A digital signature with multiple subliminal channels and its applications

Dai-Rui Lin*, Chih-I Wang, Zhi-Kai Zhang, D.J. Guan

Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung, 804, Taiwan, ROC

ARTICLE INFO

Keywords:
Subliminal channels
Digital signatures
One-time signatures
Quadratic residues
Chinese remainder theorem

ABSTRACT

In this paper, we present two schemes for embedding multiple subliminal messages into one-time signature schemes (OTSSs) proposed by Lamport (1971, 1981) [35, 36]. Our schemes have the advantage that the subliminal receivers cannot forge a valid signature since they do not share the signer’s secret key. Our schemes can also provide more than one independent subliminal message, and the numbers of subliminal messages and receivers are larger than that of the subliminal messages in previous schemes.

© 2010 Elsevier Ltd. All rights reserved.

1. Introduction

A subliminal channel is a communication channel that allows a sender to transmit an additional secret message to authorized receivers. To do this, subliminal receivers have to share a subliminal key with the signer to protect the subliminal message. Without additional knowledge, the secret message cannot be detected and discovered by any unauthorized receivers.

In 1983, Simmons first constructed a subliminal channel in a digital signature scheme [1]. Since then, many studies on subliminal channels have been published [2–23]. In the Appendix, we show the history of subliminal channel-related publications from 1983 to 2009.

Harn and Gong proposed two digital signature schemes with two subliminal channels in 1997 [3]. The main feature of their schemes was that the subliminal receivers had to share a part of the signer's secret key as the subliminal key. Therefore, their schemes could be vulnerable to conspiracy attack. That is, if a sufficient number of subliminal receivers conspire against the message signer, they can derive the secret key and forge a valid signature, thereby reducing the security of the digital signature scheme. In 1999, Jan and Tseng proposed two digital signature schemes with subliminal channels on the basis of the discrete logarithm problem [4]. Their first scheme has the same problem of conspiracy attack as Harn and Gong's scheme [3]. The security of their second scheme could also be vulnerable to conspiracy attack. The subliminal receivers can obtain information about the signer's secret key through cooperation of the receivers. Lee and Lin [24] pointed out that Jan and Tseng's schemes could be vulnerable to a dishonest receiver attack, wherein a malicious designated receiver can forge the signature and hide a new subliminal message in the signature. The new subliminal message will be accepted by other receivers. Lee and Lin also showed an improvement for avoiding the above security flaw [24].

One-time signature schemes (OTSSs) are secure, fast, and have many applications [25–27]. They are also useful in online, off-line, and forward-secure signatures [28]. Recently, several OTSSs have been proposed [25, 29, 30, 27, 31–33], and some stream signature schemes using one-time signatures have been presented [28, 34, 29, 30, 26]. Stream signature schemes are used for streamed media authentication and signing. Streamed media, such as streamed radio and video, broadcast or multicast via the Internet. In order to enable a widespread and trusted streamed media dissemination, the user needs

* Corresponding author.
E-mail address: javacpc@gmail.com (D.-R. Lin).

0898-1221/$ – see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.camwa.2010.01.001
assurance that the data stream originated from the purported sender. Therefore, using a one-time signature is a good method for signing the digital streams. The cited researchers pointed out that OTSSs have many applications.

We propose two schemes for embedding multiple subliminal messages based on an important concept: the subliminal secret keys of each of the subliminal receivers are not only independent of each other but also independent from the secret signing key. This can ensure the security of both the signature and the subliminal messages. In our two subliminal channel schemes, the subliminal keys are independent from the signer’s secret key, and this avoids vulnerability to conspiracy attacks [3]. Moreover, an attack using a malicious subliminal receiver [24] will not work in either of our schemes. That is, a malicious subliminal receiver cannot forge subliminal messages that will be accepted by other subliminal receivers that belong to the malicious subliminal receiver’s channel.

In a digital signature scheme such as RSA, ElGamal and DSA, a hash function is applied before signing to shorten the signature. This limits the size of subliminal messages that can be embedded in these signature schemes. For example, if the computation is in Z_p and the hash of the message is k bits long, the subliminal message in these schemes can be no more than log p bits. On the other hand, the size of an OTSS is usually large. This feature is useful in embedding subliminal messages because it sufficiently increases the size of the subliminal message that a meaningful long message can be sent. For example, in our second scheme, the length of the subliminal message depends on the amount of 1 bits in the hash of the message. In an average case, the subliminal message can be as large as 2 log p bits.

