, public key encryption and decryption functions.

– *Confidentiality*: We require that any diversion receivers even they collude with each other cannot obtain the inverse of d. Formally, for any PPT $\mathcal{V}$, and all sufficiently large k,

$$Pr[\mathcal{V}(1^k, E_1, \cdots, E_n, d, \mathcal{C}, Trans, D_{i_1}, \cdots, D_{i_t}) \rightarrow m : d = f(m)] \leq \Delta(k)$$

where $Trans$ is the set of transcripts between honest P and V when V outputs $(d, \mathcal{C})$, and $\{D_{i_1}, \cdots, D_{i_t}\}$ is an arbitrary set of t decryption functions in which each of them is corresponding to one in $\{E_1, \cdots, E_n\} \setminus \{E_\pi\}$ and $f(R(\mathcal{C}, D_{i_j})) \neq d$, $1 \leq j \leq t$.

– *Public Verifiability*: V only requires publicly available parameters in order to perform its computations.

– *Spontaneity*: P does not require the cooperation of any exterior entity to perform its computations during the entire process of the VE-AAHG scheme.

Remark: There is no TTP (trusted third party), no group manager, no cooperation among group membership, in pre-processing or in computing the scheme itself. Decryption is not P's task. Public-key encryption functions are assumed to be publicly available.

4 A Verifiable Encryption Scheme for Anonymous Ad Hoc Groups

We specify our scheme in this section. The scheme uses the 3-choice cut-and-choose methodology. In each of N cut-and-choose rounds, a three-move protocol (commit, challenge, respond) is conducted between Prover P and Verifier V. V flips a three-way coin to issue one of three possible challenges. Depending on the challenge, P provides suitable response. If all cut-and-choose rounds are satisfactory, V outputs a commitment d and a ciphertext $\mathcal{C}$. Otherwise, it aborts. Each receiver i attempts to decypher using its own asymmetric decryption function D_i, $1 \leq i \leq n$. At least one receiver will succeed.

4.1 Preliminaries

Let (E_i, D_i), $1 \leq i \leq n$, be public-key encryption and decryption functions. Let π index the targeted receiver. Let p, q be large primes, $q \mid p-1$, and $g \in F_p$, order(g)=q. Let the security parameter k be as large as $|q|$. Let f be defined by $x \rightarrow g^x$ which is an instantiation of the one-way group homomorphism from $\mathbb{Z}_q$ to $< g >$. Let $m \in \mathbb{Z}_q$ be a message. Let N be the number of cut-and-choose rounds. Let $H_1 : \{0,1\}^* \rightarrow \{0,1\}^k$ and $H_2 : \{0,1\}^* \rightarrow \mathbb{Z}_q$ be some statistically independent and cryptographically strong hash functions. Sometimes, we may pass in an element in $\mathbb{Z}_q$ for encryption and we implicitly assume that certain appropriate encoding method is applied. If P computes any probabilistic public-key encryption function, P needs to send the corresponding coin flip sequence to V and the sequence is to be carried on wherever the original message goes. We do not explicitly specify such in the following.

Sym(n) denotes the symmetric group of order n. It consists of all permutations on n objects.

4.2 Detailed Description

Encryption.

1. P computes $d = g^m \bmod p$ and sends d to V.
2. Repeat the following steps N times in parallel.
   a. (*Commitment*) P randomly picks $s \in_R \mathbb{Z}_q$, $r_i \in_R \{0,1\}^k$ for $1 \leq i \leq n$, and $\phi \in_R$ Sym(n). P computes

$$\begin{aligned}
\lambda &= E_1(r_1)||\cdots||E_n(r_n) \\
\gamma &= (g^{H_2(r_{\phi(1)})} \bmod p,\ \cdots,\ g^{H_2(r_{\phi(n)})} \bmod p) \\
\alpha' &= E_1(s)||\cdots||E_n(s) \\
\alpha &= H_1(\alpha') \\
\beta &= g^{H_2(r_\pi)s} \bmod p \\
\theta &= H_1(\lambda||\gamma||\alpha||\beta)
\end{aligned}$$

   P sends θ to V.

   b. (*Challenge*) V picks $b \in_R \{1, 2, 3\}$ and sends b to P.

   c. (*Response*)
      – Case $b = 1$, P sends $r_1, \cdots, r_n$, γ, α and β to V
      – Case $b = 2$, P sends λ, γ, and s to V.
      – Case $b = 3$, P sends λ, γ, α', and $s' = H_2(r_\pi)s + m \bmod q$ to V.

   d. (*Verification* by V)
      – Case $b = 1$:
         • Verify that r_1, $\cdots$, r_n are distinct.

- Verify that there exists a unique permutation $\delta \in \mathrm{Sym}(n)$ such that

$$\gamma = (g^{H_2(r_{\delta(1)})} \bmod p,\ \cdots,\ g^{H_2(r_{\delta(n)})} \bmod p)$$

- Verify that

$$\theta = H_1(\hat{\lambda}||\gamma||\alpha||\beta)$$

where $\hat{\lambda} = E_1(r_1)|| \cdots ||E_n(r_n)$.

Continue only if all verifications succeed.

– Case $b = 2$:
- Denote $\gamma = (\gamma_1, \cdots, \gamma_n)$.
- Compute

$$\tilde{\alpha} = H_1(E_1(s)\ ||\ \cdots\ ||\ E_n(s))$$

and $\beta_i = \gamma_i^s \bmod p$, for $i = 1, \cdots, n$.
- Verify that

$$\theta = H_1(\lambda||\gamma||\tilde{\alpha}||\beta_i)$$

for exactly one index $i \in \{1, \cdots, n\}$.

Continue only if the verification succeeds.

– Case $b = 3$:
- Compute $\beta' = g^{s'}/d \bmod p$
- Verify that

$$\theta = H_1(\lambda||\gamma||H_1(\alpha')||\beta')$$

Continue only if the verification succeeds.

3. (*Output*) V terminates if any verification fails in any of the N cut-and-choose Rounds. Otherwise, it outputs d and the four-tuple sequences $(\alpha', \lambda, \beta', s')$ for all Case-(b=3) Rounds to all n receivers as the ciphertext, $\mathcal{C}$.

Fig. 1 illustrates the protocol.

Decryption. Denote $\bar{\lambda}_1|| \cdots ||\bar{\lambda}_n = \lambda$ and $\bar{\alpha'}_1|| \cdots ||\bar{\alpha'}_n = \alpha'$. For d and each four-tuple sequence $(\alpha', \lambda, \beta', s')$, each receiver i, $1 \leq i \leq n$, independently performs the following steps.

1. Compute $r_i = E_i^{-1}(\bar{\lambda}_i)$ and $s = E_i^{-1}(\bar{\alpha'}_i)$.
2. Compute $m' = s' - H_2(r_i)s \bmod q$.
3. Verify that $g^{s'} = g^{m'}\beta' \bmod p$. If the verification succeeds, then receiver i is the targeted decypherer and it outputs the decrypted message m' and halts. Otherwise, the receiver repeats the steps for another four-tuple sequence.

