A Certified E-Mail Protocol Suitable for Mobile Environments

Jung Min Park
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN 47907-1285 USA
parkjm@ecn.purdue.edu

Indrajit Ray*, Edwin K. P. Chong†§, and Howard Jay Siegel†*‡§
‡Department of Electrical and Computer Engineering
‡†Department of Mathematics
§Department of Computer Science
Colorado State University
Fort Collins, CO 80523-1373 USA
{indrajit, echong, hj}@colostate.edu

Abstract—We describe a novel certified e-mail protocol that is particularly suitable for mobile environments. Our protocol uses an off-line trusted third party (TTP). Protocols with an off-line TTP—also known as optimistic protocols—have numerous practical advantages over protocols with an on-line TTP. Nonetheless, many protocols adopt an on-line TTP primarily because optimistic protocols often entail intricate cryptographic primitives that incur considerable overhead. By using a novel signature paradigm, which we call gradational signatures, we show that it is possible to construct optimistic protocols that are comparable to on-line protocols in terms of computation and communication overhead. This makes our scheme especially desirable in the mobile setting.

Keywords—certified e-mail; fair exchange; multisignatures

I. INTRODUCTION

E-mail is fast replacing postal mail as the preferred method of correspondence. However, specialized services similar to those offered by postal mail need to be supported by e-mail before the latter can gain wider acceptance. One such service is certified e-mail. Here, a receiver gets to access the e-mail’s contents if and only if the sender receives an irrefutable proof-of-receipt. We propose a novel certified e-mail protocol that uses an off-line trusted third party (TTP). Low computation overhead and minimal participation of the TTP make our scheme particularly suitable for mobile environments, where the communication devices have limited computation and storage abilities.

The problem of certified e-mail belongs to the more general class of problems known as the fair-exchange problem [2]; the goal is to ensure the fair exchange of an e-mail message and its corresponding signed receipt. Additionally, the mobile environment imposes considerable restrictions in terms of computation, storage, and communication overhead. As a result, existing fair-exchange protocols cannot be adapted easily for certified e-mail in the mobile setting. In general, fair-exchange protocols can be classified into three categories: (i) gradual-exchange protocols, (ii) protocols requiring an on-line TTP, and (iii) protocols requiring an off-line TTP. Gradual-exchange protocols are impractical for most applications because they are computationally expensive, and have considerable communication overhead. Protocols using an on-line TTP require the TTP to be available for the entire lifetime of the exchange. Thus, although such protocols are efficient in terms of computation, they are not exactly suitable for the wireless mobile setting. The third class of protocols seems more suitable. They are called “optimistic” protocols because the TTP is involved only if one of the parties behaves unfairly or aborts prematurely. Such situations are more the exception than the rule, and thus the TTP’s involvement is minimal.

To the best of our knowledge, the scheme of Ateniese and Nita-Rotaru [5] is at the vanguard of optimistic certified e-mail protocols—it offers several advantages over other schemes. However, the protocol uses verifiable encryption, which makes it more computationally intensive than on-line TTP protocols, and hence it might be impractical for the mobile setting. We present an optimistic protocol that is more efficient. For this purpose, we propose a novel signature paradigm called gradational signatures. Specifically, we employ gradational signatures based on the Guillou-Quisquater (GQ) signature scheme [8] to compute the receipt. This makes our scheme comparable, in terms of computation overhead, to on-line protocols.

In the next section, we discuss related work. We discuss properties of certified e-mail protocols that are desirable for mobile environments in Section III. In Section IV, some technical background is given. We present our protocol in Section V, and evaluate its features in Section VI. The concluding remarks are given in Section VII.

