Attacks on a Strong Designated Verifier Proxy Signature Scheme

Caixue Zhou

School of Information Science and Technology, University of Jiujiang, JiuJiang, 332005, JiangXi, P.R. China
charlesjjjx@126.com

Abstract. Strong designated verifier proxy signature (SDVPS) enables a proxy signer to sign messages on behalf of an original signer, and only the designated verifier can be convinced of the validity of the proxy signature, and the verifier cannot prove it to any third party. It is a combination of proxy signature and strong designated verifier signature. Recently, Lin et al proposed a SDVPS scheme and proved it to be unforgeable in the random oracle model. In this paper, we show their scheme is insecure and give four attacks to their scheme.

Keywords: Designated Verifier Signature, Proxy Signature, Forgery Attack, Random Oracle Model, Discrete Logarithm Assumption

1 Introduction

In 1996, Mambo et al. [1] introduced a new concept of proxy signature, in which an original signer can delegate his signing rights to a proxy signer, and then the latter can sign messages on behalf of the former. Since then, many proxy signature schemes have been proposed. Proxy signature is found to have numerous practical applications, such as in grid computing [2], in mobile agents [3] et al.

To categorize delegation types, Mambo, et al. defined three levels of delegation: (1) Full delegation. The original signer gives his secret key to the proxy signer. The proxy signer uses the key to sign documents. (2) Partial delegation. The original signer generates a delegation key from its private key and gives it to the proxy signer. The proxy signer generates a new proxy signature key from the proxy signer’s private key and the delegation key. (3) Delegation by warrant. The original signer signs a warrant which describes relative rights and information of the original signer and the proxy signer. The final proxy signature includes two parts: one is the signed warrant, and another is the proxy signature produced by the proxy signer. Proxy signature can be constructed for each of these delegation types, and the most suitable scheme should be selected depending on the user’s need for security, message length and signer’s and/or verifier’s computational ability. The partial delegation and the delegation by warrant are more secure than the full delegation because the created proxy signature is distinguishable from the original signer’s signature. The partial delegation is faster than the delegation by warrant, because in the delegation by
warrant, the verifier must verify two signatures. Due to its merits of partial delegation, Mambo et al. refer to partial delegation proxy signature as proxy signature.

However, in partial delegation, the proxy signer can abuse his delegated rights because partial delegation does not restrict the proxy signer’s signing capability. To overcome this weakness, in 1997, Kim [4] introduced a new kind of proxy signature called partial delegation with warrant. This kind of proxy signature combines the benefits of both the partial delegation and the delegation by warrant, so this delegation has fast processing speed and is appropriate for the restricting documents to be signed. Since then, most work on proxy signature has focused on the partial delegation with warrant proxy signature. In this paper, we simply refer to partial delegation with warrant proxy signature as proxy signature if it does not lead to any confusion.

In Eurocrypt 1996, Jakobsson et al. [5] introduced another new concept of designated verifier signature (DVS). Ordinary signature can convince any person of its validity, while in DVS scheme, only the designated person can be convinced of the validity of the signature, and he can not prove this to any third party, because he can also produce an indistinguishable transcript of the signature. DVS scheme provides authentication of a message without having the non-repudiation property of ordinary signature. DVS scheme is very suitable in the scenario where a signer wishes to keep privacy of his identity to other parties but not to the designated verifier. For example, if a software vendor puts digital signature on its products to allow it to authenticate them as correct, free of viruses, etc., but only wants paying customers to be able to verify the validity of the signature, then DVS schemes can be used. On one hand, when a paying customer receives the signature produced by the software vendor, he can be convinced of the validity of the signature if the signature verification is OK, because he knows he himself do not produce the signature; on the other hand, the paying customer can produce an indistinguishable transcript of the signature, so other persons can not be convinced of its validity, which prevents the paying customer from reselling the software products to other persons.

However, DVS scheme has a weakness that any person can verify it, and can make sure there are only two potential signers. Hence, if the signature is eavesdropped before arriving at its destination, then one can identify the signer. Later, in 2003, Saeednia et al. [6] introduced the concept of strong designated verifier signature (SDVS) to overcome this weakness by forcing the designated verifier to use his private key at the time of verification.