Prover P **Verifier V**

$d = g^m \bmod p$

$$\xrightarrow{\quad d \quad}$$

Repeat N times:

$s \in_R \mathbb{Z}_q$

$r_i \in_R \{0,1\}^k$, $1 \leq i \leq n$

$\phi \in_R \mathrm{Sym}(n)$

$\lambda = E_1(r_1)|| \cdots ||E_n(r_n)$

$\gamma = (g^{H_2(r_{\phi(1)})} \bmod p,\ \cdots,\ g^{H_2(r_{\phi(n)})} \bmod p)$

$\alpha' = E_1(s)|| \cdots ||E_n(s)$

$\alpha = H_1(\alpha')$

$\beta = g^{H_2(r_\pi)s} \bmod p$

$\theta = H_1(\lambda||\gamma||\alpha||\beta)$

$$\xrightarrow{\quad \theta \quad}$$

$b \in_R \{1, 2, 3\}$

$$\xleftarrow{\quad b \quad}$$

Case $b = 1$

$$\xrightarrow{\quad r_1, \cdots, r_n, \gamma, \alpha, \beta \quad}$$

$r_i \overset{?}{\neq} r_j,\ \forall i, j \in \{1, \cdots, n\}, i \neq j$

¿ $\exists!\ \delta \in \mathrm{Sym}(n),\ \gamma = (g^{H_2(r_{\delta(1)})}, \cdots, g^{H_2(r_{\delta(n)})})$

$\hat{\lambda} = E_1(r_1)|| \cdots ||E_n(r_n)$

$\theta \overset{?}{=} (\hat{\lambda}||\gamma||\alpha||\beta)$

Case $b = 2$

$$\xrightarrow{\quad \lambda, \gamma, s \quad}$$

$(\gamma_1, \cdots, \gamma_n) \leftarrow \gamma$

$\tilde{\alpha} = H_1(E_1(s)|| \cdots ||E_n(s))$

$\beta_i = \gamma_i^s \bmod p$, $1 \leq i \leq n$

¿ $\exists!\ i \in \{1, \cdots, n\},\ \theta = H_1(\lambda||\gamma||\tilde{\alpha}||\beta_i)$

Case $b = 3$

$s' = H_2(r_\pi)s + m \mod q$

$$\xrightarrow{\quad \lambda, \gamma, \alpha', s' \quad}$$

$\beta' = g^{s'}/d \bmod p$

$\theta \overset{?}{=} H_1(\lambda||\gamma||H_1(\alpha')||\beta')$

If no rejection in all N rounds,
output d and $(\alpha', \lambda, \beta', s')$ for all Case-$(b = 3)$ rounds

Note: The symbol ¿ reads as 'verify'.

Fig. 1. A verifiable encryption scheme for ad hoc groups.

4.3 Security Analysis

The overall probability that an honest verifier is cheated is no better than 3^{-N}. We believe that $N \approx 40 - 50$ should be sufficient for most applications in practice. In order to have an overwhelming chance of successful verification, P must provide compatible responses for all three cases in every cut-and-choose round. It keeps V honest. At the same time, V only views one of three possible responses. It prevents him from identifying the targeted decypherer.

Based on the proof and security analysis in Appendix A, we have the following theorem.

Theorem 1. *The VE-AAHG scheme described in Sec. 4 satisfies Targeted Decypherability, Anonymity, Confidentiality, Public Verifiability, and Spontaneity defined in Sec. 3, if the discrete logarithm problem is hard, the public key encryption functions E_i, $1 \leq i \leq n$, are secure, and H_1 and H_2 behave like ideal hash functions [3].*

We emphasize the importance of having H_2 to be a cryptographically strong hash function and making sure that H_2 is not used elsewhere in order to have the implementation of our basic scheme secure in practice. This prevents any possible interaction with the public key encryption functions especially considering the impact of given both $E(x)$ and $g^{H_2(x)}$ of some secret x. In our proofs in Appendix A, we always assume that H_2 behaves like a random oracle [3].

Our security definitions in Sec. 3 suggest the term "decypherment" to be getting the *entire* bit string of a message, m, with overwhelming probability for any m randomly chosen in $\mathbb{Z}_q$. Similar to conventional verifiable encryption schemes, d may leak certain bits of information of m. When m is short (for example, $d = g^m$ without mod p), m may even be able to recover from d directly. Hence our focus in this paper is on protecting a message from being recovered completely with non-negligible success rate if the message is randomly chosen from $\mathbb{Z}_q$. In practice, some efficient encoding mechanism can be deployed to eliminate this concern. Due to space limitation, we do not cover the details in this paper.

The proofs given in Appendix A suggest that the security of the scheme relies on the problem of inverting any of the underlying encryption schemes. Hence stronger public-key encryption functions such as those secure against adaptive chosen-ciphertext attack [11] can also be used in our scheme. This also leads us to believe that the scheme also enjoys *Perfect Separability*, in that each individual receiver can select a key pair arbitrarily and use a different kind of asymmetric cipher. Using a standard argument, our scheme trivially supports this property. We omit details in this paper.

4.4 Performance

Assume the length of p and the ranges of all the public key encryption functions are all l bits long. Note that other detailed security specifications of the public key encryption functions can be given but they would not affect the order of the scheme complexity. Therefore, we can safely simplify the performance evaluation by making the assumption above.

In one protocol run between P and V, the expected size of the transcripts is $N/3 \cdot ((6+n)k + (6n+1)l + 6) + l$ bits which is in $O((k+l)Nn)$. The expected size of the ciphertexts is $N/3 \cdot (k + (2n+1)l) + l$ which is in $O(lNn)$. Hence the complexity is linear in the size of the receiver group. For $k = 160$, $l = 1024$ and $N = 40$, the size of all the transcripts is 847KB for $n = 10$. It raises to 8MB for $n = 100$. The size of the ciphertexts would be 283KB and 2.6MB for $n = 10$ and $n = 100$, respectively. For resource-constrained systems where lightweight operation groups are used, such as elliptic curves defined over finite fields, the sizes of the transcripts and ciphertexts can be reduced three folds for the same security level.

On the network efficiency, all the N commit-challenge-respond rounds can be carried out in parallel. Hence there are only four message flows in one protocol runs.

5 Verifiable (t, t, n) Encryption for Anonymous Ad Hoc Groups

In our basic VE-AAHG scheme, only a single targeted member can decrypt the message. Here we make an extension such that a targeted t-member subset of the ad hoc group of n receivers can jointly recover the message. On the notation of (t, t, n), symbol 'n' represents that P spontaneously forms a group of n receivers; the second symbol 't' represents that t targeted members of the group can recover a message; and the first symbol 't' means that all the t targeted members need to work jointly to recover the message. By using similar notation, we propose another extension in Sec. 6 which allows *any* member of a targeted t-member subset of the ad hoc group of n receivers to recover the message. Hence the notation of the second extension is $(1, t, n)$.

Below is the verifiable (t, t, n) encryption scheme for AAHG.

Encryption. Here we use $\pi_1, \cdots, \pi_t$ to index the targeted receivers, where $t < n$ and $\pi_1, \cdots, \pi_t \in \{1, \cdots, n\}$ are distinct. The encryption algorithm is similar to the basic scheme described in Sec. 4.2, with the following modifications.

1. P also sends t to V before Commitment.
2. (Commitment) Compute as before, except
$$\beta = (g^{H_2(r_{\pi_1})} \cdot g^{H_2(r_{\pi_2})} \cdot \ldots \cdot g^{H_2(r_{\pi_t})})^s \bmod p$$
3. (Response) Compute as before, except that in Case b=3:
$$s' = (H_2(r_{\pi_1}) + \cdots + H_2(r_{\pi_t}))s + m \bmod q$$
4. (Verification)
   (a) Case b=1: Same as before.
   (b) Case b=2: Process γ, $\tilde{\alpha}$, β_i as before. Verify
$$\theta = H_1(\lambda \mid\mid \gamma \mid\mid \tilde{\alpha} \mid\mid \beta_{i_1} \cdots \beta_{i_t})$$
   for a unique t-element subset $\{i_1, \cdots, i_t\} \subset \{1, \cdots, n\}$.
   (c) Case b=3: No change.