II. RELATED WORK

Because of efficiency, many schemes employ an on-line TTP. Bahreman and Tygar [6] propose a protocol that employs the TTP as a courier of the e-mail and receipt. In [11], Zhou and Gollman describe a scheme that delivers the e-mail via a sequence of TTPs. The receipt signed by the recipient is routed back to the originator by the TTPs. Deng et al. [7] propose an efficient on-line protocol that requires only four messages. In [1], Abadi et al. discuss a protocol that uses an on-line TTP as a decryption server; that is, the e-mail message is encrypted under a key that is encrypted under the TTP’s public key.

Maintaining a fault tolerant TTP that needs to be constantly on-line can be expensive. Some schemes avoid this problem by using optimistic protocols. Protocols such as [3, 5, 10] fall into...
this class. Optimistic protocols are discussed further in Section VI. In [4], Ateniese et al. discuss a hybrid model, TRICERT, that combines the features of on-line and off-line TTP models. Their scheme employs semi-trusted servers, called “postal agents” (PA), to carry out the exchanges, and invokes the fully-trusted TTP only when disputes need to be settled. This approach alleviates the cost of maintaining an on-line TTP. The PA, however, might become a bottleneck.

III. CERTIFIED E-MAIL IN MOBILE ENVIRONMENTS

A certified e-mail protocol should have the following properties (some of which were taken from [5]):

- **Fairness:** No party should be able to corrupt the protocol or abort prematurely to gain an advantage. At the termination of the protocol, either each party gets the other party’s item, or neither party does.
- **Timeliness:** Both parties should be able to terminate the exchange within a given finite time. If the items being exchanged are time-sensitive, fairness cannot be ensured without the timeliness property.
- **Confidentiality:** Only the intended receiver (and not even the TTP) should have access to the contents of the e-mail.
- **Minimal participation of the TTP:** The TTP’s computation and storage requirements per exchange should be kept to a minimum.

The fairness and timeliness properties are basic requirements, while the remaining properties are optional requirements that might be desired in certain cases.

In mobile environments, additional properties are desirable because of the computation and storage limitations of the mobile devices. This is especially true if tamper-proof hardware such as smart cards is used to handle cryptographic computations. A typical smart card has a storage capacity limited to a few kilobytes, and employs a 32-bit RISC microprocessor operating at 15–25 MHz. We concentrate on the case where the receiver is a mobile device—such devices are far more likely to be used for receiving certified e-mail rather than sending them. The following additional properties are desirable for mobile environments:

- **Stateless receiver:** The receiver does not need to store any state information to execute the exchange protocol.
- **Minimal involvement in dispute resolutions:** The receiver’s participation in dispute resolutions is minimal.
- **Computational efficiency:** The protocol’s computation overhead should not be excessive for the receiver.
- **Minimum number of message exchanges:** The number of required message exchanges should be small.

The first property is desirable because of the mobile device’s storage limitation. The rest of the properties, directly or indirectly, determines the receiver’s computation load, which affects the device’s power consumption and battery life.

IV. TECHNICAL BACKGROUND

A. The Gradational Signature Paradigm

To ensure fairness, we distribute the computation of the receipt between the receiver and the TTP. Note that the receiver’s signature on the e-mail message acts as the receipt. We use a novel signature paradigm called gradational signatures to compute the receipts. Although gradational signatures have many traits in common with multisignatures, they possess unique characteristics that set them apart. A multisignature scheme allows multiple signers to sign a single message efficiently. The players in a typical multisignature scheme are $n (\geq 2)$ signers and a verifer. The players in the gradational signature paradigm include a primary signer, a cosigner, and a verifier. The cosigner has knowledge of its partial private key $sk$ only, while the primary signer has knowledge of its partial private key $sk$ as well as $sk$. Both keys are needed to compute the full signature $\sigma$, while only $sk$ is required to compute the partial signature $\sigma_i$. In essence, the partial private keys, $sk$, and $sk$, jointly form a full private key $sk$ that is required to compute $\sigma$. To verify $\sigma$ and $\sigma_i$, the verifier needs to use the full public key $pk$ and the partial public key $pk$, respectively. The full signature is identical to a conventional signature in form and functionality. In our model, the cosigner holds $sk$ so that it can play an auxiliary role in the full signature generation process. In contrast, the primary signer is the “owner” of the full signature, that is, the full signature provides nonrepudiation of the message signed by the primary signer.