The rest of this paper is organized as follows. In the next section, we briefly review Lamport’s OTSSs. The proposed schemes are presented in Section 3. Finally, a security analysis and concluding remarks are given in Sections 4 and 5, respectively. The Appendix shows the history of subliminal channel-related publications from 1983 to 2009.

2. Preliminaries

2.1. Review of Lamport’s one-time signature schemes

The first of Lamport’s OTSSs was presented in [35]. Suppose that the signer S wants to sign a message MSG which may be quite long. In the signature scheme, the signer signs the hashed value of MSG instead of MSG itself. Let M, whose length is n, be the hash of MSG (M = H(MSG) ∈ {0, 1}^n). An extra log_2(n + 1) bits are appended at the end of the message. The value represented in the appended bits can be the length or the checksum of the message M.

Lamport’s scheme is divided into three phases: (1) key generation, (2) signature generation, and (3) signature verification.

1. **Key generation phase**: The signer S chooses n + log n positive integers x_1, x_2, . . . , x_{n+log_n} that are large enough as the secret key SK. Then the public key is PK = H(H(x_1), H(x_2), . . . , H(x_{n+log_n})) = H(y_1, . . . , y_{n+log_n}), where y_i = H(x_i), 1 ≤ i ≤ n + log n.

2. **Signature generation phase**: The signature of M is (y_1, y_2, . . . , y_{n+log_n}). Each y_i is computed as follows: If the i-th bit of M, m_i, is equal to 1, then s_i = x_i; otherwise, s_i = H(x_i). The signature of M is σ = (s_1, s_2, . . . , s_{n+log_n}).

3. **Signature verification phase**: To verify the signature, let y_i' = H(s_i) when the i-th bit of M is 1 and y_i' = s_i when the i-th bit of M is 0. The signature is valid if H(y_1', y_2', . . . , y_{n+log_n}) = PK = H(y_1, y_2, . . . , y_{n+log_n}).

The second of Lamport’s OTSSs was presented in [36]. The length of the public key is two times that of the key used in the first of Lamport’s OTSSs. As in the previous scheme, the signer signs the hash value of MSG instead of MSG itself. Let M, whose length is n, be the hash value of MSG. Note that there are no extra log_2(n+1) bits appended to the end of the message in this scheme. We will not describe it in detail here.

2.2. Definitions

In this subsection, we will describe a generic multiple-subliminal-channel scheme by making use of the generic algorithm representation, which will be adopted to construct our two schemes. All components and several definitions are given below.

**Definition 1.** A generic multiple–subliminal-channel scheme based on Lamport’s OTSSs consists of five algorithms: Setup, KeyGen, Embed & Signing, Verifying and Extract, which are described as follows:

- **Setup**: Let l be a security parameter and H be a collision-resistant hash function, where H : {0, 1}^l → {0, 1}^{l}. There are one signer S and a receiver set R = {R_1, R_2, . . . , R_{n+log n}/2} in the proposed scheme. The corresponding subliminal key K of R is K = {k_1, k_2, . . . , k_{n+log n}/2}, where k_i ∈ {0, 1}^{l} is a prime number. Each subliminal receiver in R has to pre-share the corresponding subliminal key in K with S in advance. Assume that each receiver R_j knows j, the position of his/her subliminal message in the signature stream.

- **KeyGen**: This is the key generation algorithm. Given secure parameters l and n, the algorithm will return the candidate secret key CSK = {sk_1, sk_2, . . . , sk_{n+log n}}, where sk_i ∈ {0, 1}^{l}.

- **Embed & Signing**: This is the subliminal message embedding and signing algorithm. Given CSK, the hashed message M = (m_1, m_2, . . . , m_{n+log_n}) to be signed and the subliminal message SM = {sm_1, sm_2, . . . , sm_{n+log n}/2}, the algorithm will return the signature σ and the public key PK = {pk_1, pk_2, . . . , pk_{n+log_n}} of σ, where pk_i = H(sk_i).
• **Verifying**: The verification algorithm. Given $PK$, a message $M$, and a candidate signature $\sigma$, the algorithm will return 1 if $\sigma$ is a valid signature of $M$. Otherwise, it will return 0.

