# Online/Offline Signatures and Multisignatures for AODV and DSR Routing Security

Shidi Xu[1], Yi Mu[1], Willy Susilo[1], Xiaofeng Chen[2], Xinyi Huang[1], Fangguo Zhang[2]

[1]School of Information Technology and Computer Science
University of Wollongong, Australia
{sdx86, wsusilo, ymu}@uow.edu.au

[2]Department of Computer Science
Sun Yat-sen University,P.R.China
{isszhfg,isschxf}@mail.sysu.edu.cn

**Abstract.** Efficient authentication is one of important security requirements in mobile ad hoc network (MANET) routing systems. The techniques of digital signatures are generally considered as the best candidates to achieve strong authentication. However, using normal digital signature schemes is too costly to MANET due to the computation overheads. Considering the feasibility of incorporating digital signatures in MANET, we incorporate the notion of online/offline signatures, where the computational overhead is shifted to the offline phase. However, due to the diversity of different routing protocols, a universal scheme that suits all MANET routing systems does not exist in the literature. Notably, an authentication scheme for the AODV routing is believed to be not suitable to the DSR routing. In this paper, we first introduce an efficient ID-based online/offline scheme for authentication in AODV and then provide a formal transformation to convert the scheme to an ID-based online/offline multisignature scheme. Our scheme is *unique*, in the sense that a single ID-based online/offline signature scheme can be applied to both AODV and DSR routing protocols. We provide the generic construction as well as the concrete schemes to show an instantiation of the generic transformation. We also provide security proofs for our schemes based on the random oracle model. Finally, we provide an application of our schemes in the dynamic source routing protocol.

**Keywords:** MANET, AODV, DSR, Authentication, Digital signature, Online/offline signature, Multisignature

## 1 Introduction

The security technology deployed in the existing mobile ad hoc networks (MANET) is very weak [10]. Several well-known MANET routing protocols such as DSR [7] and AODV [11] were designed without a security consideration. Consequently, MANET routing systems face a number of security threats, from basic spoofing attacks to more complex rushing attacks. Providing full-scale security to MANET with a low computational overhead and bandwidth consumption becomes an open problem.

The security deployment to MANET is stunted by cryptographic techniques. The nature of the network requires low computational overheads, whereas existing authentication methods, such as digital signatures, are too expensive to apply. In [16], an ID-based online/offline signature scheme was proposed to provide a solution to this problem. In the online/offline notion, the computation overhead for a signing operation is shifted, so that the online computation can be very efficient. However, this scheme is not universally applicable to all the MANET routing protocols in the sense of achieving the best efficiency. We found that the online/offline signature scheme for AODV routing protocol *does not* help for securing DSR, because of their difference in packet processing operations. To date, constructing an authentication scheme that is applicable for both AODV and DSR is remaining an interesting open problem.

The online/offline digital signature scheme (IOS) was firstly introduced by Even, Goldreich and Micali [5]. The basic concept of their scheme is splitting the signature generation algorithm into two phases: offline phase and online phase. To achieve efficient performance when a message is to be signed, they utilized an offline phase to handle the most costly computation. When a message is ready, the online phase can be performed efficiently to generate the required signature.

Based on Even, Goldreich and Micali's scheme, Shamir and Tauman [14] utilizing the *hash-sign-switch* paradigm proposed an improved online/offline signature scheme. The online signing phase of their scheme maintains the efficiency of Even, Goldreich and Micali's scheme, requiring only one hash function. The new scheme is based on an ordinary digital signature scheme, in which the key size and signature size are largely reduced, compared with the original scheme.

The multisignature scheme was firstly introduced by Itakura, and Nakamura [6] in 1983. Multisignatures have been extensively studied in the literature, but due to the absence of a formal definition, there were many confusions caused.

In 2001, Micali et al. [9] provided the first formal definition of multisignature which is called Accountable Subgroup Multisignature (ASM). In essence, ASM schemes enable any subgroup, S, of a given group of potential signers, to efficiently sign a message so that the signature provably reveals the identities of the signers in the subgroup to any verifier. This scheme is based on Schnorr's signature scheme [13] therefore totally inherits the efficiency of Schnorr's signature.

**Our Contribution.** Motivated by creating a universal authentication scheme for MANET routing protocols, in this paper, we provide an affirmative answer to the open problem of constructing an authentication scheme for both AODV and DSR protocol. We firstly introduce an identity-based (or ID-based, for short) online/offline signature scheme suitable for AODV protocol and then transform this scheme to an ID-based multisignature scheme which is suitable for the DSR protocol. We provide a generic construction and the concrete constructions as instantiations of the generic construction and analyze the security and efficiency of the resulting scheme. Since the online/offline signature based authentication scheme for AODV protocol has been discussed in [16], in this paper, we only concentrate on providing an applicable authentication scheme for DSR protocol using above transformation.

**Organization of the paper.** The rest of the paper is organized as follow. In section 2, we define the notion of the ID-based online/offline signature schemes and the ID-based multisignature scheme. Then we give the generic construction from the ID-based online/offline signature scheme to the ID-based accountable subgroup multisignature scheme. In section 3, we present our concrete implementation of these signature schemes. We also prove the security and analyze the efficiency of the schemes. In section 4, we introduce the basics of DSR protocol and describe the application. Finally, we conclude the paper.

## 2 Generic Constructions

In this section, we firstly introduce the definition of bilinear pairing and GDH group. Then we review the definition of the ID-based online/offline signature scheme and accountable subgroup multisignature scheme and their security requirements. We also provide the generic construction of ID-based accountable subgroup multisignature scheme based on the ID-based online/offline signature scheme.

## 3 Cryptographic Tools: Bilinear Pairings

Let $\mathbb{G}_1$ be a cyclic additive group generated by $P$, with a prime order $q$, and $\mathbb{G}_2$ be a cyclic multiplicative group with the same prime order $p$. Let $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ be a map with with the following properties:

1. Bilinearity: $e(aP, bQ) = e(P, Q)^{ab}$ for all $P, Q \in \mathbb{G}_1, a, b \in \mathbb{Z}_q^*$;
2. Non-degeneracy: There exists $P, Q \in \mathbb{G}_1$ such that $e(P, Q) \neq 1$;
3. Computability: There is an efficient algorithm to compute $e(P, Q)$ for all $P, Q \in \mathbb{G}_1$;

The Non-degeneracy implies that when $P$ is the generator of $\mathbb{G}_1$, $e(P, P)$ is the generator of $\mathbb{G}_2$. We call such bilinear map as an admissible bilinear pairing. The problem considered in the additive group $\mathbb{G}_1$ is:

- **Computational Diffie-Hellman Problem (CDHP):** For $a, b \in \mathbb{Z}_q^*$, given $P, aP, bP$ compute $abP$.

In bilinear pairings, Decision Diffie-Hellman problem (DDHP) is easy and Computational Diffie-Hellman problem (CDHP) is still hard. That is, for $a, b \in \mathbb{Z}_q^*$, given $P, aP, bP$, computing $abP$ is infeasible.