Another important feature of our paradigm is that only $pk$ (and not $pk$) is certified by the certification authority (CA). Because $pk$ is not certified, $\sigma_i$ is only a commitment, and is not a cryptographic primitive that ensures nonrepudiation. Hence, the partial signature, unlike a conventional signature, has no intrinsic value on its own, but does force the primary signer to commit to the transaction in which it is used. In contrast, $pk$ is certified, and hence $\sigma$ provides nonrepudiation.

B. The GQ-Based Gradational Signature Scheme

We construct GQ gradational signatures by applying our gradational signature paradigm to GQ signatures. The resulting scheme has similarities with GQ multisignatures [8, 9], but also has distinct features.

KEY GENERATION. The primary signer chooses two distinct primes $p$ and $q$, and forms $N = pq$. Next, it selects an integer $\nu \in \{1, \ldots, N - 1\}$ such that $\gcd(\nu, \phi(N)) = 1$, where $\phi$ is the Euler totient function. The integers $\nu$ and $N$ are the public system parameters. The primary signer also selects a full public key $J$ such that $1 < J < N$. The integer $J$ is a numeric representation of the primary signer’s unique identity (e.g., name, account number, and serial number). Next, the primary signer randomly selects an integer $J \in Z_{N}^{\ast}$, and computes an integer $J$, that satisfies $J \equiv J (\text{mod } N)$. Here, $Z_{N}^{\ast}$ denotes the multiplicative group of integers modulo $N$. The integer $J$ is
the partial public key. The partial private keys $\beta_1$ and $\beta_2$ are determined by finding the solutions to
\[
J_i \beta_i^r \equiv 1 \pmod{N}, \quad i = 1, 2.
\]
The full private key $\beta$ is given by \(\beta \equiv \beta_1 \beta_2 \pmod{N}\).

**SIGNATURE GENERATION.** For every instance of a gradientia- tional signature operation, the primary signer selects a random integer $r$, and computes \(T = r^j \pmod{N}\). Then, the following values are computed:
\[
\begin{align*}
  d &= H(M \parallel T), \\
  D &= r \beta^t \pmod{N},
\end{align*}
\]
where $H$ is a suitable hash function, $M$ is the message being signed, and $\parallel$ denotes concatenation. The pair $(D, d)$ is the partial signature. To verify the partial signature, one needs to compute \(d^* = H(M \parallel T^r)\), where $T^r = D D_{\beta}^j \pmod{N}$. The partial signature is valid if and only if \(d = d^*\). To compute the full signature, one must first compute $D_{\beta} = D D_{\beta}^j \pmod{N}$, where $D_{\beta} = \beta^t \pmod{N}$. The pair $(D, d)$ is the full signature. To verify the full signature, the verifier computes \(d' = H(M \parallel T^r)\), where $T^r = D' D_{\beta}^j \pmod{N}$. The full signature is valid if and only if \(d = d'\).

**V. THE CERTIFIED E-MAIL PROTOCOL.**

In the following protocol, Alice is the sender (verifier), Bob is the receiver (primary signer), and Charlie acts as the TTP (cosigner). We assume that the public keys of the CA and the three parties are known to everyone. The value $M$ represents an e-mail message, $\sigma$ represents a receipt (full signature), and $\sigma_1$ represents a partial receipt (partial signature).