To combine the merits of proxy signature and designated verifier signature, Dai et al. [7] in 2003 proposed a designated verifier proxy signature (DVPS) scheme, in which a proxy signer can produce a DVPS on behalf of an original signer, and only the designated verifier can be convinced of the validity of the proxy signature, and he cannot prove it to any third party. Take the above software vendor for example, a company agent can be delegated by the software vendor to sign with the customer using a DVPS scheme. After Dai et al.’s work, Wang [8] in 2004 pointed out Dai et al.’s scheme is not secure and improved it. Li et al. [9] proposed a DVPS scheme from bilinear pairings. Later, in 2005, Wang [10] proposed another two DVPS schemes. Huang et al. [11] proposed a short DVPS scheme from bilinear pairings, and they gave the security model of a DVPS scheme. Lu et al. [12] first proposed a DVPS scheme with message recovery. Cao et al. [13] first proposed three identity-based
DVPS schemes. Since then, many other DVPS and SDVPS schemes have been proposed [14-17].

In 2012, Lin et al [18] proposed a SDVPS scheme. Compared with related schemes, their scheme has not only shorter signature length, but also lower computational costs, which is very suitable for electronic commerce, and they proved their scheme is unforgeable in the random oracle model under discrete logarithm assumption. In this paper, we show four attacks on their scheme.

The rest of the paper is organized as follows. In section 2, we review Lin et al.’s SDVPS scheme. In section 3, we give out four attacks on their scheme. We conclude our work in section 4.

2 Lin et al.’s Strong Designated Verifier Proxy Signature Scheme

Lin et al.’s scheme consists of the following four phases.

Setup: Taking as input \( k \), the system authority (SA) selects two large primes \( p \) and \( q \) where \( q = k \) (mod \( p - 1 \)). Let \( g \) be a generator of order \( q \) and \( h_i : (0,1)^* \rightarrow Z_q \), \( h_i : (0,1)^* \rightarrow Z_q \), \( h_i : Z_q \rightarrow Z_q \) be three collision resistant hash functions. The system public parameters are \( \text{params} = (p, q, g, h_1, h_2) \). Each user \( U \) chooses his private key \( s_i \in Z_q \) and computes his public key as \( y_i = g^{s_i} \).

- Proxy Credential Generation: Let \( U_0 \) be an original user delegating his signing power to a proxy signer \( U_p \). \( U_0 \) first chooses \( q_R \in Z_q \) to compute
  \[
  q_p g T \equiv d \pmod{p - 1} \quad \text{(1)}
  \]
  \[
  q T m h x d \equiv w_0 \pmod{q} \quad \text{(2)}
  \]
  Where \( w_0 \) is a warrant consisting of the identifiers of original and proxy signer, the delegation duration and so on. \( (\sigma, m_w, T) \) is then sent to \( U_p \). Upon receiving \( (\sigma, m_w, T) \), \( U_p \) computes \( Z \) as Eq. (3) and performs Eq. (4) to check its validity.
  \[
  Z = y_0^{\text{w}_0} \pmod{p} \quad \text{(3)}
  \]
  \[
  \sigma' = \text{Z}^q \pmod{p} \pmod{q} \quad \text{(4)}
  \]
  If it does not hold, \( (\sigma, m_w, T) \) is requested to be sent again.

- Proxy Signature Generation: For signing a message \( m \in (0,1)^* \) on behalf of the original signer \( U_0 \), \( U_p \) chooses \( w \in Z_q \) to compute
  \[
  s_i = h_i(\text{w}^{\sigma}) \pmod{q} \quad \text{(5)}
  \]
  and then delivers \( (m, m_w) \) along with the SDVPS \( \delta = (s_i, s_j, T) \) to a designated recipient \( U_v \).

- Proxy Signature Verification: Upon receiving \( (m, m_w) \) and \( \delta = (s_i, s_j, T) \), \( U_v \) first computes \( (r_1, r_2) \) as follows:
3 Cryptanalysis of Lin et al’s Scheme

Lin et al. proved that their scheme is existentially unforgeable under adaptive chosen message attack assuming the discrete logarithm is hard. However, we found this is not the fact. In the following, we give four attacks to their scheme.

**Attack1:** The original signer \( U_s \) can forge a valid proxy signature. First, \( U_s \) intercepts a valid proxy signature \((m, m_s, s_i, s_i, T)\) generated by \( U_r \) on behalf of \( U_s \), then \( U_s \) generates a new warrant \( m_s \), computes \( \sigma = x \cdot h_q(m_s, T) \), computes \( s_i = h_i((R, R_i, \mod p) \mod q) \). If it holds, the SDVPS \( \delta = (s_i, s_i, T) \) for \( m \) is valid.

\[
R_1 = y_i^{r_1} \mod p \quad \quad (7)
\]
\[
R_2 = (T y_i^{-h_q(m_s, T)} s_i h_s(m_s, T)) \mod p \quad \quad (8)
\]
\[ u \text{ then verifies the proxy signature by checking if } \]
\[
s_i = h_i((R, R_i, \mod p) \mod q) \quad \quad (9)
\]