**Definition 1.** *A group $\mathbb{G}$ is a gap Diffie-Hellman(GDH) if there exists a polynomial time probabilistic algorithm to compute the decisional Diffie-Hellman problem but exists no such algorithm to solve the computational Diffie-Hellman problem in $\mathbb{G}$.*

Above system parameters can be obtain through running the **GDH Parameter Generator** [4] $\mathcal{IG}$ which takes a security parameter $k \in \mathbb{Z}^+$ as input, runs in polynomial time in $k$, and outputs a prime number $q$, the description of two groups $\mathbb{G}_1, \mathbb{G}_2$ of order $q$, and the description of an admissible bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$.

**Definition 2.** *The advantage of an algorithm $\mathcal{A}$ in solving CDHP in group $\mathbb{G}$ is*

$$\mathsf{Adv}_{\mathcal{A}}^{CDH} = \Pr[\mathcal{A}(P, aP, bP) = abP : a, b \overset{R}{\leftarrow} \mathbb{Z}_q^*]$$

*where the probability is over the choice of $a$ and $b$, and the coin tosses of $\mathcal{A}$. We say that an algorithm $\mathcal{A}(t, \epsilon)$-breaks CDHP in $\mathbb{G}$ if $\mathcal{A}$ runs in time at most $t$, and $\mathsf{Adv}_{\mathcal{A}}^{CDH} > \epsilon$.*

## 3.1 ID-based Online/Offline Signature Scheme

**Definition 3.** *ID-based online/offline digital signature scheme $\mathcal{DS}$ is comprised of five polynomial time algorithms:* IO_ParamGen, IO_Ext, IO_OffSign, IO_OnSign, *and* IO_Verify.

IO_ParamGen. *The master key and parameter generation algorithm, is a probabilistic algorithm that on input a security parameter $1^k$, outputs a master key $IOSK^*$ and a parameter list params.*

IO_Ext. *The signing key issuing algorithm, is a deterministic algorithm that on input a user's identity id and a master key $IOSK^*$, returns a pair of matching public and secret keys $(iopk_{id}, iosk_{id})$.*

IO_OffSign. *The offline signing algorithm, is a probabilistiqlgorithm that on input a parameter list params and a signing key $iosk_{id}$, outputs an offline signature $S$.*

IO_OnSign. *The online signing algorithm, is a probabilistic algorithm that on input a message $m$ and an offline signature $S$, returns an online signature $\sigma$.*

IO_Verify. *The verification algorithm, is a deterministic algorithm that on input a message $m$, a user's identity id, a parameter list params, an offline signature $S$, and an online signature $\sigma$, returns 1 (*accept*) or 0 (*reject*).*

The security of the online/offline signature can be defined as followed.

**Definition 4.** *An identity based online/offline signature is said to be existentially unforgeable under chosen-message attacks if no probabilistic polynomial time adversary has a non-negligible advantage in this game:*

1. *The challenger $\mathcal{A}$ runs the setup algorithm to generate the system parameters and sends them to the adversary $\mathcal{F}$.*
2. *The adversary $\mathcal{F}$ performs the following queries:*
   - ***Key Extraction Query*** $\mathbf{O}^{\mathbf{IOS}}_{\mathbf{Ext}}$*: $\mathcal{F}$ produces an identity $ID$ and receives corresponding secret key $D_{ID}$.*
   - ***Offline Signing Query*** $\mathbf{O}^{\mathbf{IOS}}_{\mathbf{OffSign}}$*: $\mathcal{F}$ produces an identity $ID$, and receives an offline signature generated by offline signing oracle using the secret key corresponding to $ID$.*
   - ***Online Singing Query*** $\mathbf{O}^{\mathbf{IOS}}_{\mathbf{OnSign}}$*: $\mathcal{F}$ produces a message $m$, and receives a online signature generated by online signing oracle. The online signature is corresponding to the offline signature.*
3. *After a polynomial number of queries, $\mathcal{F}$ produces a tuple $(ID^*, m^*, S^*, \sigma^*)$ of identity $ID^*$, whose secret key was never asked in key extraction query. Besides, the pair $(ID^*, m^*)$ was never asked in online/offline signing queries.*

*The success probability of winning the above game is defined by $\mathsf{Succ}_{\mathcal{A}}^{EF-IOS-CMA}(\ell)$. An online/offline signature scheme is secure if the success probability of above attack is negligible.*

$$\mathsf{Succ}_{\mathcal{A}}^{EF-IOS-CMA}(\ell) \leq \epsilon,$$

*where $\epsilon$ is negligible.*

### 3.2 ID-based Accountable Subgroup Multisignature Scheme

Multisignature schemes, since firstly introduced by Itakura and Nakamura [6], have been extensively studied in the literature. However the first formal definition of multisignature scheme was provided by Micali et al. [9]. Their scheme, named Accountable Subgroup Multisignatures (ASM), enables any subgroup $G_{Sub}$ of a given group $G$ of potential signers, to sign a message efficiently, so that the signature provably reveals the identity of the signers in $G_{Sub}$ to any verifier.

According to Micali et al, we extend the definition of ASM to ID-based ASM.

**Definition 5.** *An ID-based accountable subgroup multisignature consists of four components. We assume that the total group $G_{Sub}$ consists of $L$ signers.*

AM_ParamGen. *The master key and parameter generation algorithm, is a probabilistic algorithm that on input a security parameter $1^k$, outputs a master key $AMSK^*$ and a parameter list $params$.*

AM_KeyGen. *The signing key issuing algorithm, is a probabilistic algorithm that on input a subgroup $G_{Sub}$, a user's identity id and a master key $AMSK^*$, returns a pair of matching public and secret keys $(ampk_{id}, amsk_{id})$ for each user in the group.*

AM_Signing. *The signing algorithm, is a probabilistic algorithm that on input the following from each signer:*

1. *a description of subgroup $G_{Sub}$*
2. *the public key $ampk_i$ of each member in $G_{Sub}$*
3. *the message $m$*
4. *the signer's secret key $amsk_i$*

*produces a signature $\sigma$ which is generated jointly by all the members of $G_{Sub}$.*

$\mathsf{AM\_Verifying}$. *The verification algorithm, is a deterministic algorithm, on input the following*

1. *a description of subgroup $G_{Sub}$*
2. *the public key $ampk_i$ of each member in $G_{Sub}$*
3. *the message $m$*
4. *the signature $\sigma$*

*outputs 1 (*accept*) or 0 (*reject*).*

The security definition of ID-based multisignature scheme can be adapted from the ID-based online/offline signature scheme.