**REGISTRATION.** The registration protocol needs to be performed only once, after which it can support any number of exchanges. Before the registration protocol is executed, Bob selects the public parameters $(N, v)$, and generates the full public key $J_1$ using his unique identity. He also chooses the partial public key $J_2$, and computes the corresponding $J_2$. Next, Bob computes the partial private keys $\beta_1$ and $\beta_2$. He then contacts the CA to get $J_1$ and the parameters $(N, v)$ certified. After verifying that $J$ matches Bob’s identity, the CA issues a certificate $C_{CA}$ (to Bob) that guarantees the authenticity of $J$ and $(N, v)$. It is assumed that the three items can be recovered from $C_{CA}$. We assume that the registration protocol is performed via private and authenticated channels. In practice, such channels can be implemented by using Message Authentication Codes (MAC) in conjunction with encryption schemes.

1. Bob sends to Charlie $C_{CA}$, $J_1$, $J_2$, and $\beta_1$.
2. Charlie first extracts $J$ from $C_{CA}$. He then checks the following congruence relations:
\[
\begin{align*}
  J &\equiv J_1 J_2 \pmod{N}, \\
  J_2 \beta_1^r &\equiv 1 \pmod{N}.
\end{align*}
\]
If the above two relations hold, Charlie creates a voucher $V_c$ by signing $J_1$. It is assumed that $J_1$ can be recovered from $V_c$. Charlie sends $V_c$ to Bob, and securely stores $\beta_2$.

**Remark 1.** The voucher $V_c$ issued by Charlie guarantees the authenticity of Bob’s partial public key $J_1$. The voucher also (implicitly) implies that Charlie has the ability to convert Bob’s partial receipt (i.e., his partial signature) into the corresponding receipt (i.e., Bob’s full signature). This is true because Bob will generate the voucher only after verifying the congruence relations of Step 2 of the registration protocol. We can show that when the keys satisfy these relations, possessing the partial private key $\beta_1$ (and the public parameters) is sufficient for converting a partial signature into the corresponding full signature.

**Remark 2.** If the number of primary signers is large, it requires Charlie to securely store a correspondingly large number of $\beta_1$’s (one for each primary signer). This can be avoided by using the following technique: Charlie concatenates $\beta_2$ and Bob’s unique identification $ID_b$ to form $\beta_1 \parallel ID_b$, and then encrypts this value via some symmetric encryption scheme $E_\lambda(\cdot)$, where $\lambda$ denotes the encryption key. Charlie then signs the concatenated value of $J_1$ and $E_\lambda(\beta_2 \parallel ID_b)$, This value $\text{Sig}_C(J_1 \parallel E_\lambda(\beta_2 \parallel ID_b))$, where $\text{Sig}_C(\cdot)$ denotes the signature generation algorithm, is used as the voucher $V_c$. Charlie can extract $\beta_2$ from $V_c$ (using $\lambda$), and only needs to securely store $\lambda$ instead of storing $\beta_1$’s for each primary signer.

**EXCHANGE.** Alice initiates the exchange protocol with Bob. The messages exchanged in the exchange protocol (when both parties act honestly) are illustrated in Fig. 1.

1. Alice encrypts $M$ with Bob’s public key via some asymmetric encryption scheme, and computes the hash of this value. We denote this value as $H(AE_b(M))$. Alice concatenates $H(AE_b(M))$ and a header $HD_A$, and signs this value via some signature scheme. (See Remark 3 for details on the contents of $HD_A$.) We denote this value as $s_A = \text{Sig}_A(HD_A \parallel H(AE_b(M)))$. The value $s_A$ is sent to Bob.
2. Based on the information in $HD_A$, Bob determines whether he should receive or reject the e-mail message. If
Bob decides to continue with the protocol, he computes the partial receipt \( \sigma_t = (D, d) \) (i.e., the partial signature). He then concatenates a header \( HD_s \) and a timestamp \( T \) before signing the two items using an arbitrary signing algorithm. We denote this value as \( s_b = \text{Sig}_a(HD_s \| T) \). We assume that the items that are signed can be extracted from \( s_b \).

Bob sends \( s_b, \sigma_t, V_c, \) and \( C_{CA} \) to Alice.