Decryption. Same as before, except that t targeted decypherers jointly compute

$$m' = s' - (H_2(r_{\pi_1}) + \cdots + H_2(r_{\pi_t}))s \bmod q$$

6 Verifiable $(1, t, n)$ Encryption for Anonymous Ad Hoc Groups

We introduce another extension to our basic scheme, the verifiable $(1, t, n)$ encryption scheme for AAHG. Different from the verifiable (t, t, n) encryption scheme for AAHG described in Sec. 5, the verifiable $(1, t, n)$ encryption for AAHG allows *any* one in a targeted set of t receivers to recover the encrypted message.

Encryption. Let $\pi_1, \cdots, \pi_t$ be the index of t targeted receivers. The encryption algorithm is similar to the basic scheme described in Sec. 4.2, with the following modifications.

1. P also sends t to V before Commitment.
2. (Commitment) Same as before except
$$\beta = g^{H_2(r_{\pi_1})s} \bmod p \,||\cdots||\, g^{H_2(r_{\pi_t})s} \bmod p$$
3. (Response) Same as before except in Case b=3, replace the original s' with
$$s'_i = H_2(r_{\pi_i})s + m \bmod q$$
for $i = 1, \cdots, t$.
4. (Verification) Same as before except in
   (a) Case b=2, verify that
$$\theta = H_1(\lambda \,||\, \gamma \,||\, \tilde{\alpha} \,||\, (\beta_{i_1}||\cdots||\beta_{i_t}))$$
   for a unique t-member ordered tuples $\{i_1, \cdots, i_t\} \subset \{1, \cdots, n\}$.
   (b) Case $b = 3$, compute
$$\beta' = g^{s'_1}/d \bmod p \,||\, \cdots \,||\, g^{s'_t}/d \bmod p$$
5. (Output) Same as before except replacing the original s' with $s'_1, \cdots, s'_t$.

Decryption. The decryption algorithm is similar to the basic scheme described in Sec. 4.2, with the following modifications.

– Denote $\bar{\beta}'_1 \,||\, \cdots \,||\, \bar{\beta}'_t = \beta'$.
– Step 2 is modified as: receiver i computes $m'_{i,j} = s'_j - H_2(r_i)s \bmod q$ for $j = 1, \cdots, t$.

– Step 3 is modified as: receiver i checks if $g^{s'_j} \overset{?}{=} g^{m'_{i,j}}\bar{\beta}'_j \bmod p$ for $j = 1, \ldots, t$. If one of them equal, then receiver i is one of the targeted decypherers.

The modifications from our basic scheme to the (t, t, n) extension and $(1, t, n)$ extension cause further alterations. For example, the $(1, t, n)$ extension allows at least t targeted receivers to recover a message. This is because the t targeted receivers of one Case-$(b=3)$ round do not need to be identical to that of another Case-$(b=3)$ round. Similarly, the (t, t, n) extension allows at least one targeted t-member subset of a group of n receivers to work jointly to recover the message.

7 Concluding Remarks

In this paper, we propose a new notion of Verifiable Encryption in Anonymous Ad Hoc Groups (VE-AAHG) which allows the prover to spontaneously specify any set of n receivers and send an encrypted message such that the verifier can make sure that the encrypted message can be decrypted by at least one of the receivers. Yet the verifier knows nothing about the identity of the targeted receiver. The complexity of our proposing scheme is linear in the size of the receiver group.

We further propose two extensions to the basic scheme. The first one allows a targeted subset of t receivers out of n receivers to work together and recover the message. The second one allows any member of a targeted subset of t receivers out of n receivers to recover the message. Both extensions preserve the anonymity of those specific t receivers and their complexities are also linear in the size of the receiver group. All the proposed schemes also enjoy perfect separability. We consider these schemes to have many useful applications in practice.

We believe that other intriguing and efficient VE-AAHG schemes and various security models can be attained. Other variants and features may also be constructed. For example, it would be interesting to construct a general verifiable (k, t, n) encryption scheme for AAHG or a similar scheme which has the deniability property.

References

1. N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures. In *Proc. EUROCRYPT 98*, pages 591–606. Springer-Verlag, 1998. Lecture Notes in Computer Science No. 1403.
2. F. Bao. An efficient verifiable encryption scheme for encryption of discrete logarithms. In *Proc. Smart Card Research and Applications (CARDIS) 1998*, pages 213–220. Springer-Verlag, 2000. Lecture Notes in Computer Science No. 1820.
3. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proc. 1st ACM Conference on Computer and Communications Security*, pages 62–73. ACM Press, 1993.
4. J. Camenisch and I. Damgård. Verifiable encryption, group encryption, and their applications to separable group signatures and signature sharing schemes. In *Proc. ASIACRYPT 2000*, pages 331–345. Springer-Verlag, 2000. Lecture Notes in Computer Science No. 1976.

5. J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocations. In *Proc. EUROCRYPT 2001*, pages 93–118. Springer-Verlag, 2001. Lecture Notes in Computer Science No. 2045.
6. J. Camenisch and A. Lysyanskaya. An identity escrow scheme with appointed verifiers. In *Proc. CRYPTO 2001*, pages 388–407. Springer-Verlag, 2001. Lecture Notes in Computer Science No. 2139.
7. J. Camenisch and M. Michels. Separability and efficiency for generic group signature schemes. In *Proc. CRYPTO 99*, pages 413–430. Springer-Verlag, 1999. Lecture Notes in Computer Science No. 1666.
8. J. Camenisch and V. Shoup. Practical verifiable encryption and decryption of discrete logarithms. In *Proc. CRYPTO 2003*, pages 126–144. Springer-Verlag, 2003. Leture Notes in Computer Science No. 2729.
9. B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch. Verifiable secret sharing and achieving simultaneity in the presence of faults. In *Proc. 26th IEEE Symp. on Foundations of Comp. Science*, pages 383–395, Portland, 1985. IEEE.
10. J. Kilian and E. Petrank. Identity escrow. In *Proc. CRYPTO 98*, pages 169–185. Springer-Verlag, 1998. Lecture Notes in Computer Science No. 1642.
11. C. Rackoff and D. Simon. Noninteractive zero-knowledge proof of knowledge and chosen ciphertext attack. In *Proc. CRYPTO 91*, pages 433–444. Springer, 1991. Lecture Notes in Computer Science No. 576.
12. C. P. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3), 1991.
13. M. Stadler. Publicly verifiable secret sharing. In *Proc. EUROCRYPT 96*, pages 191–199. Springer-Verlag, 1996. Lecture Notes in Computer Science No. 1070.

A Proof of Theorem 1

Targeted Decypherability:

(*Completeness*) Clear.

(*Soundness*) Assume Prover P has an overwhelming probability of passing the verification in all cut-and-choose rounds. Then in each round, P must supply the following parameters in response to various challenge values generated by an honest V, and these parameters must pass all verifications in their respective Cases: $\check{\theta}$ (for Commitment); $\hat{r}_1, \cdots, \hat{r}_n, \hat{\gamma}, \hat{\alpha}, \hat{\beta}$ (for Case $b = 1$); $\tilde{\lambda}, \tilde{\gamma}, \tilde{s}$ (for Case $b = 2$); $\bar{\lambda}, \bar{\gamma}, \bar{\alpha}', \bar{s}'$ (for Case $b = 3$). We show below then there exists a unique decypherer. Furthermore, it recovers the message m accurately, for all $m \in \mathbb{Z}_q$.