**Definition 6.** *An ID-based multisignature (IBMS) of subgroup $S \subseteq G$ is said to be existentially unforgeable under chosen-message attacks if no probabilistic polynomial time adversary has a non-negligible advantage in producing a tuple $(\sigma, m, S)$ such that:*

1. *The challenger $\mathcal{A}$ runs the setup algorithm to generate the system parameters and sends them to the adversary $\mathcal{F}$.*
2. *The adversary $\mathcal{F}$ performs the following queries:*
   - ***Key Generation Query** $\mathbf{O^{AM}_{KGN}}$: $\mathcal{F}$ produces an identity $ID$ of the uncorrupted player in $S$ and receives corresponding secret key $D_{ID}$ and its temporary signing commitment $S$ for current signing session.*
   - ***Signing Query** $\mathbf{O^{AM}_{Sign}}$: $\mathcal{F}$ produces a message $m$, and receives a signature generated by signing oracle using the secret key corresponding to $ID$.*
3. *After a polynomial number of queries, $\mathcal{F}$ produces a tuple $(m^*, \sigma^*, S^*)$ such that*
   - *$\sigma^*$ is a valid signature on the message $m$ by the subgroup $S$ of players.*
   - *there exists an uncorrupted player $P^* \in S$ who has never been asked by $\mathcal{F}$ to execute the signing query on $m$ and $S$.*

*The success probability of winning the above game is defined by $\mathsf{Succ}_{\mathcal{A}}^{EF-IMS-CMA}(\ell)$. An ID based multisignature scheme is secure if the success probability of the above attack is negligible. In other words,*

$$\mathsf{Succ}_{\mathcal{A}}^{EF-IBMS-CMA}(\ell) \leq \epsilon,$$

*where $\epsilon$ is negligible.*

### 3.3 Generic Construction of IBMS from IOS

We observe the similarity between online/offline signature and multisignature: the offline signing phase does not involve any message in computation, therefore the resulting offline signature together with the signer's identity can be used as the public key for verifying online signature, which in turn can be treated as the signature in multisignature scheme. We provide the generic construction of multisignature scheme based on identity based online/offline signature scheme (Figure 1).

**Theorem 1.** *The ID-based multisignature scheme is secure only if the corresponding ID-based online/offline signature scheme is existentially unforgeable against chosen-message attacks.*

*Proof.* Suppose there is a polynomial time adversary $\mathcal{A}^{EF-IOS-CMA}$ who breaks the ID-based online/offline signature scheme. The ID-based accountable subgroup multisignature can be broken by running the same queries performed by $\mathcal{A}^{EF-IOS-CMA}$ with the help of same forger $\mathcal{F}^{EF-IOS-CMA}$:

```
IBMS_ParamGen ($1_k$)
    ($IOSK^*$, $params$) $\leftarrow$ IO_ParamGen($1^k$)
    $IBMSK^* \leftarrow IOSK^*$
    return ($IBMSK^*$, $params$)
IBMS_KeyGen ($G_{Sub}$, $id$, $AMP_{pub}$)
    ($iosk_{id}$, $iopk_{id}$) $\leftarrow$ IO_Ext(id, IBMSK$^*$, params)
    $ibmsk_{id} \leftarrow iosk_{id}$
    $ibmpk_{id} \leftarrow iopk_{id}$ return ($ibmpk_{id}$, $ibmsk_{id}$)
AM_Signing ($m$, $G_{Sub}$, $ibmsk_{id}$)
    $C_{id} \leftarrow$ IO_OffSign(id, iosk$_{id}$, params)
    $\sigma_{id} \leftarrow$ IO_OnSign(m, id, S, params, ibmsk$_{id}$)
    $\widetilde{\sigma} \leftarrow \Sigma^{G_{Sub}}(\sigma_{id})$
    $\widetilde{C} \leftarrow \Sigma^{G_{Sub}}(C_{id})$
    return ($\widetilde{\sigma}$, $\widetilde{C}$)
AM_Verifying ($m$, $G_{Sub}$, $\widetilde{\sigma}$, $\widetilde{C}$)
    $b \leftarrow$ IO_Verify(m, G$_{Sub}$, params, $\widetilde{\sigma}$, $\widetilde{C}$)
    return $b$
```

**Fig. 1.** Generic Construction from IOS to IBMS

1. The challenger $\mathcal{A}^{EF-IBMS-CMA}$ runs the IO_ParamGen algorithm to generate the system parameters and sends them to the forger $\mathcal{F}^{EF-IOS-CMA}$.
2. The adversary $\mathcal{F}^{EF-IOS-CMA}$ performs the following queries:
   - **Key Generation Query $\mathbf{O^{IBMS}_{KGN}}$:** $\mathcal{F}^{EF-IOS-CMA}$ provides an identity $ID$ of the uncorrupted player in $G_{Sub}$ to **Key Extraction Query $\mathbf{O^{IOS}_{Ext}}$** and **Offline Signing Query $\mathbf{O^{IOS}_{OffSign}}$** of ID-based online/offline signature. It receives corresponding secret key $D_{ID}$, and an offline signature as its temporary public key for current signing session.
   - **Signing Query $\mathbf{O^{IBMS}_{Sign}}$:** $\mathcal{F}^{EF-IOS-CMA}$ produces a message $m$, and receives a signature generated by **Online Singing Query $\mathbf{O^{IOS}_{OnSign}}$** of ID-based online/offline signature.
3. After a polynomial number of queries, $\mathcal{F}^{EF-IOS-CMA}$ produces a tuple $(m^*, \sigma^*, S^*)$ of identity $ID^*$, whose secret key was never asked in key extraction query and the pair $(ID^*, m^*)$ was never asked in online/offline signing queries. Obviously this resulting tuple satisfies the following:
   - $\sigma^*$ is a valid signature on the message $m$ by the subgroup $G_{Sub}$ of players.
   - there exists an uncorrupted player $P^* \in G_{Sub}$ who has never been asked by $\mathcal{F}^{EF-IOS-CMA}$ to execute the **Signing Query $\mathbf{O^{IOS}_{OnSign}}$** on $m$ and $G_{Sub}$.

$\square$

## 4 The Concrete Schemes

In this section, we provide a concrete ID-based online/offline signature scheme, and transform this scheme into the accountable subgroup multisignature scheme using the above general construction. We also prove the security of these two signature schemes.

### 4.1 Online/Offline Signature Scheme

Our scheme involves four algorithms: System Setup, ID Extract, Offline Signing, Online Signing and Verify.

Setup. Given $\mathbb{G}_1$ and its generator $P$, pick a random $s \in \mathbb{Z}_q^*$, and set $P_{pub} = sP$. Choose a cryptographic hash function $H_0 : \{0,1\}^* \rightarrow \mathbb{G}_1$ and $H_1 : \{0,1\}^* \times G_1 \rightarrow \mathbb{Z}_q^*$. The system parameters are $(P, P_{pub}, H_0, H_1)$. The master key is $s$. $H_0$ and $H_1$ behave as random oracles.

Extract. Given an identity $ID$, the algorithm computes $D_{ID} = sH_0(ID)$ and outputs it as the private key related to $ID$ corresponding to $Q_{ID} = H_0(ID)$.

OffSign. Given a secret key $D_{ID}$, pick random $r, x \in \mathbb{Z}_q^*$, output the offline signature pair $(S, R)$, where $S = D_{ID} - xP_{pub},\ \ R = rP$.

OnSign. Given a message $m$ and offline signature $S$, compute the online signature as $\sigma = H_1(m, R)r + x$. The resulting signature is a triple $(S, \sigma, R)$.