• **Extract**: The extraction algorithm. Given $k$, the position $P$ of $K$, the algorithm will recover the subliminal message $SM$. Otherwise, it will return 0.

**Definition 2** (Quadratic Residue and Quadratic Non-residue Modulo $p$). Let $p$ be an odd prime, and $0 < a < p$; if $a$ is said to be a quadratic residue (QR) modulo $p$, then $x^2 \equiv a \mod p$, such that $a$ has exactly two square roots modulo $p$. If no such $x$ exists, then $a$ is called a quadratic non-residue (QNR) modulo $p$.

**Definition 3** (Collision-resistant One-way Hash Function). $H$ is a collision-resistant one-way hash function if it is computationally infeasible to derive $x$ from a given hashed value $H(x)$ or to find two different values $x, y$ such that $H(x) = H(y)$.

**Definition 4** (Chinese Remainder Theorem, CRT). If the integers $p_1, p_2, \ldots, p_k$ are pairwise relatively prime, then the system of simultaneous congruences $x \equiv a_i \pmod{p_i}$, where $0 < i \leq k$, has a unique solution $x \pmod{N = p_1p_2 \cdots p_k}$.

### 2.3. Security requirements

According to [37], we present security notions for the proposed schemes. The security notions are **Unforgeability**, **Indistinguishability** and **Inextricability**. They are described as follows:

• **Unforgeability**: Given a valid OTSS, even malicious subliminal receivers cannot forge either the subliminal messages or the signature.

• **Indistinguishability**: Given a valid OTSS, no one can distinguish whether or not the signature contains subliminal messages except the subliminal receivers.

• **Inextricability**: Given a valid OTSS that contains subliminal messages, no one can extract subliminal messages from the signature except the subliminal receivers.

### 3. The proposed schemes

In this section, we will focus on how to embed a subliminal message into the first of Lamport’s OTSSs [35]. Our method also worked for the second of Lamport’s OTSSs [36], but we will not describe this in detail here.

We introduce two schemes, described in Sections 3.1 and 3.2. In our first scheme, to enable sending different messages to different subliminal receivers within the same signature, more than one group of specific secret receivers is required. Note that each of the subliminal messages is a 1-bit one. In our second scheme, we improve upon previous works by enlarging the subliminal message; that is, each of the subliminal messages is an $l$-bit one.

#### 3.1. OTSSs with multiple bits—a subliminal channels scheme

In general, there are three keys in a digital signature with a subliminal channel: the secret key $SK$, the public key $PK$, and the subliminal key $K$. The secret key is used to sign the message; the public key is used to verify the signature; and the subliminal key is used to hide and recover the subliminal message. We describe the details of the five algorithms as follows.

• **Setup**: Let $l$ be a secure parameter and $H$ be a collision-resistant one-way hash function. Note that the each receiver in $R$ knows $j$ his/her own subliminal message sequence position $j$ in the signature. Each of the subliminal receivers $R_1, R_2, \ldots, R_{(n+\log n)/2}$ needs to share a subliminal key $[k_1, k_2, \ldots, k_{(n+\log n)/2}]$ with $S$ in advance. Let $k_i \equiv 3 \pmod{4} \in \{0, 1\}$ be a prime number. In this scheme, we assume that each subliminal channel receiver knows the position $j$ of his/her subliminal message in the signature.

• **KeyGen**: $S$ chooses a candidate secret key $CSK = \{sk_1, sk_2, \ldots, sk_{n+\log n}\}$, where $sk_i \in \{0, 1\}^l$.

• **Embed&Signing**: Let us have the subliminal message $SM = \{sm_1, sm_2, \ldots, sm_{(n+\log n)/2}\}$, where $sm_i \in \{0, 1\}$ as represented by “Yes = 1” or “No = 0”, respectively. Let the signing message be $M = \{m_1, m_2, \ldots, m_{n+\log n}\}$, where $m_i \in \{0, 1\}$. $S$ generates the signature $\sigma$ as follows:

  - If $m_i = 0$, $S$ cannot embed $sm_i$, So $S$ just keeps $sk_i$ and $s_i = H(sk_i)$.
  - If $m_i = 1$, $S$ chooses a new $sk_i$ and lets $s_i = sk_i$ as follows:

    
    $sk_i = \begin{cases} 
    \text{QNR of } k_i, & \text{if } S \text{ wants to embed } sm_i = 0 \\
    \text{QR of } k_i, & \text{if } S \text{ wants to embed } sm_i = 1 
    \end{cases}$

    
    
    Note that $S$ can only embed $sm_i$ in $sk_i$ for the corresponding $m_i = 1$.