3. Using the keys and parameters recovered from \( V_c \) and \( C_{CA} \), Alice verifies \( \sigma_t \). If it is valid, Alice encrypts \( M \) with Bob’s public key via an asymmetric encryption scheme \( AE_b(\cdot) \). Alice sends the encryption to Bob.

4. Bob decrypts and reads the e-mail message. Then, he computes the receipt \( \sigma = (D, d) \), and sends this to Alice.

**Remark 3.** The headers (i.e., \( HD_1 \) and \( HD_2 \)) contain the necessary protocol information such as the exchanging parties’ identities and the cryptographic algorithms being used. In \( HD_2 \), a short description of the e-mail content should also be included. Using this information, Bob can decide whether to accept or reject the e-mail before committing to it.

**Remark 4.** The e-mail message (i.e., \( M \)) is encrypted under Bob’s public key to ensure privacy.

**Remark 5.** Note that Bob does not need to store any state information to execute the exchange protocol.

**Remark 6.** When Bob computes the partial and full signatures, he actually creates them using \( H(AE_b(M)) \) instead of \( M \); that is, he computes \( d = H(H(AE_b(M)) \| T) \) instead of \( d = H(M \| T) \) (see Section IV-B). Therefore, \( \sigma = (D, d) \) should be interpreted in a special way: it is considered a valid receipt of \( M \) if and only if \( d' = d \), where \( d' = H(H(AE_b(M)) \| T') \), \( T' = D' J^d \mod N \), and \( AE_b(\cdot) \) is the asymmetric encryption algorithm used to encrypt \( M \).

**DISPUTE RESOLUTION.** If Alice does not receive the receipt, or if the receipt is invalid, she initiates a dispute resolution protocol by contacting Charlie.

1. Alice sends the following items to Charlie: \( s_b, \sigma_t, V_c, C_{CA}, \) and \( AE_b(M) \).

2. Charlie extracts the keys and parameters from \( V_c \) and \( C_{CA} \), and verifies the partial receipt \( \sigma_t \). He then checks to see whether the time on \( T \) has expired; if the time has expired, Charlie stops the protocol and refuses Alice’s request. If everything is in order, Charlie converts the partial receipt \( \sigma_t \) into the corresponding receipt \( \sigma \) using his partial private key \( \beta_c \). Charlie sends \( \sigma \) to Alice, and forwards \( AE_b(M) \) to Bob.

**Remark 7.** Bob needs to include a timestamp in \( s_b \) (see Step 2 of the exchange protocol) to ensure the timeliness property. Consider the following scenario: after Step 2 of the exchange protocol, Alice aborts the protocol. After a deliberate delay, she initiates a dispute resolution protocol with Charlie. Unaware of Alice’s malicious intentions, Charlie sends \( \sigma \) to Alice, and forwards \( AE_b(M) \) to Bob. If the contents of the e-mail were time-sensitive, the forwarded information to Bob is useless. The inclusion of \( T \) in \( s_b \) prevents such a scenario.

**Remark 8.** In Step 2 of the dispute resolution protocol, Charlie needs to forward \( AE_b(M) \) to Bob to ensure fairness. This is because Alice might attempt to obtain the receipt via the dispute resolution protocol without sending \( AE_b(M) \) to Bob.

**Remark 9.** Note that Bob does not participate in the dispute resolution protocol. He only needs to receive \( AE_b(M) \) if it is forwarded to him.

VI. EVALUATIONS

A. Comparison with Other Schemes

In Table 1, we compare our protocol with the other protocols discussed in the paper. The numbers in the square brackets correspond to the reference numbers of the schemes, and “G.S.” denotes our protocol. Protocols with an on-line TTP do not have dispute resolution protocols, and hence the fourth criterion is not applicable to them. For optimistic protocols, the last criterion refers to the number of message exchanges in the exchange protocol (and not the dispute resolution protocol).

