

# EFFICIENT DESIGNATED VERIFIER SIGNATURE SCHEMES

Shahrokh Saeednia, Steve Kremer, Olivier Markowitch  
Université Libre de Bruxelles  
Bd du Triomphe - CP212  
1050 Bruxelles, Belgium  
{saeednia,skremer,omarkow}@ulb.ac.be

*This paper proposes a designated verifier signature based on the Schnorr signature scheme. One of the advantages of the new scheme compared with the one proposed by Jakobsson, Sako and Impagliazzo is that not only the designated verifier (Bob) cannot convince a third party (Cindy) that a signature is originated by a given signer (Alice), but also nobody else other than Bob can even check the validity or invalidity of such a signature without the intervention of Bob.*

*Other advantages of our scheme are the low computational cost and the small size of the resulting signature. Generating a signature requires only 1 modular exponentiation, while the verification needs 2 modular exponentiations.*

## INTRODUCTION

In 1996, Jakobsson, Sako and Impagliazzo [2] and Chaum [1] introduced designated verifier signatures and private signatures, respectively, that are based on the same idea. In the following, we only focus on Jakobsson et al.'s scheme, as it resulted in an academic publication. Designated verifier signatures provide authentication of a message, without however providing non-repudiation. They have the property that they convince one specified recipient that they are valid, but unlike other 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 in an indistinguishable way. Therefore, when Bob receives a signature from Alice, he will certainly trust that it is originated from Alice upon verifying it, since he knows that he has not generated it himself. However, another user, Cindy, has no reason to accept such a signature as Alice's one, because she knows that Bob is fully capable to produce it himself.

Designated verifier signatures are very useful in situations where the signer of a message should be able to specify who may be convinced by his/her signature. Let us consider the following example.

Suppose that a public institution initiates a call for tenders, asking some companies to propose their prices for a set of instruments and tasks to be accomplished. The institution may require the companies to sign their offers in order to make sure that they are actually authentic and originated from whom they claim to be. This is a valid requirement, but no company involved in this process desires its offer to affect other tenderers' decisions. That is, a company may capture the signed offer of a competitor on the transmission line (to the institution) and prepares its offer consequently in order to increase its chance to be selected by the institution.

To prevent this, the companies may obviously encrypt their offers and signatures in order that they may only be read and verified by the institution. But, nothing prevents the latter to reveal them once decrypted. Indeed, since the institution's goal is to obtain a good price (as low as possible), it could show some signed offers to some other companies to influence them in making "good" offers.

So, the here raised question is about the conflict between authenticity and privacy. Designated verifier signature is a solution to this problem. With such signatures, while the institution is convinced about the origin and the authenticity of an offer, it cannot transfer the *conviction* to others.

**Related works.** To the best of our knowledge, only three schemes providing the designated verifier property exist. Because of page limitation, we do not give here their detailed description. The first schemes are proposed independently by Jakobsson et al. [2] and Chaum [1]. These are historically the first designated verifier signature schemes, but are computationally rather inefficient. For instance, Jakobsson et al.'s scheme requires 5 modular exponentiations for generation, as well as verification.

More recently, in 2001, Rivest, Shamir and Tauman introduced ring signatures [3]. Their scheme allows to generate a signature, with a group of potential signers. Setting the size of the group to two, yields a designated verifier scheme, that is very efficient. It merely requires 1 modular exponentiation at the generation, and only 4 modular multiplication to verify the signature. However, the exponentiation is computed with respect to a large RSA modulus, which results in a scheme that is less efficient than the one proposed in this paper.

**Strong designated verifier.** Although designated verifier signatures are *signer ambiguous*, in the sense that one cannot verify whether the real signer or the designated verifier issued the signature, they remain universally verifiable, i.e. everyone can convince himself that there are only two possible signers. Hence, considering again the example of the institution making a call of tenders, if the companies' offers are sent just being signed using a designated verifier signature, these signatures may be captured on the line, before arriving at the institution and one can identify the signer, because one knows that the institution could not fake the signature. One possible solution, that is however expensive in terms of computation, is to publicly encrypt each signature. This stronger requirement, called *strong designated verifier*, was briefly discussed in the appendix of [2].

In this paper, we introduce a new designated verifier scheme, based on the Schnorr signature scheme [4]. This new scheme provides the strong designated verifier property at no additional cost. The scheme is extremely efficient, as it merely needs 1 modular exponentiation to generate the signature, and 2 modular exponentiations in order to verify it, i.e. no additional exponentiation is required with respect to the original Schnorr signature. All previously presented schemes required an additional public-key encryption, in order to become strong. Moreover our new scheme provides signatures that are very small in size.

## DEFINITIONS

**Designated verifier proofs.** Our goal is to allow Alice proving the validity of a statement  $\Omega$  to Bob in such a way that, while Bob is convinced of this fact, he cannot transfer this conviction to other people.

As suggested in [2], when Alice wants to convince Bob—and only Bob—of the truth of the statement  $\Omega$ , she should prove the statement “ $\Omega \vee$  I know Bob's secret key”. Bob, who is aware that he has not prepared the proof himself and knows that Alice does not know his secret key, will accept the validity of the first part of the statement (i.e.,  $\Omega$ ) while no other verifier will be able to decide which part of the disjunction is true. Definitions of designated verifier proofs are given in [2]. We believe that these definitions, though completely persuasive, do not fully capture our intuition of the designated verifier proofs and therefore propose new alternate definitions here.