Then the signature $\sigma$ of hashed message $M$ is $\sigma = \{s_1, s_2, \ldots, s_{n+\log n}\}$. The corresponding public key is $PK = H(pk_1, pk_2, \ldots, pk_{n+\log n}) = H(H(sk_1), H(sk_2), \ldots, H(sk_{n+\log n}))$. $S$ publishes the $\sigma$ and the $PK$ to the verifier.

• **Verifying**: If $m_i = 0$, then let $y_i = sl_i$. Otherwise, let $y_i = H(sl_i)$. The signature is valid if $H(y_1, y_2, \ldots, y_{n+\log n}) = PK$. 


• **Extract:** Only the subliminal channel receiver \( R_j \) can extract the subliminal message \( sm_j \) using \( k_j \). The subliminal message can be computed as follows:
  - If \( m_i = 0 \), there is no subliminal message in \( s_i \).
  - If \( m_i = 1 \), \( R_j \) computes \( sm_j \) from \( s_i \) using \( k_j \) (where \( s_i = sk_i \)) as follows:
    \[
    sm_j = \begin{cases} 
    0, & \text{if } s_i \text{ is a QNR of } k_j \\
    1, & \text{if } s_i \text{ is a QR of } k_j.
    \end{cases}
    \]

3.1.1. **Extension of the first proposed scheme**

In the first scheme, we can embed more than one subliminal message in the same \( sk_i \) for different subliminal receivers, which are known to have the same position \( j \). However, this will increase the size of the signature.

For example, we want to embed three independent subliminal channels into three secret receivers. Let the receiver set be \( R = \{A, B, C\} \), sharing \( p_A, p_B \) and \( p_C \), respectively, as the subliminal keys, with \( S \).

If the \( j \)-th subliminal message bit is \( sm_j = 0 \) for \( A \), \( sm_j = 1 \) for \( B \) and \( sm_j = 0 \) for \( C \), we choose \( sk_i \) such that it is a QNR of \( p_A \), a QR of \( p_B \) and a QNR of \( p_C \) simultaneously. This \( sk_i \) exists for any \( p_A \equiv p_B \equiv p_C \equiv 3 \pmod{4} \), and it can be computed as follows:

1. Randomly choose \( r \in \mathbb{Z}_N^* \) for \( N = p_A p_B p_C \).
2. Compute \( a = r^2 \mod N \).
3. Compute all eight square roots of \( a \) in \( \mathbb{Z}_N^* \).
4. Exactly one of these eight square roots will satisfy the requirements for \( sk_i \).

We show the possibility that this scheme may hide three independent subliminal channels in \( sk_i \). It is easy to generalize the scheme to hide any number of subliminal messages.

3.2. **OTSSs with multiple blocks—a subliminal channels scheme**

In the second scheme, the size of the subliminal message can be much larger than that of the first one. The signer \( S \) hides the subliminal message in blocks rather than in bits. The block size is determined by a symmetric-key cipher, which is used in the subliminal message encoding.

• **Setup:** As discussed above, let \( l \) be a secure parameter and \( H \) be a one-way hash function. Each receiver in \( R \) knows his/her own sequence position \( j \) in the signature. We assume that the subliminal receiver \( R_j \) and \( S \) share the subliminal key \( k_j \) in advance. Let \( F_{sk} \) be a secure symmetric-key cipher function (e.g., AES, 3DES) and encrypt using key \( k_j \).

• **KeyGen:** \( S \) chooses a candidate secret key \( CSK = \{sk_1, sk_2, \ldots, sk_{n+log n}\} \), where \( sk_i \in \{0, 1\}^l \).

• **Embed&Signing:** Let us have the subliminal message \( SM = \{sm_1, sm_2, \ldots, sm_{n+log n}/2\} \), where \( sm_i \leq l \). Let the signing message be \( M = \{m_1, m_2, \ldots, m_{n+log n}\} \), where \( m_i \in \{0, 1\} \). The signature \( \sigma \) of message \( M \) is \( \sigma = \{s_1, s_2, \ldots, s_{n+log n}\} \), where the \( s_i \)'s are defined as follows:
  - If \( m_i = 0 \), \( S \) cannot embed \( sm_i \). So \( S \) just keeps \( sk_i \) and \( s_i = H(sk_i) \).
  - If \( m_i = 1 \), \( S \) chooses a new \( sk_i \) and lets \( s_i = sk_i \), where \( sk_i = F_{sk}(sm_i) \in \{0, 1\}^l \).