Verify. Given a signature tuple $(S, \sigma, R)$ of a message $m$ for an identity $ID$, check whether the following equation holds

$$e(S + \sigma P_{pub},\ P) = e(Q_{ID} + H_1(m, R)R,\ P_{pub})$$

The equation holds since:

$$\begin{aligned}
e(S + \sigma P_{pub},\ P) &= e(D - xP_{pub} + (H_1(m, R)r + x)P_{pub},\ P) \\
&= e(D - xP_{pub} + H_1(m, R)rP_{pub} + xP_{pub},\ P) \\
&= e(D + H_1(m, R)rP_{pub},\ P) \\
&= e(s(Q_{ID} + H_1(m, R)rP),\ P) \\
&= e(Q_{ID} + H_1(m, R)R,\ P_{pub})
\end{aligned}$$

Signing algorithms satisfy the requirement of online/offline signature as the actual message signing takes only one hash. The size of our signature is $2\log_2 \rho + \log_2 q$, in which $\rho$ stands for the safe length of GDH group $\mathbb{G}_1$.

## 4.2 Security Analysis

To prove our scheme is existentially unforgeable under adaptive chosen-message attack, we use Libert and Quisquater's proof technique [8].

**Theorem 2.** *In the random oracle model, if a probabilistic polynomial time forger $\mathcal{F}$ has an advantage $\varepsilon$ in forging an online/offline signature with running time t and asking $H_0$,$H_1$,key extraction oracle and online/offline signing oracle $q_{H_0}$, $q_{H_1}$, $q_e$ and $q_s$ times respectively, then the CDH problem can be solved with an advantage*

$$\varepsilon' > (\frac{1}{q_e} \cdot (1 - \frac{1}{q_e + 1})^{q_e+1})(\varepsilon - \frac{q_S(q_{H_1} + q_S) + 1}{2^k})$$

*with running time $t' < t + (q_{H_0} + q_e + 2q_s)t_m$, where $t_m$ is the time to compute a scalar multiplication in $\mathbb{G}_1$.*

*Proof.* Firstly we assume the existence of a forger $\mathcal{F}$, which by performing queries, finally produces a valid online/offline signature tuple. On the other hand, a probabilistic polynomial time algorithm - attacker $\mathcal{A}$ which answers all the queries asked by forger $\mathcal{F}$ finally solves the CDH problem. We further assume $(aP, bP) \in \mathbb{G}_1 \times \mathbb{G}_1$ is a random instance of the CDH problem taken as input by attacker $\mathcal{A}$. The system public key is initialized as $P_{pub} = aP$. Then $\mathcal{A}$ answers all the queries as followed:

**ID hash query:** when an identity ID is submitted to $H_0$ oracle, $\mathcal{A}$ flips a coin $T \in 0,1$ which yields 1 with probability $\delta$ and 0 with probability $1-\delta$. $\mathcal{A}$ then randomly chooses $u_i \in \mathbb{Z}_q^*$. If $T = 0$, $\mathcal{A}$ sets the value $Q_i$ as $u_i P$. Otherwise, $Q_i$ is set as $ubP$. $\mathcal{A}$ records the tuple $(ID_i, u_i, T_i)$ in a list $L_0$, and returns $Q_i$ as the answer.

**Key extraction query:** when $\mathcal{A}$ receives a key extraction query, it firstly checks whether the corresponding tuple $(ID_i, u_i, T_i)$ exists in $L_0$. If it does not exist, $\mathcal{A}$ outputs "failure" and halts. If it exists $\mathcal{A}$ further checks the value of $T_i$. If $T_i = 0$, it computes the secret key as $uP_{pub} = uaP$ and returns it to $\mathcal{F}$. Otherwise, it outputs "failure" and halts.

**Message hash query:** $\mathcal{A}$ maintains a list $L_1$ for message hash queries. When a tuple $(Q_i, m_i, R_i)$ is submitted to $H_1$ oracle, $\mathcal{A}$ firstly checks the existence of the tuple in $L_1$. If it exists, the value will be returned. Otherwise, $\mathcal{A}$ randomly chooses $v_i \in \mathbb{Z}_q^*$, stores the tuple $(Q_i, m_i, v_i, R_i)$, and returns $v_i$ to $\mathcal{F}$ as the answer.

**Offline signing query:** $\mathcal{A}$ randomly chooses $\alpha_i, t_i, \beta_i \in \mathbb{Z}_q^*$ and defines offline signature as $S_i = (t_i - \alpha_i)P_{pub} + R' = (t_i - \alpha_i)aP + \beta_i^2 P,\ R_i = \beta_i P$ and $R_i' = \beta_i^2 P$. The tuple $(S_i, R_i, R_i')$ and $\alpha_i$ are stored for future use.

**Online signing query:** when $\mathcal{A}$ receives an online signing query on message $M_i$ for an identity $ID_i$, it firstly retrieves the corresponding $u_i$ from $L_0$. The previously computed offline signature tuple $(S_i, R_i)$ and value $\alpha_i$ is also retrieved. Then it defines the message hash value $H_1(m_i, R_i) = \beta_i^{-1}(t_i - u_i)$, and the online signature as $\sigma_i = \alpha_i$. If the message hash value has been defined before, $\mathcal{A}$ output "failure" and halts. Otherwise, the signature tuple $(S_i, \sigma_i, R_i)$ is returned to $\mathcal{F}$.

The resulting signature tuple passes the verification since:

$$\begin{aligned} e(S_i^* + \sigma_i^* P_{pub}, P) &= e(t_i P_{pub} - \alpha_i P_{pub} + \alpha_i P_{pub}, P) \\ &= e(t_i aP, P) \end{aligned}$$

$$\begin{aligned} e(Q_i^* + H_1(m_i^*, R_i^*)R_i^*, P_{pub}) &= e(u_i P + (\beta_i^{-1}(t_i - u_i))\beta_i P, P_{pub}) \\ &= e(t_i P, aP) \end{aligned}$$

Eventually, the forger $\mathcal{F}$ produces a valid signature tuple $(S^*, \sigma^*, R^*)$ for message $M^*$ of identity $ID^*$ and gives it to $\mathcal{A}$. $\mathcal{A}$ firstly recovers the tuple $(ID^*, u^*, T^*)$ in list $L_0$ to check the value of $T$. if $T = 0$, $\mathcal{A}$ outputs "failure" and halts. Otherwise, the entry of $(Q^*, m^*, v_i^*, R_i^*)$ must be in the list $L_1$ with overwhelming probability. If this entry does not exist, $\mathcal{A}$ outputs "failure" and halts. As the resulting signature tuple is valid, the following equation holds:

$$e(S^* + \sigma^* P_{pub}, P) = e(Q_{ID}^* + H_1(m^*, R_i^*)R^*, P_{pub}) \tag{1}$$

Besides we have $H_1(m^*, R_i^*) = v_i$, $P_{pub} = aP$, and $Q_i = u_i bP$. According to (1) we can get:

$$\begin{aligned} e(S^* + \sigma^* aP, P) &= e(u_i bP + v_i R^*, aP) \\ e(S^* + \sigma^* aP, P) &= e(u_i bP, aP)e(v_i R^*, aP) \\ e(S^* + \sigma^* aP - v_i aR^*, P) &= e(u_i bP, aP) \end{aligned}$$

The solution to the CDH instance $(aP, bP)$ is $u_i^{-1}(S^* + \sigma^* aP - v_i aR^*)$.