Since H_1 is ideal, we have $\hat{\lambda} \mid\mid \hat{\gamma} \mid\mid \hat{\alpha} \mid\mid \hat{\beta} = \tilde{\lambda} \mid\mid \tilde{\gamma} \mid\mid \tilde{\alpha} \mid\mid \tilde{\beta} = \bar{\lambda} \mid\mid \bar{\gamma} \mid\mid \bar{\alpha} \mid\mid \bar{\beta}$ with overwhelming probability, where

$$\begin{aligned}
&\hat{\lambda} = E_1(\hat{r}_1) \mid\mid \cdots \mid\mid E_n(\hat{r}_n)\\
&\tilde{\alpha} = H_1(E_1(\tilde{s}) \mid\mid \cdots \mid\mid E_n(\tilde{s}))\\
&\tilde{\beta} = \tilde{\gamma}_\ell^{\tilde{s}}, \ell \text{ is the unique index found in Verification Case } b{=}2\\
&\bar{\alpha} = H_1(\bar{\alpha}')\\
&\bar{\beta} = g^{\bar{s}'}/d \bmod p
\end{aligned}$$

Combining and with overwhelming probability, we have

$$\bar{\beta} = g^{\bar{s}'}/d = \tilde{\beta} = \tilde{\gamma}_\ell^{\tilde{s}} = \hat{\gamma}_\ell^{\tilde{s}} = g^{H_2(\hat{r}_{\delta(\ell)})\tilde{s}} \bmod p$$
$$\bar{s}' - \bar{m} = H_2(\hat{r}_{\delta(\ell)})\tilde{s} \bmod q$$

where $\delta \in \mathrm{Sym}(n)$ is the permutation from Case b=1, and $\bar{m}$ is such that $g^{\bar{m}} = d$.

Denote $\bar{\lambda} = \bar{\lambda}_1 \mid\mid \ldots \mid\mid \bar{\lambda}_n$ and let $\bar{r}_i = E_i^{-1}(\bar{\lambda}_i)$, for $1 \le i \le n$. That $\bar{\lambda} = \hat{\lambda}$ implies $\bar{r}_{\delta(\ell)} = \hat{r}_{\delta(\ell)}$. Denote $\bar{\alpha}' = \bar{\alpha}'_1 \mid\mid \ldots \mid\mid \bar{\alpha}'_n$ and let $\bar{s}_i = E_i^{-1}(\bar{\alpha}'_i)$, for $1 \le i \le n$. That $\bar{\alpha} = \tilde{\alpha}$ implies $\bar{s}_{\delta(\ell)} = \tilde{s}$. Therefore,

$$\bar{m} = \bar{s}' - H_2(\bar{r}_{\delta(\ell)})\bar{s} = \bar{s}' - H_2(E_{\delta(\ell)}^{-1}(\bar{\lambda}_{\delta(\ell)}))E_{\delta(\ell)}^{-1}(\bar{\alpha}'_{\delta(\ell)}) \bmod q$$

Member $\delta(\ell)$, $1 \le \delta(\ell) \le n$, can decypher the message $\bar{m}$ satisfying $g^{s'} = g^{\bar{m}}\beta'$, where $\beta' = \bar{\beta}$. From Case b=2, δ is unique. From Case b=1, $\hat{r}_1, \cdots, \hat{r}_n$ are distinct and thus δ is unique. Therefore, the decypherer is unique with overwhelming probability.

We already have $g^{\bar{m}} = d$ above. In a Completeness proof, P is honest and $d = g^m$. Therefore, $\bar{m} = m$, the message recovery is accurate.

The above proves that if V satisfies with probability non-negligibly higher than 2/3 in an individual cut-and-choose round, then there exists a unique decypherer for that round. In our current cut-and-choose scheme, the unique decypherer from different rounds may differ.

Anonymity:

Assume the verifier V can compute the identity of the targeted decypherer with probability $1/n + \epsilon(k)$, where ϵ is a non-negligible function. We say that ϵ is non-negligible if there exists a polynomial ρ such that $\epsilon(k) > 1/\rho(k)$. Then V must solve one of the following problems with probability at least $1/n + \epsilon(k)$.

A. V can compute the identity (with probability at least $1/n + \epsilon(k)$) in Case $b = 1$ of an individual round.
B. V can compute the identity in Case $b = 2$ of an individual round.
C. V can compute the identity in Case $b = 3$ of an individual round.
D. V can compute the identity based on transcripts (commit, challenge, response) of multiple rounds, all of which are Cases $b = 3$.
E. V can compute the identity based on all N transcripts.

Problem A: Given $(r_1, \cdots, r_n, \gamma, \alpha, \beta)$ where $r_i \in \{0,1\}^k$, $1 \le i \le n$, $\gamma = (g^{H_2(r_{\phi(1)})}, \cdots, g^{H_2(r_{\phi(n)})})$, $\alpha = H_1(E_1(s)||\cdots||E_n(s))$, $\beta = g^{H_2(r_\pi)s}$ for some $\phi \in \mathrm{Sym}(n)$, $s \in F_q$, and $\pi \in \{1, \cdots, n\}$, find π.

In the following, we show in the random oracle model that if Problem A is easy, the discrete logarithm problem (DLP) is easy.

Lemma 1. *Suppose a PPT algorithm V, with H_1 and H_2 being random oracles, solves Problem A with probability at least $1/n + \epsilon(k)$. There exists a PPT algorithm $\mathcal{M}$, which invokes V and simulates the view of V by answering all its H_1-queries and H_2-queries, can compute the discrete logarithm problem (DLP) with probability at least $\frac{n}{n-1}\epsilon(k)$.*

We construct $\mathcal{M}$ as follows.

$\mathcal{M}$ = "On input $Y \in G$,

1. Randomly pick $r_1, \cdots, r_n \in_R \{0,1\}^k$ and $R_1, \cdots, R_n \in_R F_q$. Set the values of $H_2(r_i) = R_i$, for all $1 \leq i \leq n$.
2. Randomly pick $\alpha \in_R \{0,1\}^k$.
3. Set $\beta = Y$ and $\gamma = (g^{R_1}, \cdots, g^{R_n})$
4. Randomly generate secure asymmetric encryption functions $\bar{E}_1, \cdots, \bar{E}_n$ (whose decryption functions are generated by, and known to, $\mathcal{M}$).
5. Run V on corresponding inputs and reply all the queries of H_1 and H_2 in the following manner.
   - For any H_2-query with input r_i, $1 \leq i \leq n$, R_i is replied.
   - For a H_1-query with input $Z_1 || \cdots || Z_n$, compute $s'_i \leftarrow \bar{E}_i^{-1}(Z_i)$, $1 \leq i \leq n$, and determine if $s'_1 = \cdots = s'_n$ and $Y = g^{R_\ell s'_1}$, for some $1 \leq \ell \leq n$. If they are true, output $R_\ell s'_1$ and halt. Otherwise, randomly pick an element from $\{0,1\}^k \setminus \{\alpha\}$ as the reply.
   - For any other queries of H_1 and H_2, random numbers are generated in the corresponding range of H_1 and H_2 as the replies.
   - For query consistency, for any query with an input value which has been received before, the same reply as the last time is returned.
6. Halt with no output if V stops."