**Attack2:** An adversary \( U_r \), first forges a warrant \( m_r \), which records himself being the proxy signer, then he chooses two random numbers \( t, w \in Z_q \), and computes \( T = (y_t)^{s_2} \mod p \), \( s_i = h_i((y_t)^{s_2} \mod p) \mod q \), \( s_2 = w - (s_t \cdot h_i(m_r, T)) h_i(m_r, T) \mod q \). Finally, the forged proxy signature is \((m, m_r, s_i, s_i, T)\), and the \( U_r \) is the original signer, and the \( U_r \) is the proxy signer. The following shows \((m, m_r, s_i, s_i, T)\) can pass the verification equation. The designated receiver \( U_r \), computes \( R_1 = y_i^{r_1} \mod p \), \( R_2 = (T y_i^{-h_q(m_r, T)} s_i h_s(m_r, T)) \mod p \), then

\[
R_1 R_2 = y_i^{r_1} (T y_i^{-h_q(m_r, T)} s_i h_s(m_r, T)) \mod p = y_i^{r_1} (g^{r_1} s_i h_s(m_r, T)) \mod p = y_i^{r_1} \mod p \quad \quad (10)
\]
SO \( s_i = h_i((y_t)^{s_2} \mod p) \mod q) = h_i((R, R_i) \mod p) \mod q) \).

**Attack3:** An adversary \( U_r \), first forges a warrant \( m_r \), which records himself being the original signer and \( U_r \) being the proxy signer, then he chooses three random number \( w, t, d \in Z_q \), and computes \( T = g^t \mod p \), then he registers
his public key in CA as \( y_x = (y_z^{g_x})^{-1} \mod p \), and computes proxy signature as follows, \( s_i = h_i((y_z^{\mod p}) \mod q) \), \( s_j = w - (d^i + t)h_j((m,T) \mod q) \). Finally the forged proxy signature is \((m,m',s_x,s_y,T)\), and the \( U_x \) is the original signer, the \( U_y \) is the proxy signer. The following shows \((m,m',s_x,s_y,T)\) can pass the verification equation.

The designated receiver \( U_x \), computes \( R_x = y_x^2 \mod p \), then
\[
R_1 = (T_y \cdot y_x^{s_x(s_x-s_y)} \cdot y_x^{s_y}) \mod p ,
\]
\[
R_2 = y_x^{s_x(s_x-s_y)} \cdot y_x^{s_y} \mod p .
\]

**Attack4:** An adversary \( U_x \) first forges a warrant \( m' \), which records \( U_x \) being the original signer and himself \( U_y \) being the proxy signer, then he chooses three random numbers \( w, t, d \in Z, \) and computes \( T = g^d \mod p \mod q \), then he registers his public key in CA as \( y_x = y_z^{h_x}g^d \mod p \), and computes proxy signature as follows, \( s_i = h_i((y_z^{\mod p}) \mod q) \), \( s_j = w - (d^i + t)h_j((m,T) \mod q) \). Finally, the forged proxy signature is \((m,m',s_x,s_y,T)\), and the \( U_x \) is the original signer, the \( U_y \) is the proxy signer. The following shows \((m,m',s_x,s_y,T)\) can pass the verification equation.

The designated receiver \( U_x \), computes \( R_x = y_x^2 \mod p \), then
\[
R_1 = (T_y \cdot y_x^{s_x(s_x-s_y)} \cdot y_x^{s_y}) \mod p ,
\]
\[
R_2 = y_x^{s_x(s_x-s_y)} \cdot y_x^{s_y} \mod p .
\]

**Discussion:** The above four attacks show designing a secure SDVPS scheme is not an easy task. The main reasons of Lin et al.’s scheme being insecure are that (1) the proxy signer’s secret key does not acting on the warrant \( m_x \); (2) the public keys of original signer and proxy signer are not including in the hash function, respectively; (3) There is a linear relationship between the variable \( T \) and the proxy signer

4 Conclusion

Designated verifier proxy signature can combine the merits of proxy signature and designated verifier signature. Recently, Lin et al. proposed a SDVPS scheme, which has not only shorter signature length, but also lower computational costs. In this paper, we point out their scheme is forgeable by giving four attacks. The future work is to design lattice-based proxy signature or proxy signature in multivariate public key cryptography.
Acknowledgments. This work is supported by the Key Program of Jiujiang University under Grant No. 2013ZD02.

References