Similar to Libert and Quisquater's analysis, $\mathcal{A}$'s probability of success involves three parts. Firstly, $\mathcal{A}$'s probability of failure caused by a conflict over $H_1$ is at most $q_S(q_{H_1} + q_S)/q$. Secondly, since $H_1$ is a random oracle, the probability of producing a valid forgery without asking $H_1(m^*)$

is $1/2^k$. Finally, the probability of $\mathcal{A}$ succeeds in a key extraction query is $\delta(1-\delta)^{q_e}$. The function $\delta(1-\delta)^{q_e}$ is maximized at $\delta = 1/(q_e+1)$. Thus we have the result

$$\begin{aligned}\delta(1-\delta)^{q_e} &= \frac{1}{q_e+1}\cdot(1-\frac{1}{q_e+1})^{q_e}\\ &= \frac{1}{q_e}\cdot(1-\frac{1}{q_e+1})^{q_e+1}\end{aligned}$$

Eventually it comes that $\mathcal{A}$'s advantages is at most

$$(\frac{1}{q_e}\cdot(1-\frac{1}{q_e+1})^{q_e+1})(\varepsilon - \frac{q_S(q_{H_1}+q_S)+1}{2^k})$$

□

## 4.3 ID-based Multisignature Scheme

We adapt our ID-based online/offline signature scheme to the ID-based multisignature scheme according to our generic construction. The resulting scheme consists of four algorithms: system setup, key generation, signing and verifying.

**Setup.** Given $\mathbb{G}_1$ and its generator $P$, pick a random $s \in \mathbb{Z}_q^*$, and set $P_{pub} = sP$. Choose a cryptographic hash function $H_0 : \{0,1\}^* \to \mathbb{G}_1$ and $H_1 : \{0,1\}^* \to \mathbb{Z}_q^*$. The system parameters are $(P, P_{pub}, H_0, H_1)$. The master key is $s$. $H_0$ and $H_1$ behave as random oracles.

**KeyGen.** For each player $P_i$ $(1 \le i \le L)$ in $G$, given an identity $ID_i$, the algorithm computes $D_{ID_i} = sH_0(ID_i)$ and outputs it as the private key related to $ID_i$ corresponding to $Q_i = H_0(ID_i)$.

**Signing.** Each player $P_i$ $(1 \le i \le L)$ in $G$ pre-computes the following:

1. randomly choose $x_i, r_i \in \mathbb{Z}_q^*$
2. compute the signing commitment for the current session as $C_i = D_i - x_iP_{pub}$, $R_i = r_iP$, and $U_i = x_iP$
3. broadcast $(C_i, R_i, U_i)$ to all the players.

Suppose the players in a subgroup $S = P_{i_1}, ..., P_{i_l}$ wish to jointly sign a message $m$. Upon receiving $(C_i, R_i)$ from all the other players, each of them does the following:

1. verify the received public key by checking the equality of the equation:

$$e(C_i, P) = e(Q_i - U_iP, P_{pub})$$

2. if the equality holds, compute

$$\widetilde{C} = \sum_{i=1}^{l} C_i = \sum_{i=1}^{l} D_i - \sum_{i=1}^{l} x_iP_{pub}$$

3. compute $\widetilde{R} = \sum_{i=1}^{l} R_i = \sum_{i=1}^{l} r_iP$.
4. compute the signature as
   (a) each signer computes the signature $\sigma_i = H_1(m)r_i + x_i$ and broadcasts to all the signer $P_{i_i}$ $(1 \le j \le l)$
   (b) upon receiving all the $\sigma_j$, each signer computes $\widetilde{\sigma} = \sum_{i=1}^{l} \sigma_i = H_1(m)\sum_{i=1}^{l} r_i + \sum_{i=1}^{l} x_i$.

The resulting multisignature for message $m$ is $(\widetilde{\sigma},\ \widetilde{C},\ \widetilde{R})$. To further reduce the signature size, we combine $\widetilde{\sigma}$ and $\widetilde{C}$ to obtain a new parameter $\widetilde{V}$ by

$$\widetilde{V} = \widetilde{C} + \widetilde{\sigma} P_{pub}$$

The final signature is a pair $(\widetilde{V},\ \widetilde{R})$.

**Verifying.** The multisignature can be verified by all the group members who possess the pair $(\widetilde{V}, \widetilde{R})$. Given signature $\widetilde{\sigma}$, commitment $\widetilde{R}$ and message $m$, check wether the following equation holds

$$e(\widetilde{V}, P) = e(\sum_{j=1}^{l} Q_j + H_1(m, \widetilde{R})\widetilde{R}, P_{pub})$$

The equation holds since

$$\begin{aligned} e(\widetilde{V},\ P) &= e(\widetilde{C} + \widetilde{\sigma} P_{pub},\ P) \\ &= e(\sum_{j=1}^{l} D_j - \sum_{j=1}^{l} x_j P_{pub} + (H_1(m, \widetilde{R}) \sum_{j=1}^{l} r_j + \sum_{j=1}^{l} x_j) P_{pub},\ P) \\ &= e(\sum_{j=1}^{l} D_j + H_1(m, \widetilde{R}) \sum_{j=1}^{l} r_j P_{pub},\ P) \\ &= e(s(\sum_{j=1}^{l} Q_j + H_1(m, \widetilde{R}) \sum_{j=1}^{l} r_j P),\ P) \\ &= e(\sum_{j=1}^{l} Q_j + H_1(m, \widetilde{R})\widetilde{R},\ P_{pub}) \end{aligned}$$

## 4.4 Security Analysis

We still start from assuming the existence of a forger $\mathcal{F}$ and an attacker $\mathcal{A}$, and initialize the system public key as $P_{pub} = aP$. Since the target subgroup we are supposed to attack contains one uncorrupted signer $ID_*$ (we obtain the secret keys of all the other corrupted signers), $\mathcal{A}$ only needs to simulate $P_{uncorrupted}$ during key generation and signing. A big difference to the previous proof is that instead of letting $\mathcal{A}$ flip a coin to decide the corresponding identity is to be attack or not, we calculate the probability of getting a fixed identity by using Cha-Cheon's ID attack [1]. This probability can be used to replace $\delta$.

**Theorem 3.** *In the random oracle model, if a probabilistic polynomial time forger $\mathcal{F}$ has an advantage $\varepsilon$ in forging an ASM with running time $t$ and asking $H_0$,$H_1$,key extraction oracle and signing oracle $q_{H_0}$, $q_{H_1}$, $q_e$ and $q_s$ times respectively, then the CDH problem can be solve with an advantage*

$$\varepsilon' > ((1 - \frac{1}{q})\frac{1}{q_{H_0}})(\varepsilon - \frac{q_S(q_{H_1} + q_S) + 1}{2^k})$$

*with running time $t' < t + (q_{H_0} + 4q_e)t_m$, where $t_m$ is the time to compute a scalar multiplication in $\mathbb{G}_1$.*

*Proof.* In running our simulation, $\mathcal{A}$ answers all the queries the same as in proving online/offline signature scheme. However, the probability calculation is different. We still consider three parts:

1. $\mathcal{A}$'s probability of failure caused by a conflict over $H_1$ is at most $q_S(q_{H_1} + q_S)/q$.
2. the probability of producing a valid forgery without asking $H_1(m^*, R^*)$ is $1/2^k$
3. the probability of $\mathcal{A}$ succeeds in a key extraction query is $\delta(1-\delta)^{q_e}$