Since V is a PPT, the complexity of $\mathcal{M}$ is also in polynomial time. Let $\mathbf{E}$ denote the event that V queries H_1 with $Z_1 || \cdots || Z_n$ such that $s = \bar{E}_1^{-1}(Z_1) = \cdots = \bar{E}_n^{-1}(Z_n)$ and $Y = g^{R_\ell s}$ for some $1 \leq \ell \leq n$. Then

$$\begin{aligned} 1/n + \epsilon(k) &\leq \Pr[\text{V solves Problem A}] \\ &\leq \Pr[\mathbf{E}]\Pr[\text{V solves Problem A}|\mathbf{E}] + \Pr[\bar{\mathbf{E}}]\Pr[\text{V solves Problem A}|\bar{\mathbf{E}}] \\ &\leq \eta(k) \cdot 1 + (1 - \eta(k)) \cdot (1/n) = (1/n) + (1 - 1/n)\eta(k) \end{aligned}$$

where $\eta(k) = \Pr[\mathbf{E}]$. In the event $\mathbf{E}$, which has non-negligible probability $\eta(k) \geq n/(n-1) \cdot \epsilon(k)$, $\mathcal{M}$ obtains its DLP answer. This contradicts Theorem assumptions.

Remark: To see why V cannot do better than random guess in the event $\bar{\mathbf{E}}$, assume V mysteriously obtains a value X such that $\beta = g^X$. Notice that for each R_i, $1 \leq i \leq n$, there is a value s_i such that $X = R_i \cdot s_i$. In the event $\bar{\mathbf{E}}$, the outcome of the following n queries are yet to be generated by $\mathcal{M}$'s random tape: $H_1(\bar{E}_1(s_i) || \cdots || \bar{E}_n(s_i))$, $1 \leq i \leq n$. These outcomes are not yet generated by the time V returns its output to $\mathcal{M}$, and thus V essentially cannot do better than random guess, even if it mysteriously knows the discrete logarithm of β. The detailed proof is technical and omitted.

(*V colluding with diversion receivers*) By similar approach, we can see that V' cannot do better than the probability of $1/n + \Delta(k)$ to compute the identity even if all the corresponding decryption functions D_i, $1 \leq i \leq n$, are known.

Problem B is equivalent to Problem *B'* below.

Problem B' : Given ℓ such that $\pi = \phi(\ell)$, $E_1(r_1)||\cdots||E_n(r_n)$, $g^{H_2(r_{\phi(1)})}$, $\cdots$, $g^{H_2(r_{\phi(n)})}$, s, and $g^{H_2(r_\pi)s}$, compute π. Note $\phi \in \mathrm{Sym}(n)$ and $r_1, \cdots, r_n$ are unspecified.

Lemma 2. *Suppose a PPT V', after making q_{H_2} queries of H_2, computes Problem B' with probability $1/n + \epsilon(k)$. There exists a PPT $\mathcal{M}$ which invokes V' and answers all H_2-queries, can invert one of the asymmetric encryption functions $E_1, \cdots, E_n$ with probability at least $\frac{n}{n-1} \cdot \frac{1}{q_{H_2}} \cdot \epsilon(k)$.*

To compute at least one of the asymmetric inversions $E_1^{-1}(Z_1), \cdots, E_n^{-1}(Z_n)$, $\mathcal{M}$ randomly generates $\phi \in_R \mathrm{Sym}(n)$, π, s, $Y_1, \cdots, Y_n$, and invokes V' with inputs ℓ such that $\pi = \phi(\ell)$, $Z_1||\cdots||Z_n$, $g^{Y_{\phi(1)}}, \cdots, g^{Y_{\phi(n)}}$, s, and $g^{Y_\pi s}$.

Let $\mathbf{E}$ denote the event that V' queries H_2 with z satisfying $Z_i \leftarrow E_i(z)$ for some i, $1 \le i \le n$. Let $\Pr\{\mathbf{E}\} = \eta(k)$. Then

$$\begin{aligned} 1/n + \epsilon(k) &\le \Pr\{V' \text{ solves}\} \\ &= \Pr\{\mathbf{E}\}\Pr\{V' \text{ solves } |\mathbf{E}\} + \Pr\{\bar{\mathbf{E}}\}\Pr\{V' \text{ solves } |\bar{\mathbf{E}}\} \\ &\le \eta(k)\cdot 1 + (1-\eta(k))(1/n) = (1/n) + (1-1/n)\eta(k) \end{aligned}$$

Note in the event of $\bar{\mathbf{E}}$, V' essentially cannot do better than random guess even if he knows all values $r_1, \cdots, r_n$ because the query outcomes $H_2(r_{\phi(1)}), \cdots H_2(r_{\phi(n)})$ are yet to be randomly generated by $\mathcal{M}$'s random tape by the time V' completes.

Note that $\mathcal{M}$ has to identify the occurence of event . This may be accomplished to test any given query z of H_2 such that $Z_i = E_i(z)$, for some i, $1 \le i \le n$. However, $\mathcal{M}$ may not be able to do this if a probabilistic public-key encryption function E_i's random tape for generating Z_i is unknown. $\mathcal{M}$ has to randomly pick one query out of q_{H_2} H_2-queries and hopes it is the right moment of event $\mathbf{E}$. Therefore, $\mathcal{M}$ succeeds in inverting at least one of the asymmetric encryption function with probability at least $\frac{n\epsilon(k)}{(n-1)q_{H_2}}$. This contradicts the Theorem assumption that each E_i is secure against PPT adversaries.

(*V colluding with diversion receivers*) Modify Problem *B'* such that t $(t < n)$ corresponding decryption functions excluding D_π are also given, we can construct for contradiction another reduction master to invert one of the unknown asymmetric encryption functions if there exists a PPT which can solve the modified problem with probability at least $1/(n-t) + \epsilon(k)$. The detailed proof is omitted.

Problem C: We re-iterate Problem C below. We assume V has t colluders, denoted $E_1^{-1}, \cdots, E_t^{-1}$ without loss of generality.

Problem C : Given $\lambda_i = E_i(r_i)$, $1 \le i \le n$, $\gamma = g^{H_2(r_{\phi(1)})}||\cdots||g^{H_2(r_{\phi(n)})}$, $\alpha' = E_1(s)||\cdots||E_n(s)$, $s' = H_2(r_\pi)s + m$, $d = g^m$, $\theta = H_1(\lambda||\gamma||H_1(\alpha')||g^{s'}/d)$, $E_1, \cdots, E_n, E_1^{-1}, \cdots, E_t^{-1}$, H_1, H_2, and $t < \pi \le n$, compute π. Note $\phi \in \mathrm{Sym}(n)$, s, and $r_i, \cdots, r_n$ are unspecified.

Problem C is reducible to inverting at least one of the asymmetric encryptions $E_{t+1}, \cdots, E_n$, under the random oracle model. We use a special form of the random oracle.

(*A formulation of the random oracle with n permutable back patches*) For asymmetric encryption functions E_1, $\cdots$, E_n, let $\mathcal{H}(t, E_1, \cdots, E_n)$ denote the random oracle which generates its outputs in the following way:

1. $\mathcal{H}$ randomly generates n distinct values Z_1, $\cdots$, $Z_n \in \{1, \cdots, q\}$.
2. $\mathcal{H}$ randomly generates n distinct values Y_1, $\cdots$, $Y_n \in \{1, \cdots, q\}$.
3. Let X be a query to $\mathcal{H}$. If X has never been queried before, $\mathcal{H}$ checks if $E_i(X) = Z_i$ for some i, $1 \leq i \leq n$. Case yes and $1 \leq i \leq t$, set $\mathcal{H}(X) = Y_i$. Case yes and $t < i \leq n$, randomly select a member Y from $\{Y_{t+1}, \cdots, Y_n\}$ which has never been selected before and set $\mathcal{H}(X) = Y$. Case no, set $\mathcal{H}(X)$ to a random member of $\{1, \cdots, q\} \setminus \{Y_1, \cdots, Y_n\}$ which has never been outputted by $\mathcal{H}$ before. On duplicated queries X, $\mathcal{H}$ maintains consistency with previous outputs.