  Note that \( S \) can only embed \( sm_i \) in \( sk_i \) for the corresponding \( m_i = 1 \).

The corresponding public key is \( PK = H(pk_1, pk_2, \ldots, pk_{n+log n}) = H(H(sk_1), H(sk_2), \ldots, H(sk_{n+log n})) \). \( S \) publishes the \( \sigma \) and the \( PK \) to the verifier.

• **Verifying:** If \( m_i = 0 \), then \( y_i = s_i \). Otherwise \( y_i = H(s_i) \). The signature is valid if \( H(y_1, y_2, \ldots, y_{n+log n}) = PK \).

• **Extract:** Only the subliminal channel receiver \( R_j \) can recover the subliminal message \( sm_j \) via \( k_j \). The subliminal message can be computed as follows:
  - If \( m_i = 0 \), there is no subliminal message in \( s_i \).
  - If \( m_i = 1 \), \( R_j \) decrypts \( sm_j \) from \( s_i \) by using the key \( k_j \) (where \( s_i = sk_i \)) as \( sm_j = F_{sk_i}^{-1}(s_i) \).

3.2.1. **Extension of the second proposed scheme**

In the second scheme, assume that there are three subliminal receivers \( A, B \) and \( C \) who can recover the subliminal message. The subliminal receivers \( A, B, C \) and \( S \) need to share the subliminal keys \( k_A, k_B, k_C \) and \( p_A, p_B, p_C \) in advance, where \( p_A, p_B, p_C \) are positive integers that are pairwise relatively prime.

If the \( i \)-th bit of \( M \) is 1, \( S \) embeds the subliminal message in the secret key \( sk_i \). First, \( S \) uses the \( A, B, C \) keys \( k_A, k_B, k_C \) to encrypt the subliminal messages \( sm_{A,i}, sm_{B,i}, sm_{C,i} \) using a symmetric-key cipher function \( F_{sk} \) (e.g., AES, 3DES). \( S \) can obtain \( sk_{A,i} = f_{sk}(sm_{A,i}), sk_{B,i} = f_{sk}(sm_{B,i}) \) and \( sk_{C,i} = f_{sk}(sm_{C,i}) \). Finally, \( S \) uses the Chinese remainder theorem to compute \( sk_i \) from \( sk_{A,i}, sk_{B,i}, sk_{C,i} \) by solving \( sk_i \equiv sk_{A,i} \pmod{p_A} \) for \( sk_i \), where \( R \in \{A, B, C\}, sk_i \in \mathbb{Z}_N^* \) and \( N = p_A p_B p_C \).

We summarize the above procedure as follows:

• If \( m_i = 0 \), \( S \) cannot embed the subliminal message and just chooses \( sk_i \) randomly.
5. Security analysis

In both of our schemes, the SK = {sk1, sk2, ..., skn+log n}, where ski is chosen randomly if mi = 0 and is computed referring to subliminal message smi if mi = 1. The relationship between secret keys of different signatures can be derived only if the attacker A knows SM or K. Hence, the security of signing and verifying in our schemes is the same as that in the original OTSS scheme.
5.1. Unforgeability

**Theorem 1 (Unforgeability of Signature).** If an adversary $A$ can successfully forge a valid signature with non-negligible probability $\varepsilon$ then it will violate the collision-resistant property of the hash function with the same probability $\varepsilon \leq \frac{n + \log n}{2^{n+1}}$.

**Proof.** Assume that there is an attacker $A$ that succeeds in an existential forgery for the proposed schemes using an adaptive chosen message attack with non-negligible probability $\varepsilon$. That is:

1. $A$ runs on input the public key $PK$ and the signature $\sigma$ of OTSS.
2. $A$ adaptively asks for the versions of the message $H(MSG_j) = \{m_1, j, m_2, j, \ldots, m_n, j\}$ for the signing oracle; let the signatures of $H(MSG_j)$ be $\sigma_j$, for $j = 1, \ldots, 2^n$.
3. Eventually, $A$ outputs a valid signature $\sigma^* = \{H(s_1, \ast), H(s_2, \ast), \ldots, H(s_n + \log n, \ast)\}$, satisfying that $\Pr[\text{Validifying}(PK, \sigma^*) = 1 \land \sigma^* \notin \{\sigma_1, \sigma_2, \ldots, \sigma_{2^n} > \}] \geq \varepsilon$.