In this case, $\delta$ involves two parts:

– the probability of producing a valid message-signature tuple $(ID_*, m, \sigma^*)$ without any query of $H_0$ is at least $1 - \frac{1}{q}$
– the probability that$ID^*$ is chosen randomly from the space of $H_0$ is at least $\frac{1}{q_{H_0}}$

The resulting probability that a target ID appears in our simulation is $\delta \geq (1 - \frac{1}{q})\frac{1}{q_{H_0}}$. Thus in (3) we have

$$1 - \delta = 1 - \frac{1}{q_{H_0}} + \frac{1}{q \cdot q_{H_0}} > 1 - \frac{1}{q_{H_0}}$$

$$\begin{aligned}\delta(1-\delta)^{q_e} &= ((1 - \frac{1}{q})\frac{1}{q_{H_0}})(1 - \frac{1}{q_{H_0}})^{q_e} \\ &> (1 - \frac{1}{q})\frac{1}{q_{H_0}}\end{aligned}$$

Eventually it comes that $\mathcal{A}$'s advantages is at most

$$((1 - \frac{1}{q})\frac{1}{q_{H_0}})(\varepsilon - \frac{q_S(q_{H_1} + q_S) + 1}{2^k})$$

□

### 4.5 Efficiency Comparison

To compare the efficiency, we assume the safe length of GDH group $\mathbb{G}_1$ is $\rho$ and the order of multiplicative group is $q$. We analyze the efficiency of signature schemes in relation to four indicators: signature size, pre-computation cost, signing cost, verification cost and problem based. We define the pre-computation phase to include all the operations taken irrelevant to the message to be signed. The signing phase only contains the operation aiming at the message. The signing cost and pre-computation cost are justified in terms of the elliptic curve scalar multiplications (ESM), or exponentiations being used. The verification cost is justified by counting the number of pairings being used. We also assume the multisignature is generated by $n$ signers.

We choose five existing ID-based multisignature schemes, in which three of them use bilinear pairings and the other two are based on RSA. Besides our scheme (**IBMS**, based on IOS), another four schemes include: **SOK-IBMS** [12] proposed by Sakai et al., **CZK-IBMS** [3], the ID-based blind multisignature scheme proposed by Chen et al, based on Cha-Cheon scheme [1], **WH-IBMS** [15] proposed by Wu and Hsu, **CLL-IBMS** [2] proposed by Chang et al.

We firstly look at the comparison between single ID-based signature schemes in **Table 1**. It is obvious that our online/offline signature scheme is efficient in online signing since no ESM needs to be performed. Two RSA based signature schemes are efficient in signing and verification, but signature sizes are apparently larger than others.

The comparison of the ID-based multisignature schemes is listed in **Table 2**. We can see that the Chen et al.'s scheme is very efficient in average, requiring $2n$ scalar multiplications in the pre-computation phase and the signing phase. Our scheme preforms the same number of scalar multiplications ($2n$) in pre-computation phase. However, the actual signing phase needs only 1 scalar multiplication. We can draw this conclusion that our ID-based multisignature scheme preserves the advantage of its original scheme (ID-based online/offline signature scheme), which is able to shift the computational overhead to the pre-computation phase.

| | Signature Size | Pre-computation | Signing Cost | Verification Cost | Problem Based |
|---|---|---|---|---|---|
| SOK-IBS | $2\log_2 \rho$ | 1 ESM | 1 ESM | 2 pairings | CDHP |
| Cha-Cheon | $2\log_2 \rho$ | 1 ESM | 1 ESM | 2 pairings | CDHP |
| IOS | $2\log_2 \rho + \log_2 q$ | 2 ESM | 0 ESM | 2 pairings | CDHP |
| WH-IBS | $\log_2 N$ | N/A | 1 expon. | 2 expon. | RSA |
| CLL-IBS | $\log_2 N$ | N/A | 1 expon. | 3 expon. | RSA |

**Table 1.** ID-based Signature Efficiency Comparison

| | Signature Size | Pre-computation | Signing Cost | Verification Cost | Problem Based |
|---|---|---|---|---|---|
| SOK-IBMS | $(n+1)\log_2 \rho$ | $n$ ESM | $\Sigma_{i=1}^{n} i$ ESM | 3 pairings | CDHP |
| CZK-IBMS | $2\log_2 \rho$ | $n$ ESM | $n$ ESM | 2 pairings | CDHP |
| IBMS | $2\log_2 \rho$ | $2n$ ESM | 1 ESM | 2 pairings | CDHP |
| WH-IBMS | $\log_2 q$ | N/A | $n$ expon. | $(n+1)$ expon. | RSA |
| CLL-IBMS | $\log_2 q$ | N/A | $2n$ expon. | 3 expon. | RSA |

**Table 2.** ID-based Multisignature Efficiency Comparsion

## 5 Application to the DSR Protocol

We firstly introduce some basics of the DSR protocol and analyze its security requirements. Then we will provide the implementation of ASM over DSR.

### 5.1 DSR Protocol

DSR stands for dynamic source routing protocol, presented by Johnson and Maltz [7] in 1996. It is an on-demand routing protocol based on the concept of source routing, which means the initiator knows the complete hop-by-hop route to the destination. To perform DSR, each node is required to maintain a route cache which contains the topology information of the network. The route cache is consistently updated to reflect the current situation of the network.

DSR consists of two phases: route discovery and route maintenance. Route discovery is preformed by using route request and route reply packets. When a node wants to send data to another node, it firstly searches its route cache to see if there is a route to this destination. If yes, this route will be used. Otherwise, this node generates a route request packet (RREQ) which consists of a data structure called *route record* listing the IP addresses of all the intermediate nodes. This RREQ will be broadcasted to neighbors. Each of the neighboring nodes will search its own route cache to see if there exists an active route to the destination. If not, it appends its own IP address to *route record* and rebroadcasts it to its neighbors. This process will be continued until the RREQ packet reaches the destination. The original message is not changed during the transmission (except the RREQ data length field which is a number). The resulting route will be found in the *route record*.

In replying the RREQ, the destination node generates a route reply packet (RREP) and sends it back to the initiator by two ways. It could search its route cache, use the route already existed, or perform its own route discovery. It could also simply reverse the sequence of hops in *record list*.

The route maintenance is performed using route error packets (RERR) and acknowledgements (ACK). RERR is sent whenever a fatal transmission problem occurs. The nodes receiving RERR will delete the entry of the error hop in their route caches. On the other hand, ACK is used to verify the availability of route links. The result of ACK will be used to update route caches in order to reflect the current topology.

### 5.2 Security Consideration

The design of the original DSR protocol does not include any security; therefore, it faces several attacks. The most serious attack is the unauthorized modification of route packets. For example, a malicious intermediate node can remove the IP addresses of other nodes in the *route record*. This will result in a fake route to be received by the destination. Besides, malicious nodes can perform an attack called *route cache poisoning*, by fabricating and sending spoofed packets. All the nodes who received the spoofed packets will update their route caches accordingly, therefore poison the route caches.