Remark: The above remains a random oracle.

Lemma 3. *Suppose a PPT algorithm V, knowing the decryption functions E_1^{-1}, $\cdots$, E_t^{-1}, solves Problem C with probability at least $1/(n-t)+\epsilon(k)$. There exists a PPT algorithm, $\mathcal{M}$, which invokes V and simulates the view of V by replacing its queries to H_2 by queries to $\mathcal{H}(t, E_1, \cdots, E_n)$, can compute (the discrete logarithm of) at least one of (distinct) $E_{t+1}^{-1}(Z_{t+1})$, $\cdots$, $E_n^{-1}(Z_n)$, with probability at least $\frac{n-t}{n-t-1}\epsilon(k)$. The complexity of $\mathcal{M}$ is in the same order as that of V.*

In the following we abbreviate $\mathcal{H} = \mathcal{H}(t, E_1, \cdots, E_n)$. Assume PPT algorithm V solves Problem C with probability $1/(n-t)+\epsilon(k)$. $\mathcal{M}$ randomly generates, on behalf of $\mathcal{H}$, the values Z_1, $\cdots$, Z_n, Y_1, $\cdots$, Y_n. Further, $\mathcal{M}$ randomly generates m, s, and ℓ with $t \leq \ell \leq n$. Then $\mathcal{M}$ invokes $\mathcal{V}$ with inputs $\lambda_1 = Z_1$, $\cdots$, $\lambda_n = Z_n$, $\gamma = g^{Y_1}||\cdots||g^{Y_n}$, $\alpha' = E_1(s)||\cdots||E_n(s)$, $s' = Y_\ell s + m$, $d = g^m$, $\theta = H_1(\lambda||\gamma||H_1(\alpha')||g^{s'}/d)$. As $\mathcal{V}$ executes, $\mathcal{M}$ records its queries to $\mathcal{H}$, $\mathcal{M}$ simulates $\mathcal{H}$ in answering queries including flips coins, and $\mathcal{M}$ checks the eventual output from $\mathcal{V}$. Note $\mathcal{M}$ can simulate $\mathcal{H}$ in polynomial time.

Let $\bar{\mathbf{E}}$ denote the event $\mathcal{V}$ does not query $\mathcal{H}$ with any input X satisfying $E_i(X) = Z_i$ for any i with $t < i \leq n$. In the event $\mathbf{E}$, V queries $\mathcal{H}$ with an input X satisfying $H_i(X) = Z_i$ for some i, $t < i \leq n$. Then $\mathcal{M}$ has obtained the desired asymmetric decryption $X = E_i^{-1}(Z_i)$. In the event $\bar{\mathbf{E}}$, the correspondence between the Y_i's and the Z_i's in the third step of $\mathcal{H}$ specification has not yet been decided by the time $\mathcal{V}$ completes. The value of π such that there exists X with $E_\pi(X) = Z_\pi$ and $\mathcal{H}(X) = Y_\ell$ has yet to be decided with at least $n-t$ equally probable candidates $\pi \in \{t+1, \cdots, n\}$. $\mathcal{M}$ will have to flip additional coins in order to choose π among the candidates. Therefore

$$\begin{aligned}\frac{1}{n-t} + \epsilon(k) &\leq \Pr\{V \text{ succeeds}\} \\ &= \Pr\{\mathbf{E}\}\Pr\{V \text{ succeeds}|\mathbf{E}\} + \Pr\{\bar{\mathbf{E}}\}\Pr\{V \text{ succeeds}|\bar{\mathbf{E}}\} \\ &\leq \eta \cdot 1 + (1-\eta)\frac{1}{n-t}\end{aligned}$$

where $\eta = \Pr\{\mathbf{E}\}$. $\mathcal{M}$'s probability of success is at least η and $\eta \geq \frac{n-t}{n-t-1}\epsilon(k)$.

Problem D: For simplicity, we prove for the scenario where there are two rounds with Case b=3. Other scenarios are similar. Problem D is reiterated below, where V has t colluders denoted $E_1^{-1}, \cdots, E_t^{-1}$ without loss of generality.

Problem D: Given g^m, $E_1, \cdots, E_n$, $\lambda = E_1(r_1)||\cdots||E_n(r_n)$, $\bar{\lambda} = E_1(\bar{r}_1)||\cdots||E_n(\bar{r}_n)$, $\gamma = g^{H_2(r_{\phi(1)})} \;||\; \cdots \;||\; g^{H_2(r_{\phi(n)})}$, $\bar{\gamma} = g^{H_2(\bar{r}_{\bar{\phi}(1)})}||\cdots||g^{H_2(\bar{r}_{\bar{\phi}(n)})}$, $\alpha' = E_1(s)||\cdots||E_n(s)$, $\bar{\alpha}' = E_1(\bar{s})||\cdots||E_n(\bar{s})$, $s' = H_2(r_\pi)s + m$, $\bar{s}' = H_2(\bar{r}_\pi)\bar{s} + m$, $\theta = H_1(\lambda||\gamma||H_1(\alpha')||g^{s'}/d)$, $\bar{\theta} = H_1(\bar{\lambda}||\bar{\gamma}||H_1(\bar{\alpha}')||g^{\bar{s}'}/d)$, and $E_1^{-1}, \cdots, E_t^{-1}$, compute π, where $t < \pi \leq n$.

Problem D is reducible to inverting at least one of the asymmetric encryptions $E_{t+1}, \cdots, E_n$, under the random oracle model. Like Problem C, We use a special form of the random oracle.

(*A formulation of the random oracle in terms of* $2n$ *relations*) For asymmetric encryption functions $E_1, \cdots, E_n$, let $\mathcal{H}_D(t, E_1, \cdots, E_n)$ denote the random oracle which generates its outputs in the following way:

1. $\mathcal{H}_D$ randomly generates $2n$ distinct values $Z_1, \cdots, Z_n, \bar{Z}_1, \cdots, \bar{Z}_n$ such that $Z_i, \bar{Z}_i$ are in the range of E_i, for all i, $1 \leq i \leq n$.
2. $\mathcal{H}_D$ randomly generates $2n$ distinct values $Y_1, \cdots, Y_n, \bar{Y}_1, \cdots, \bar{Y}_n \in_R \mathbb{Z}_q$.
3. Let X be a query to $\mathcal{H}_D$. If X has never been queried before, $\mathcal{H}_D$ checks if $E_i(X) = Z_i$ for some i, $1 \leq i \leq n$. Case yes and $1 \leq i \leq t$, set $\mathcal{H}_D(X) = Y_i$. Case yes and $t < i \leq n$, randomly select a member Y from $\{Y_{t+1}, \cdots, Y_n\}$ which has never been selected before and set $\mathcal{H}_D(X) = Y$. Case no, set $\mathcal{H}_D(X)$ to a random member of $\{1, \cdots, q\} \setminus \{Y_1, \cdots, Y_n\}$ which has never been outputted by $\mathcal{H}_D$ before. Otherwise : $\mathcal{H}_D$ checks if $E_i(X) = \bar{Z}_i$ for some i, $1 \leq i \leq n$. If yes and $1 \leq i \leq t$, set $\mathcal{H}_D(X) = \bar{Y}_i$. If yes and $t < i \leq n$, randomly select a member Y from $\{\bar{Y}_{t+1}, \cdots, \bar{Y}_n\}$ which has never been selected before and set $\mathcal{H}_D(X) = Y$. If no, set $\mathcal{H}_D(X)$ to a random member of $\{1, \cdots, q\} \setminus \{Y_1, \cdots, Y_n, \bar{Y}_1, \cdots, \bar{Y}_n\}$ which has never been outputted by $\mathcal{H}_D$ before. On duplicated queries X, $\mathcal{H}_D$ maintains consistency with previous outputs.