It is possible to prove that this leads to an algorithm $F$ that finds the collision of a one-way hash function $H$ such that $H(x_i) = H(x'_i)$ with probability $\frac{\varepsilon}{(n + \log n)/2}$. We assume that no adversary can find the collision of the $H$ function such that $H(x_i) = H(x'_i)$ with probability better than $1/2^n$. For the case of a one-time signature scheme, this leads to $\frac{\varepsilon}{(n + \log n)/2} \leq 1/2^n$, which means that one cannot forge signatures with probability better than $\varepsilon \leq \frac{n + \log n}{2^{n+1}}$. We can see that forging a signature in our schemes is no easier than in Lamport’s OTSSs.

In both of our schemes, the subliminal channel receivers’ secret keys $K$ are independent of the $SK$. This implies that the subliminal receivers cannot have any advantage in forging a valid signature. Thus, it will not compromise the security of the OTSS. Moreover, the adversary $A$ or the malicious subliminal receiver $\mathcal{M}$ cannot modify subliminal messages while maintaining the validity of the signature. Because $\mathcal{M}$ has to change the value of the corresponding $sk_i$, the verification will fail because the public key $PK$ cannot be changed. In other words, it is computationally infeasible to find two different values $sk_i, sk'_i$ such that $H(sk_i) = H(sk'_i)$.

We can see that forging a signature in our schemes is no easier than in Lamport’s OTSSs. $\square$

**Theorem 2 (Unforgeability of Subliminal Messages).** Except the subliminal receivers, nobody can reconstruct the subliminal message $SM$ successfully, including malicious subliminal receivers $\mathcal{M}$.

**Proof.** In order to replace the subliminal message $SM$, assume that a malicious subliminal receiver $\mathcal{M}$ may want to guess the other subliminal key $K$. In the proposed schemes, the signature $\sigma$ is computed from $SM$ and $M$. Assume that there exists a trapdoor function that $g_K$ computes with the trapdoor key $K$, denoted as $g_K()$.

Now, let $e = g_K(SM, M)$. If the malicious $\mathcal{M}$ tries to guess $K$, then $\mathcal{M}$ has to collect $e, SM$ and $M$ first. Because the other subliminal message $SM$ is unknown, $\mathcal{M}$ has to try all possible $K$ and fix $SM$ as “meaningful”. It is computationally infeasible to derive the $K$ for replying to the subliminal message $SM$ if the trapdoor function $g_K()$ is secure. $\square$
5.2. Indistinguishability and inextricability of subliminal messages

If the i-th bit of M is 1, then the subliminal message sm is embedded in the secret key sk. The warden will attempt to discover whether there are any hidden subliminal messages. Let keyGen be a subliminal key generation algorithm that takes the security parameter l as input and outputs a secret key sk. Let AES be a secure symmetric encryption function that takes the secret key ki and a subliminal message sm as input, and outputs an encrypted message sk. Let randGen be a random number generation algorithm that takes the security parameter l as input and outputs a secret key sk. Let \( f_{ki}^{-1}(\cdot) \) be a decryption oracle that on input sk returns sm. Let W be a distinguishing algorithm that given sk can distinguish whether it has a hidden subliminal message. Consider the following experiments:

<table>
<thead>
<tr>
<th>Experiment Exp(^{\text{ward-ind-0}}(W)):</th>
<th>Experiment Exp(^{\text{ward-ind-1}}(W)):</th>
</tr>
</thead>
<tbody>
<tr>
<td>( sk_i \leftarrow \text{randGen}(1^l) )</td>
<td>( k_i \leftarrow \text{keyGen}(1^l) )</td>
</tr>
<tr>
<td>( d \leftarrow W^{f_{ki}^{-1}(\cdot)}(sk_i) )</td>
<td>( sk_i \leftarrow \text{AES}(k_i, sm_i) )</td>
</tr>
<tr>
<td>return d;</td>
<td>( d \leftarrow W^{f_{ki}^{-1}(\cdot)}(sk_i) )</td>
</tr>
<tr>
<td></td>
<td>return d;</td>
</tr>
</tbody>
</table>