The above attacks are usually prevented using digital signatures. By introducing digital signatures to the DSR protocol, we can provide both data integrity (against unauthorized modification of routing packets) and hop-by-hop authentication. However, although digital signatures do not prevent fabricating spoofed routing packets, it enables the tracing of malicious nodes, then secures the routing process in a more active manner. However, a normal digital signature is not applicable to DSR situation because of computation complexity: the number of signatures is increasing during routing processes, and the verification time linearly increases. To add security features to the DSR protocol in an efficient manner, we firstly take a closer look at the structure of DSR.

The data structure of RREQ consists of two fields: IP fields and route request fields. IP fields contains source address, destination address and hop limit. Route request fields contains option type, option data length, identification, target address, and route record. When a RREQ is received, the option data length fields will be increased by 4 and the node's IP address will be appended to the end of the route record. Other fields will remain unchanged during the whole route discovery process.

One signature scheme applicable to the DSR situation is called multisignature scheme. Since most fields in a RREQ packet are immutable during the whole routing process, it is possible to create a multisignature over the immutable fields by each node. The option data length fields which is mutable can be protected by a hash chain as in AODV for protecting hop count field. The other mutable field, route record field, due to containing only IP addresses, can be regarded as the public key of each node in an ID-based signature scheme.

### 5.3 Installation of IBMS over DSR

Since the IBMS scheme described in section 4 is "Accountable Subgroup" signature scheme, to enable the installation over DSR, we firstly define the total signers' group to include all the mobile nodes in MANET. The maximum size of the total group $G$ should agree with the network capacity. We then define the subgroup $S$ to include the mobile nodes involved in a routing operation. Therefore, each routing operation will accordingly form a subgroup whose maximum size equals the maximum hop count allowed by DSR protocol. Before a DSR based network is initialized, the total group is set as empty $G \leftarrow \phi$. Mobile nodes will be added to the total group $G$ once they enter the network. Similarly, the subgroup $S$ is initialized as empty $\phi$ as well, and mobile nodes will be added to the subgroup $S$ when they are involved in some routing operations.

To perform the ID-based authentication, we assume the existence of an offline key generation center (KGC). KGC runs the system $\mathsf{Setup}$ algorithm to generate all the parameters required. Each node, before entering the network, has to submit its credential to KGC. The KGC will run the key generation algorithm ($\mathsf{KeyGen}$) to generate a public-secret key pair for each node. One straightforward method is to use a node's IP address as its public key and get the secret key generated over it. The parameters and keys will be transmitted to mobile nodes through a secure channel. Once a node has obtained all the necessary parameters, it can start to do all the pre-computations according to signing algorithm, in order to achieve the best efficiency in signing.

When a RREQ is issued, the initiator runs the signing algorithm Signing to generate a signature over all the immutable fields. The mutable fields, the RREQ data length field and the route address field, are excluded and their values are set to 0 during the signature generation.

The RREQ along with the signature will be broadcasted to next hop neighbors. The next hop nodes will firstly run the verification algorithm Verifying to evaluate the signature validity. To run this algorithm, the verifier firstly needs to extract the IP addresses, which are also the public keys of previous hop nodes, from the RREQ. Accordingly, if a malicious node deliberately removes some IP addresses from the RREQ, the signature will not pass the verification and the route carried by the RREQ will be considered as incorrect and rejected. Therefore, by preforming the signature verification, both the signature and the route are authenticated.

If the signature is valid, the verifier (the next hop neighbor) will produce a new signature over the immutable field of the original received message. This node then appends its own IP address to the RREQ and broadcasts the RREQ along with the signature. The neighbors of the third hop will perform the same operations as the neighbors of the second hop did to produce signatures over the original RREQ generated by the initiator. This process will continue until the RREQ reaches the target node. The target node, after verifying and accepting the RREQ, will respond with a RREP. This RREP will be transmitted back to the initiator along the route discovered. In this condition, the signature of the RREP will be processed the same as the RREQ. The signing process is shown in **Figure 2**.

Our signing algorithm is given in **Figure 3**. In addition, to further improve efficiency, the verification process can be delayed. In this sense, when a node receives the RREQ along with the signature, it will generate a new signature before verifying the received one. However, this node will not update its route cache until the received signature is verified.

One arguable point of using using multisignature in a sequential form is that the node is able to remove itself from the path. We argue that removing itself does not make any sense. To remove itself, a node passes the routing packet to its next hop neighbor without changing anything. For example, the node M receives a route packet from node A and passes the packet to node B without adding its IP address to *route record* and increasing the value of data length field. There are two situations that could happen. Firstly, if node A is in the neighborhood of node B and node M's behavior actually results in a legal route which is one hop shorter. This route will be accepted by node B, or generated by node B sooner or later. On the other hand, if node A is not in the neighborhood of node B, removing node M results in node B to receive a packet from a distant node. Since node B constantly uses acknowledge packet (ACK) to confirm the link, it will detect the illegality of this packet and finally drop it.

## 6 Conclusion

We introduced the notion of ID-based online/offline signature scheme and accountable subgroup multisignature scheme. We presented a generic construction of ID-based accountable multisignature scheme based on ID-based online/offline signature scheme. We also provided a concrete scheme of ID-based online/offline signature scheme and transformed it into the ID-based accountable subgroup multisignature scheme using our generic construction. Our scheme is proved secure against existential forgery under adaptive chosen message attacks based on the random oracle model assuming that CDHP problem is hard. We compared our scheme with other ID-based multisignature schemes and concluded the transformation could inherit the quick signing