Remark: The above remains a random oracle.

Lemma 4. *Suppose a PPT algorithm* V*, knowing the decryption functions* E_1^{-1}*,* $\cdots$*,* E_t^{-1}*, solves Problem D with probability at least* $1/(n-t) + \epsilon(k)$*. There exists a PPT algorithm,* $\mathcal{M}$*, which invokes* V *and simulates the view of* V *by replacing its queries to* H_2 *by queries to* $\mathcal{H}_D(t, E_1, \cdots, E_n)$*, can compute (the discrete logarithm of) at least one of (distinct)* $E_{t+1}^{-1}(Z_{t+1})$*,* $\cdots$*,* $E_n^{-1}(Z_n)$*, with probability at least* $\frac{n-t}{n-t-1}\epsilon(k)$*. The complexity of* $\mathcal{M}$ *is in the same order as that of* V*.*

In the following we abbreviate $\mathcal{H}_D = \mathcal{H}_D(t, E_1, \cdots, E_n)$. Assume PPT algorithm V solves Problem D with probability $1/(n-t) + \epsilon(k)$. $\mathcal{M}$ randomly generates, on behalf of $\mathcal{H}_D$, the values $Z_1, \cdots, Z_n, Y_1, \cdots, Y_n$. and $\bar{Z}_1, \cdots, \bar{Z}_n$, $\bar{Y}_1, \cdots, \bar{Y}_n$. Further, $\mathcal{M}$ randomly generates m, s, $\bar{s}$, ℓ with $t \leq \ell \leq n$. $\bar{\ell}$ with $t \leq \bar{\ell} \leq n$. Then $\mathcal{M}$ invokes $\mathcal{V}$ with inputs g^m, $E_1, \cdots, E_n$, $\lambda = Z_1||\cdots||Z_n$, $\bar{\lambda} = \bar{Z}_1||\cdots||\bar{Z}_n$, $\gamma = g^{Y_1}||\cdots||g^{Y_n}$, $\bar{\gamma} = g^{\bar{Y}_1}||\cdots||g^{\bar{Y}_n}$, $\alpha' = E_1(s)||\cdots||E_n(s)$,

$\bar{\alpha}' = E_1(\bar{s})||\cdots||E_n(\bar{s})$, $s' = Y_\ell s + m$, $\bar{s}' = \bar{Y}_{\bar{\ell}}\bar{s} + m$, $\theta = H_1(\lambda||\gamma||H_1(\alpha')||g^{s'}/d)$, $\bar{\theta} = H_1(\bar{\lambda}||\bar{\gamma}||H_1(\bar{\alpha}')||g^{\bar{s}'}/d)$, E_1^{-1}, $\cdots$, E_t^{-1}.

The rest of the proof is similar to that of the previous Lemma and omitted.

Problem E: This problem can be re-iterated as a collection of *Problem E(i)*, $1 \le i \le 3^N$:

$$\{\text{Find } \pi \text{ from } \mathcal{T}_i : m \leftarrow F_q; \mathcal{T}_i \leftarrow (i, P)\}$$

where P is a honest prover defined in Sec. 4 and $\mathcal{T}_i$ is a particular transcript formally specified as follows.

(Transcripts $\mathcal{T}_i$) Let $i = (b_1, \cdots, b_N)$ in ternary notation. Let

$$\{1, \cdots, N\} = \mathbf{A} \cup \mathbf{C} \cup \mathbf{F} = \{a_1, \cdots, a_{N_1}\} \cup \{c_1, \cdots, c_{N_2}\} \cup \{f_1, \cdots, f_{N_3}\}$$

where $N = N_1 + N_2 + N_3$ and $\mathbf{A}$, $\mathbf{C}$ and $\mathbf{F}$ are the sets of indices corresponding to Rounds with $b = 1, 2, 3$, respectively.

$$\begin{aligned}
\mathcal{T}_i \leftarrow P : \{&(E_1, \cdots, E_n) \leftarrow \mathcal{G}(1^k);\\
&d; \theta^{(i)}, 1 \le i \le N; (r_1^{(i)}, \cdots, r_n^{(i)}), i \in \mathbf{A};\\
&\gamma^{(i)}, 1 \le i \le n; \alpha^{(i)}, i \in \mathbf{A}; \beta^{(i)}, i \in \mathbf{A};\\
&\lambda^{(i)}, i \in \mathbf{C} \cup \mathbf{F}; s^{(i)}, i \in \mathbf{C}; \alpha'^{(i)}, i \in \mathbf{F}; s'^{(i)}, i \in \mathbf{F}\}
\end{aligned}$$

(*Problem E*) Solve Problem $E(1)$, $\cdots$, or $E(3^N)$.

Lemma 5. *Suppose a PPT algorithm V, with H_1 and H_2 being random oracles, achieves*

$$\max_{1 \le i \le 3^N} \Pr[V \text{ solves problem } E(i)] \ge \frac{1}{n} + \epsilon(k)$$

for some non-negligible function ϵ. There exists a PPT algorithm $\mathcal{M}$, which invokes V and simulates the view of V by answering all its H_1-queries and H_2-queries, can solve at least one of the following problems: Discrete Logarithm Problem or inverting a secure public-key encryption function, with probability at least $(n-1)/n \cdot \epsilon(k)$.

We construct $\mathcal{M}$ as follows.

$\mathcal{M}$ = "On
1. inputs $\{Y^{(i)} \in G : i \in \mathbf{A}\}$, for each $a \in \mathbf{A}$,
   (a) Randomly pick $r_1^{(a)}, \cdots, r_n^{(a)} \in_R \{0,1\}^k$ and $R_1^{(a)}, \cdots, R_n^{(a)} \in_R F_q$. Set the values of $H_2(r_i) = R_i^{(a)}$, for all $1 \le i \le n$.
   (b) Randomly pick $\alpha^{(a)} \in_R \{0,1\}^k$.
   (c) Set $\beta^{(a)} = Y^{(a)}$ and $\gamma^{(a)} = (g^{R_1^{(a)}}, \cdots, g^{R_n^{(a)}})$
   (d) Generate n secure public-key encryption functions $\bar{E}_1^{(a)}, \cdots, \bar{E}_n^{(a)}$ at random (whose decryption functions are generated by, and known to, $\mathcal{M}$).

(e) Run V on corresponding inputs and reply all the queries of H_1 and H_2 in the following manner.
   – For any H_2-query with input $r_i^{(a)}$, $1 \leq i \leq n$, $R_i^{(a)}$ is replied.
   – For a H_1-query with input $Z_1^{(a)}||\cdots||Z_n^{(a)}$, compute the inversion $s_i'^{(a)} \leftarrow \bar{E}_i^{-1(a)}(Z_i^{(a)})$, $1 \leq i \leq n$, and determine if $s_1'^{(a)} = \cdots = s_n'^{(a)}$ and $Y^{(a)} = g^{R_\ell^{(a)} s_1'^{(a)}}$, for some $1 \leq \ell \leq n$. If they are true, output $R_\ell^{(a)} s_1'^{(a)}$ as the discrete logarithm of $Y^{(a)}$ and halt. Otherwise, randomly pick an element from $\{0,1\}^k \setminus \{\alpha^{(a)}\}$ as the reply.
   – For any other queries of H_1 and H_2, random numbers are generated in the corresponding range of H_1 and H_2 as the replies.
   – For query consistency, for any query with an input value which has been received before, the same reply as the last time is returned.

