Information Sciences 281 (2014) 365–372

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

On delegatability of designated verifier signature schemes

Kyung-Ah Shim

National Institute for Mathematical Sciences, KT Daedoek 2nd Research Center, 463-1, Jeonmin-dong, Yuseong-gu, Daejeon, Republic of Korea

ARTICLE INFO

Article history:
Received 26 November 2008
Received in revised form 30 April 2014
Accepted 7 May 2014
Available online 4 June 2014

Keywords:
Delegatability
Designated verifier signature
Digital signature
Identity-based system
Strong designated verifier signature
Universal designated verifier signature

ABSTRACT

Lipmaa et al. introduced a new security notion of designated verifier signature schemes, non-delegatability: neither a signer nor a designated verifier can delegate the signing rights to any third party without revealing their secret keys. In this paper, we classify designated verifier signature schemes into three types and then discuss delegatability of existing designated verifier signature schemes, strong designated verifier signature schemes and universal designated verifier signature schemes, and open research issues.

1. Introduction

Jacobson et al. [6] introduced the notion of designated verifier signatures. Designated verifier signature (DVS) schemes provide authentication of a message, without having the non-repudiation property of traditional signatures: they convince one and only one specified recipient that they are valid, but unlike standard digital signatures, nobody else can be convinced about their validity or invalidity. The reason is that the designated verifier in these schemes is able to create a signature intended to himself that is indistinguishable from a real signature. These signatures have several applications such as e-voting, call for tenders and software licensing. Steinfeld et al. [22] extended notion of the DVS schemes to universal DVS (UDVS) schemes which allow a signature holder to convert a standard signature into a designated signature specified to any designated verifier of his choice. Jakobsson et al. [6] proposed a stronger notion called a strong designated verifier signature (SDVS) scheme in which a designated verifier uses his secret key to verify the validity or invalidity of a signature designated ti himself. After then, a number of DVS, SDVS, and UDVS schemes have been proposed [21,6,19,13,4,23]. Lipmaa et al. [17] introduced a new security notion of designated verifier signature schemes, non-delegatability. DVS scheme is delegatable if either a signer or a designated verifier can delegate the signing rights (either with respect to a concrete designated verifier or with respect to all designated verifiers) to some third party without disclosing its secret key. Although revealing the signing rights does not mean revealing the secret key, delegatability, especially with respect to a concrete designated verifier, is highly undesirable in many applications. For example, in an e-voting scenario where a voter signs messages by using a delegatable DVS scheme (with the tallier being the designated verifier), one voter can delegate its voting right to a coercer that can then vote instead of the voter. Therefore, such an e-voting protocol is coercible. Moreover, in many e-commerce applications, one can use a DVS scheme so that the signer is a subscriber to an e-service provided by a service provider who is the designated verifier. If the DVS scheme is delegatable, the signer can send some delegation token to a

E-mail address: kashim@nims.re.kr

http://dx.doi.org/10.1016/j.ins.2014.05.020
0020-0255/

non-subscriber who can then enjoy the service for free. Thus, it is guaranteed that in the schemes, if an adversary can create a valid signature, it knows the secret key of the signer or the verifier. A number of DVS schemes turned out to be delegatable [16,11,18,23]. In this paper, we classify designated verifier signature schemes into three types and then discuss delegatability of existing designated verifier signature schemes, strong designated verifier signature schemes and universal designated verifier signature schemes, and open research issues.

The rest of the paper is organized as follows. In Section 2, we describe definitions of DVS and UDVS schemes. In Section 3, we divide DVS schemes into three types and describe types I and II DVS schemes. In Section 4, we present delegatability attacks on the DVS, SDVS, and UDVS schemes in [9,29,7,15,20,30,3,22,14,5,4,8,2,1,12,21,10,26]. Concluding remarks are given in Section 5.

2. Definitions of designated verifier signature schemes

We describe the definitions of SDVS schemes and UDVS schemes [12,4,26,10,27,21].