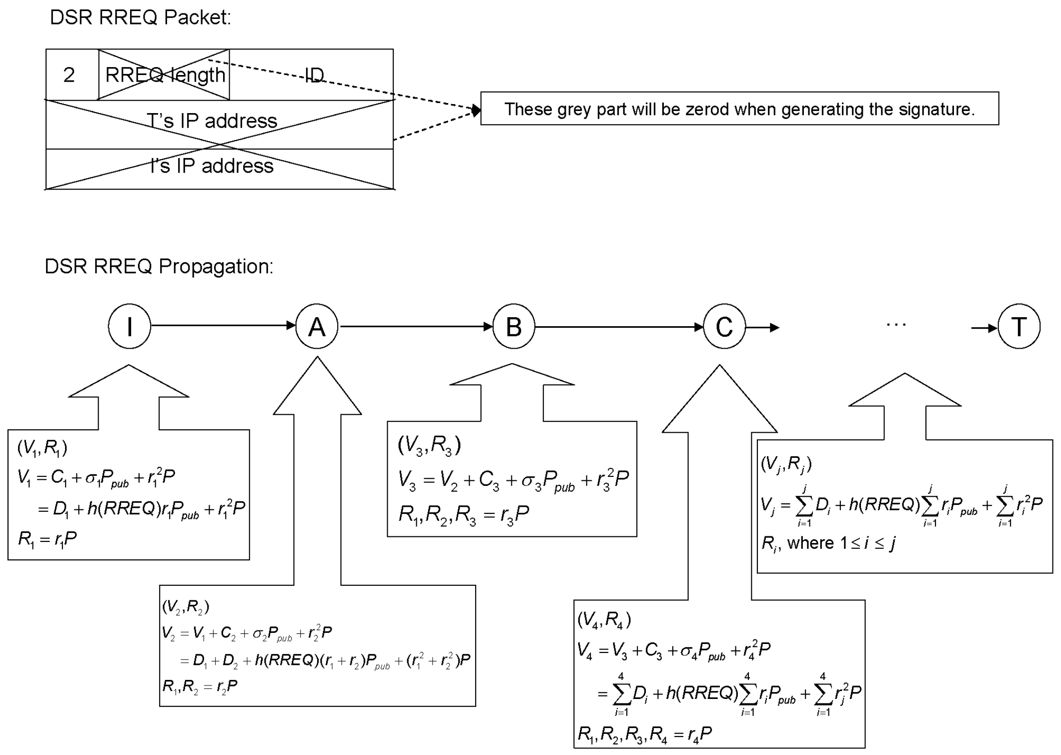
DSR RREQ Packet:
2
RREQ length
ID
T's IP address
I's IP address
These grey part will be zerod when generating the signature.
DSR RREQ Propagation:
I
A
B
C
…
T
$(V_1, R_1)$
$V_1 = C_1 + \sigma_1 P_{pub} + r_1^2 P$
$= D_1 + h(RREQ) r_1 P_{pub} + r_1^2 P$
$R_1 = r_1 P$
$(V_2, R_2)$
$V_2 = V_1 + C_2 + \sigma_2 P_{pub} + r_2^2 P$
$= D_1 + D_2 + h(RREQ)(r_1 + r_2) P_{pub} + (r_1^2 + r_2^2) P$
$R_1, R_2 = r_2 P$
$(V_3, R_3)$
$V_3 = V_2 + C_3 + \sigma_3 P_{pub} + r_3^2 P$
$R_1, R_2, R_3 = r_3 P$
$(V_4, R_4)$
$V_4 = V_3 + C_3 + \sigma_4 P_{pub} + r_4^2 P$
$= \sum_{i=1}^{4} D_i + h(RREQ) \sum_{i=1}^{4} r_i P_{pub} + \sum_{i=1}^{4} r_j^2 P$
$R_1, R_2, R_3, R_4 = r_4 P$
$(V_j, R_j)$
$V_j = \sum_{i=1}^{j} D_i + h(RREQ) \sum_{i=1}^{j} r_i P_{pub} + \sum_{i=1}^{j} r_i^2 P$
$R_i$, where $1 \le i \le j$

**Fig. 2.** IBMS signature generation process in DSR

capability from the online/offline signature scheme. We provided the application over the DSR protocol and argue that our scheme is especially suitable for DSR where the routing messages are modified by appending IP addresses and discussed the implementation issue of the DSR protocol.

## References

1. J. Cha and J. Cheon. An id-based signature from gap-diffie-hellman groups. In *Proceedings of Public Key Cryptography - PKC 2003*, volume 2567, pages 1–24. Springer-Verlag, 2003.
2. C. Chang, I. Lin, and K. Lam. An id-based multisignatures scheme without reblocking and predetermined signing order. In *Computer Standards and Interfaces*, pages 407–413. Elsevier Science Inc., 2004.
3. X. Chen, F. Zhang, and K. Kim. Id-based multi-proxy signature and blind multisignature from bilinear pairings. In *Proceedings of KIISC'2003*, pages 11–19, 2003.
4. D.Boneh, B. Lynn, and H. Shacham. Short signature fromt eht weil pairing. In *Proceedings of Asiacrypt '01, Lecture Notes in Computer Sciences*, volume 2248, pages 514–532. MANET working group, 2001.
5. S. Even, O. Goldreich, and S. Macali. On-line/off-line digital signatures. In *Proceedings of Advances in Cryptology: Crypto '89*. Springer, 1990.
6. K. Itakura and K. Nakamura. A public-key cryptosystem suitable for digital multisignatures. NEC Research and Development, 1983.
7. D. B. Johnson, D. A. Maltz, and Y. C. Hu. *The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)*, 2004.
8. B. Libert and J-J. Quisquater. The exact security of an identity based signature and its applications. In *Cryptology ePrint Archive, Report 2004/102*, 2004.

The node $n_t$ $(1 \leq t \leq L)$ in $G$ receives the signature $(\widetilde{V_{t-1}}, R_1, ..., R_{t-1})$, from its previous hop, where

$$\begin{aligned}\widetilde{V_{t-1}} &= \sum_{j=1}^{t-1}(C_j + \sigma_j P_{pub}) \\ &= \sum_{j=1}^{t-1} D_j + \sum_{j=1}^{t-1} H_1(RREQ, R_j) r_j P_{pub} \\ R_{t-1} &= r_{t-1}P\end{aligned}$$

**Signing.** The node $n_t$ does the followed:

1. randomly choose $x_t, r_t \in \mathbb{Z}_q^*$
2. compute $C_t = D_t - x_t P_{pub}$
3. compute $\sigma_t = H_1(RREQ) r_t + x_t$

With previous received signature $(\widetilde{V_{t-1}}, \widetilde{R_{t-1}})$, the current signer computes:

$$\begin{aligned}\widetilde{V_t} &= \widetilde{V_{t-1}} + (C_t + \sigma_t P_{pub}) \\ &= \sum_{j=1}^{t} D_j + \sum_{j=1}^{t} H_1(RREQ, R_j) r_j P_{pub}\end{aligned}$$

The final signature is $(\widetilde{V_t}, R_1, ..., R_t)$.

**Verifying.** The node $n_t$ checks if the equation holds

$$e(\widetilde{V_{t-1}},\ P) = e(\sum_{j=1}^{t-1} Q_j + \sum_{j=1}^{t-1} H_1(RREQ, R_j) R_j,\ P_{pub})$$

**Fig. 3.** Detailed Algorithm for DSR RREQ packet

9. S. Micali, K. Ohta, and L. Reyzin. Accountable-subgroup multisignatures. In *Proceedings of the 8th ACM Conference on Computer and Communications Security: CCS '01*. Springer, 2001.
10. P. Papadimitratos and Z. J. Haas. Secure routing for mobile ad hoc networks. In *Proceedings of the SCS Communication Networks and Distributed Systems Modeling and Simulation Conference: CNDS 2002*, 2002.
11. C. E. Perkins, E. M. Royer, and S. R. Das. *Ad Hoc On-Demand Distance Vector (AODV) Routing*, 2003.
12. R. Sakai, K. Ohgishi, and M. Kasahara. Cryptosystems based on pairing. In *Proceedings of Symposium on cryptography and Information Security: SCIS 2000*, 2000.
13. C. P. Schnorr. Efficient signature generation by smart card. *Journal of Cryptology*, 4:161–174, 1990.
14. A. Shamir and Y. Tauman. Improved online/offline signature schemes. In *Proceedings of Advances in Cryptology: Crypto '01*. Springer-Verlag, 2001.
15. T. Wu and C. Hsu. Id-based multisignatures with distinguished signing authorities for sequential and broadcasting architectures. In *Applied Mathematics and Computation*, pages 349–356. Elsevier Science Inc., 2002.
16. S. Xu, Y. Mu, and W. Susilo. An efficient authentication scheme for manet routing. In *Proceedings of the Embedded and Ubiquitous Computing: EUC 2005 Workshops*. Springer, 2004.