2. and inputs $(Z_1^{(a)}, \cdots, Z_n^{(a)})$, $a \in \mathbf{C}$", where each $Z_i^{(a)}$ is in the range of a secure public-key encryption function E_i, $1 \leq i \leq$, for each $a \in \mathbf{C}$, $\mathcal{M}$ sets $\lambda^{(a)} = Z_1^{(a)}||\cdots||Z_n^{(a)}$ and randomly generates $Y_i^{(a)} \in_R G$, $1 \leq i \leq n$ and $s^{(a)} \leftarrow F_q$, and set $\gamma^{(a)} = (Y_1^{(a)}, \cdots, Y_n^{(a)})$. Then $\mathcal{M}$ runs V on corresponding inputs and reply all the queries of H_1 and H_2 in the similar manner to the above. Besides ensuring randomness in replies and maintaining query consistency, $\mathcal{M}$ evaluates the input value of each H_2-query, denoted by $r^{(a)}$, and determine if $Z_i^{(a)} = E_i(r^{(a)})$, for some i, $1 \leq i \leq n$. If this is the case, $\mathcal{M}$ outputs $r^{(a)}$ as the plaintext of $Z_i^{(a)}$ and halts;
3. and inputs $(Z_1^{(a)}, \cdots, Z_n^{(a)})$, $a \in \mathbf{F}$", where each $Z_i^{(a)}$ is in the range of a secure public-key encryption function E_i, $1 \leq i \leq$, for each $a \in \mathbf{F}$, $\mathcal{M}$ prepares the appropriate inputs for V and invokes V and answering all the queries of H_1 and H_2 in the similar manner to Step 2 above.
4. Halt with no output if V stops."

If V does not make any queries to H_1 or to H_2 in any of the N rounds, then he can best randomly guess π because $\mathcal{M}$ has not decided on the value of π yet. If V makes any "qualified" query, then $\mathcal{M}$ succeeds.

Remark on Probabilistic Public-key Encryption Functions : In the formulation of $\mathcal{H}$ and $\mathcal{H}_D$ above, the functions are required to check if $E_i(X)$ is computed to Z_i for some i, $1 \leq i \leq n$. In general, this may not be feasible if E_i is probabilistic while the corresponding encryption coin flip sequence as well as E_i^{-1} are unknown. In our scheme described in Sec. 4.1, the corresponding coin flip sequence of E_i for yielding Z_i from X is also given. One may also consider the coin flip sequence to be part of the message X. For the case when coin flip sequences are not carried over to places where encryptions are not required, for example, computing $g^{H_2(r_{\phi(i)})}$, we can use the technique of the proof of Lemma 2 instead.

Therefore for simplicity, we assume that all the underlying public key encryption functions used in most parts of our proof are deterministic.

Proof of Confidentiality:

If a PPT adversary V can break Confidentiality, then it must be able to break Confidentiality in at least one of the following five scenarios:

A. V can break Confidentiality in an individual Round with $b = 1$.
B. V can break Confidentiality in an individual Round with $b = 2$.
C. V can break Confidentiality in an individual Round with $b = 3$.
D. V can break Confidentiality in multiple Rounds all with $b = 3$.
E. V can break Confidentiality based on transcripts from all N Rounds.

Problem A: If V can compute m from inputs $d = g^m$, $\theta = H_1(\lambda||\gamma||\alpha||\beta)$ and $r_1, \cdots, r_n, \gamma, \alpha, \beta$, then V can compute the discrete logarithm of d because other parameters are all unrelated to m.

Problem B: If V can compute m from inputs $d = g^m$, $\theta = H_1(\lambda||\gamma||\alpha||\beta$ and λ, γ, and s, then V can compute the discrete logarithm of d because other parameters are all unrelated to m.

Problem C: We re-iterate Problem C below.

Problem C': Given $d, \theta, \lambda, \alpha', s', E_1, \cdots, E_n, H_1, H_2, s, \pi, \phi \in \mathrm{Sym}(n)$ where $d = g^m$, $\theta = H_1(\lambda||\gamma||\alpha||\beta)$, $\lambda = E_1(r_1)||\cdots||E_n(r_n)$, $\alpha' = E_1(s)||\cdots||E_n(s)$, $\gamma = g^{H_2(r_{\phi(1)})}||\cdots||g^{H_2(r_{\phi(n)})}$, $s' = H_2(r_\pi)s + m$, for some $m, r_1, \cdots, r_n, E_1^{-1}, \cdots, E_t^{-1}$, $t < \pi \leq n$, compute m.

Problem C', is strictly easier than Problem C: solving Problem C while the following parameters are given as additional inputs: s, π, ϕ.

Now a classic problem:

Simple Proof of Knowledge (SPoK):

$$\{\text{Compute } x \text{ from } (\sigma, y, \alpha) : x, r \leftarrow F_q; y = g^x, \alpha = g^r, \sigma = x + r\}$$

This is computationally equivalent to finding the private key in Schnorr's Identification Scheme [12] and is also equivalent to the discrete logarithm problem (DLP). To see this, suppose there is an algorithm Oracle^{SPoK} which accepts on inputs y, α, σ described in *SPoK* above and outputs x such that $y = g^x$ (assume that the domain parameters are publicly known). Then we can apply the crooked attacking technique [12] to build the following algorithm DLPSolver to solve DLP.

```
DLPSolver = "On input $y$,
    1. Random pick $r \leftarrow F_q$ and set $\sigma = r$.
    2. Compute $\alpha = g^r y^{-1}$.
    3. Output $x = \mathsf{Oracle}^{SPoK}(y, \alpha, \sigma)$."
```

Mutual reduction between SPoK and Problem C with $t \geq 1$:

Lemma 6. *SPoK is computationally equivalent to Problem C'. Therefore suppose a PPT algorithm solves Problem C with non-negligible probability, then there exists a PPT algorithm that solves SPoK with non-negligible probability. In other words, we have SPoK $\equiv_P$ Problem $C' \leq_P$ Problem C.*

Sketch of Proof: The following correspondence does it:

$$x = m, y = d, r = H_2(r_\pi)s, \sigma = s'$$

Remark: Our proof of Confidentiality in Theorem 1 does no use the random oracle model. Our proof of Theorem 1, Confidentiality, Per-Round Unique Decypherability, Public Verifiability, and Spontaneity does not use the random oracle model. Only the proof of Signer-Ambiguity (Anonymity) depends on the random oracle model.

Problem D: Problem D is mutually reducible with SPoK with multiple witnesses. Proof omitted.

Simple Proof of Knowledge (SPoK) with multiple witnesses:

$$\{\text{Compute } x \text{ from } (y, \sigma_1, \alpha_1, \cdots, \sigma_\ell, \alpha_\ell) : x, r_1, \cdots, r_\ell \leftarrow F_q;$$
$$y = g^x, \alpha = g^{r_1}, \sigma = x + r_1, \cdots, \alpha = g^{r_\ell}, \sigma = x + r_\ell\}$$

Problem E: All rounds with b=1 or 2 do not involve m. Therefore solving Problem E is reduced to solving Problem D.

Public Verifiability and Spontaneity:

These properties are implied directly from the scheme described in Sec. 4.