Components of SDVS Schemes. A SDVS scheme $\mathcal{SDVS} = (\textbf{KeyGen}, \textbf{DV}-\textbf{Sign}, \textbf{DV}-\textbf{Verify}, \textbf{Transcript Simulation})$ is specified by the following four polynomial time algorithms:

KeyGen. Randomized key generation algorithm **KeyGen** takes input 1^k, where $k \in \mathbb{Z}^+$ is a security parameter, and outputs a secret/public key pair (sk, pk).
DV-Sign. Signing algorithm **DV-Sign** takes input the secret key sk_S of a signer, the public key pk_V of a designated verifier and a message m, and outputs a designated verifier signature τ, where $\tau \leftarrow \textbf{DV}-\textbf{Sign}(sk_S, m, pk_V)$.
DV-Verify. Verification algorithm **DV-Verify** takes input the secret key sk_V of a designated verifier, the public key pk_S of a signer, a message m and a designated verifier signature τ on m, and outputs `True` if $\textbf{DV}-\textbf{Verify}(sk_V, m, pk_S, \tau) = 1$, otherwise, $\perp$.
Transcript Simulation. Via the **Transcript Simulation** algorithm, a designated verifier, who holds its private key sk_V can always produce identically distributed transcripts that are indistinguishable from the original proof.

Component of ID-based DVS schemes. An ID-based DVS scheme $\mathcal{IBDVS} = (\textbf{Setup}, \textbf{Extract}, \textbf{DV}-\textbf{Sign}, \textbf{DV}-\textbf{Verify}, \textbf{Transcript Simulation})$ is specified by the following five polynomial time algorithms:

Setup. Randomized parameter generation algorithm **Setup** takes input 1^k, where $k \in \mathbb{Z}^+$ is a security parameter, and outputs publicly known system parameters `params` including a master public/secret key pair (mpk, msk).
Extract. Private key extraction algorithm **Extract** takes input an identity ID and a master secret msk, and outputs a private key $S_{ID} \leftarrow \textbf{Extract}(msk, ID)$.
DV-Sign. Signing algorithm **DV-Sign** takes input the private key S_{ID_S} of a signer, the identities ID_S and ID_V of a signer and a designated verifier, respectively, and a message m, and outputs a designated verifier signature τ, where $\tau \leftarrow \textbf{DV}-\textbf{Sign}(S_{ID_S}, m, ID_S, ID_V)$.
DV-Verify. Verification algorithm **DV-Verify** takes input the private key S_{ID_V} of a designated verifier, ID_S, ID_V, a message m and a designated verifier signature τ on m, and outputs `True` if $\textbf{DV}-\textbf{Verify}((S_{ID_V}, m, ID_S, ID_V, \tau) = 1$, otherwise, $\perp$.
Transcript Simulation. Via the **Transcript Simulation** algorithm, a designated verifier, who holds its private key S_{ID_V} can always produce identically distributed transcripts that are indistinguishable from the original proof.

The UDVS scheme allows any holder of a standard signature to designate the signature to any verifier. The designated verifier can check that the message was signed by the signer, but is unable to convince anyone else of this fact.

Components of UDVS Schemes. A UDVS scheme $\mathcal{UDVS} = (\textbf{KeyGen}, \textbf{Sign}, \textbf{Verify}, \textbf{Designation}, \textbf{U}-\textbf{DV}-\textbf{Verify}, \textbf{Transcript Simulation})$ based on a standard signature scheme $\mathcal{SS} = (\textbf{KeyGen}, \textbf{Sign}, \textbf{Verify})$, is specified by six polynomial time algorithms with the following functionality:

KeyGen, Transcript Simulation. These algorithms are the same as those in the DVS schemes.
Sign. Standard signing algorithm **Sign** takes input a signer's secret key sk_S and a message m, and outputs a standard signature $\sigma \leftarrow \textbf{Sign}(sk_S, m)$.
Verify. Standard verification algorithm **Verify** takes input a signer's public key pk_S and a standard signature σ under pk_S on a message m, and outputs `True` if $\textbf{Verify}(pk_S, m, \sigma) = 1$, otherwise, $\perp$.
Designation. Designation algorithm **Designation** takes input a signer's public key pk_S, a designated verifier's public key pk_V and a standard signature σ on a message m under pk_S, and outputs a designated verifier signature $\tau \leftarrow \textbf{Designation}(pk_S, pk_V, m, \sigma)$.
U-DV-Verify. Verification algorithm **U-DV-Verify** takes input a signer's public key pk_S, a designated verifier's public/secret key pair (pk_V, sk_V) and a designated verifier signature τ on a message m, and outputs `True` if $\textbf{U}-\textbf{DV}-\textbf{Verify}(sk_V, pk_S, pk_V, m, \tau) = 1$, otherwise, $\perp$.