In on-line TTP protocols, reliable channels\(^1\) [2] between the two parties and the TTP need to be maintained at all times, which might be difficult for wireless mobile environments. Although optimistic protocols avoid these problems, many schemes adopt an on-line TTP because of the considerable computation and communication overhead incurred by optimistic protocols. The optimistic protocol of Asokan et al. [3] requires the parties to perform a cut-and-choose protocol, which is costly. In [10], Schneier and Riordan propose an optimistic

---

\(^{1}\) For simplicity, we assume that \( M \) is encrypted directly with an asymmetric encryption algorithm. In practice, one would first encrypt \( M \) with key \( k \) via some symmetric encryption algorithm, and then encrypt \( k \) with an asymmetric encryption scheme.

---

\(^{4}\) A channel that is always operational with a known upper bound of the time delay. An attacker cannot delay any messages beyond the known upper bound.
An efficient optimistic protocol ensures a weaker guarantee of fairness. If one party obtains the other's item without sending its, the exchange protocol fails. We presented a novel approach for constructing optimistic certified e-mail protocols. Despite the advantages of optimistic protocols, many protocols adopt an online TTP because of considerable computation and communication overhead. We solved this problem by applying a novel signature paradigm—gradational signatures—to construct an efficient exchange protocol using GQ signatures. In fact, the exchange phase of our certified e-mail protocol requires very little computation and communication overhead beyond what is typically required in on-line protocols. Such features are particularly suitable for mobile devices with limited computation and storage capabilities.

### TABLE I. COMPARISON OF CERTIFIED E-MAIL PROTOCOLS

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>confidentiality</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>TTP participation</td>
<td>on-line</td>
<td>on-line</td>
<td>on-line</td>
<td>off-line</td>
<td>off-line</td>
<td>off-line</td>
<td>off-line</td>
</tr>
<tr>
<td>stateless receiver</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>receiver's participation in dispute resolutions</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>receiver's computation load</td>
<td>light</td>
<td>light</td>
<td>light</td>
<td>light</td>
<td>heavy</td>
<td>medium</td>
<td>light</td>
</tr>
<tr>
<td>number of message exchanges</td>
<td>6</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>8</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

### VII. Conclusions

We presented a novel approach for constructing optimistic certified e-mail protocols. Despite the advantages of optimistic protocols, many protocols adopt an online TTP because of the considerable computation and communication overhead. We solved this problem by applying a novel signature paradigm—gradational signatures—to construct an efficient exchange protocol using GQ signatures. In fact, the exchange phase of our certified e-mail protocol requires very little computation and communication overhead beyond what is typically required in on-line protocols. Such features are particularly suitable for mobile devices with limited computation and storage capabilities.

### References


B. Evaluation of Fairness

We do not present a formal analysis of fairness in this paper. In the following, we informally describe the two different notions of fairness that apply to the entities of the exchange protocol (i.e., Alice and Bob).

Fairness is ensured for Alice as long as she is assured of receiving $\sigma$ after sending $AE_\sigma(M)$ to Bob. Recall that Alice obtains $\sigma$, before sending $AE_\sigma(M)$, and Charlie obtains $\beta_2$ in the registration protocol. Even if Bob refuses to send $\sigma$, Alice can obtain it via Charlie who is able to compute $\sigma$ using $\sigma_1$ and $\beta_2$. If Alice fails to receive $\sigma_1$, fairness is still ensured (for Alice) because she can refuse to send $AE_\sigma(M)$ in the next step. In this case, both parties terminate the exchange protocol without obtaining the desired items.

Fairness is ensured for Bob as long as Alice is unable to forge $\sigma$ using the values sent to her in Step 2 of the exchange protocol. It is infeasible for Alice to forge $\sigma$ (without the help of Charlie) if we assume the infeasibility of breaking the GQ gradational signature scheme by a polynomial time adversary.