We let \( \text{Adv}^{\text{ward-ind}}_{f_{ki}^{-1}(\cdot)}(W) = \Pr[\text{Exp}^{\text{ward-ind-0}}(W) = 1] - \Pr[\text{Exp}^{\text{ward-ind-1}}(W) = 1] \) be the advantage of W in attacking \( f_{ki}^{-1}(\cdot) \). Let \( \text{Adv}^{\text{ward-ind}}_{f_{ki}^{-1}(\cdot)}(t) = \max_{ki} \{ \text{Adv}^{\text{ward-ind}}_{f_{ki}^{-1}(\cdot)}(W) \} \) be the advantage of W, defined as the maximum, over all adversaries W that have time-complexity at most t, of the advantage of W in attacking \( f_{ki}^{-1}(\cdot) \).

**Theorem 3** (Indistinguishability and Inextricability of Subliminal Messages). No third party (warden W) can determine whether there are any hidden subliminal messages unless the subliminal receiver picks them up. If a warden W can successfully distinguish a random number sk with a probability greater than non-negligible probability, then it will violate the security of a symmetric encryption AES with the same probability.

**Proof.** In both of our schemes, the warden W cannot decide whether there are any hidden subliminal messages (even when the same subliminal message is sent twice) because the verification steps are the same as those in the OTSS. Even if the warden W assumes that there are hidden subliminal messages in sk, it is still impossible to extract the subliminal messages because ki is unknown, unless the warden W can break the symmetric encryption function, such as AES.

If we have two or more independent subliminal channels for the same sk, in our first scheme, assume that the subliminal message is sent to B. Then, the receiver A cannot obtain B’s subliminal message because A knows ki but not kg. The probability of guessing a correct message bit is \( \frac{1}{2} \) because there are equal numbers of QRs and QNRs in \( Z_{kg} \) when kg is prime. Furthermore, the probability of guessing the l-bit message correctly is negligible. Therefore, no information is revealed on whether sk has a subliminal message.

The above security analysis shows that the probability of determining whether an OTSS hides a subliminal message is negligible assuming a secure encryption function. Therefore, the proposed schemes satisfies the security requirement of indistinguishability and inextricability.

6. Conclusions

In this paper, we have presented two schemes for embedding multiple subliminal messages into OTSSs. Both of our schemes completely achieve multiple subliminal channels without sharing the signer’s secret key. The characteristics of our schemes are summarized below.

- The subliminal receivers do not use a part of the signer’s secret key to embed the subliminal message; this eliminates conspiracy problems.
- The proposed schemes are secure assuming the block cipher (e.g., AES, 3-DES) is secure.
- The security between the OTSS and the subliminal channel is independent.
- We achieve multiple subliminal channels for multiple receivers in both of our schemes.
- In our second scheme, the size of the subliminal channels is much larger than that of the subliminal channels of previous schemes.

Note that our schemes have two important properties as compared to the related ones. First, both of our schemes do not share any part of the signer’s secret keys like those of \([3,4]\); therefore, an attacking method such as conspiracy attack does not work for our schemes. Second, not only the number of subliminal receivers but also the lengths of the subliminal messages are much larger than those for previous schemes \([20,3,4,6,8]\). Consequently, the proposed schemes are more flexible and practical. The comparison between \([20,3,4,6,8]\) and our schemes is shown in Table 1. The other properties and the history of previous works are shown in the Appendix.
Table 1
The comparison between [20, 3, 4, 6, 8] and our schemes.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Share signer's secret key</td>
<td>Forgery attack</td>
<td>Conspiracy attack</td>
<td>Conspiracy attack</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td># of subliminal messages</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>Multiple 1 bits</td>
<td>Multiple blocks</td>
</tr>
<tr>
<td># of subliminal receivers</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>(\approx \frac{(n + \log n)}{2})</td>
<td>(\approx \frac{(n + \log n)}{2})</td>
</tr>
</tbody>
</table>

Fig. A.1. Overview of subliminal channel-related publications from 1983–2009.

Acknowledgements

The authors would like to thank the anonymous referees of the paper for their valuable comments.

Appendix. Overview of subliminal channel-related publications from 1983–2009

See Fig. A.1.

References