We can define the designated verifier property in the following way. If Bob, after having received a proof (signature) from Alice, has a way to prove to Cindy the truth of a given statement, then he can produce indistinguishable transcripts

by his own. As a consequence, whatever Bob can do with the “real” transcripts, he will be able to do with the “simulated” transcripts as well. Thus, Cindy being aware of this fact, will never be convinced by Bob’s proof, whatever the protocol that Bob initiates.

Put in more formal words, we can define designated verifier proofs as follows:

**Definition 1** *Let  $P(A, B)$  be a protocol for Alice to prove the truth of the statement  $\Omega$  to Bob. We say that Bob is a designated verifier if he can produce identically distributed transcripts that are indistinguishable from those of  $P(A, B)$ .*

**Strong designated verifier proofs.** In some circumstances, Cindy may be convinced with high probability that a designated verifier proof intended to Bob is actually generated by Alice, as Bob would not or could not generate it himself. For example:

1. When Bob is believed to be honest, Cindy would trust that Bob does never deviate from his prescribed protocol, so that by seeing a signature, she would be convinced that it is originated by Alice.
2. When Cindy is sure that Bob has not yet seen a signature intended to himself, she would be convinced that the signature is not “forged” by Bob.

In all these cases, we need a stronger notion of designated verifier proofs. We say that a proof is *strong designated verifier* if transcripts of a “real” proof may be simulated by anybody in such a way that they are indistinguishable for everybody other than Bob. So, accordingly to our definition of designated verifier proofs, we define the strongness as follows:

**Definition 2** *Let  $P(A, B)$  be a protocol for Alice to prove the truth of the statement  $\Omega$  to Bob. We say that  $P(A, B)$  is a strong designated verifier proof if anybody can produce identically distributed transcripts that are indistinguishable from those of  $P(A, B)$  for everybody, except for Bob.*

**Remark.** When Bob can prove to Cindy that he doesn’t know his secret key (for example by showing that his public key is of the form  $f(\text{identity})$ , where  $f$  is a one way hash function), then Cindy, by seeing a signature, is convinced that it is originated from Alice since Bob could not simulate the transcripts. Another similar problem arises when Bob and Cindy secretly share Bob’s signature secret key. Bob has to cooperate with Cindy when verifying Alice’s signatures intended

to him. While the first problem does not apply to our scheme, it seems very difficult to overcome the second one. However, both problems are solved when assuming that the signature public keys are certified by an authority, that verifies that a given user actually knows his secret key corresponding to the certified public key.

## DESCRIPTION OF THE SCHEME

As is the case in all DL based schemes, we assume that some common parameters are initially shared between the users: a large prime  $p$ , a prime factor  $q$  of  $p - 1$ , a generator  $g \in Z_p^*$  of order  $q$  and a one-way hash function  $h$ .

Each user  $i$  chooses his secret key  $x_i \in Z_q$  and publishes the corresponding public key  $y_i = g^{x_i} \pmod{p}$ .

In order to sign a message  $m$  for Bob, Alice selects two random values  $k$  and  $t$  in  $Z_q$  and computes

$$\begin{aligned} c &= y_b^k \pmod{p}, \\ r &= h(m, c), \\ s &= kt^{-1} - rx_a \pmod{q}. \end{aligned}$$

The triple  $(r, s, t)$  is then the signature of the message  $m$ .

Knowing that a signature is originated from Alice, Bob may verify its validity by checking whether  $h(m, (g^s y_a^r)^{tx_b} \pmod{p}) = r$ .

As we can see, nobody else other than Bob can perform this verification, since his secret key is involved in the verification equation. This precisely means that our scheme verifies the strong designated verifier property. Hereafter, we show that even if Bob reveals his secret key, he cannot convince another party, say Cindy, of the validity of such a signature.

Indeed when Cindy is given Bob's secret key, she can certainly check the consistency of the signature in the same way as Bob. But, there is no reason that she accepts it as an Alice's signature, because Bob is capable to generate the same transcripts in an indistinguishable way. To do so, Bob may select  $r'$  and  $s'$

at random and compute

$$\begin{aligned}
c &= g^{s'} y_a^{r'} \pmod{p} \\
r &= h(m, c) \\
\ell &= r' r^{-1} \pmod{q} \\
s &= s' \ell^{-1} \pmod{q} \\
t &= \ell x_b^{-1} \pmod{q}.
\end{aligned}$$

Then  $c = (g^s y_a^r)^{tx_b} \pmod{p}$  and  $h(m, c) = r$ . In fact

$$\begin{aligned}
(g^s y_a^r)^{tx_b} \pmod{p} &= \\
(g^s y_a^r)^\ell \pmod{p} &= \\
g^{s\ell} y_a^{r\ell} \pmod{p} &= \\
g^{s'} y_a^{r'} \pmod{p} &= c
\end{aligned}$$

and  $h(m, c) = r$  by definition.

**Remarks:**

1. To be complete, let us notice that Cindy should start by checking whether the received secret key is actually Bob's one ( $g^{x_b} \pmod{p} \stackrel{?}{=} y_b$ ), because without it she is not even convinced that the signature is made by "Alice or Bob", as anybody may have simulated Alice's signature (intended to himself) and give his secret key to Cindy as the Bob's one.
2. Instead of revealing his secret key, Bob can prove to Cindy the consistency of a signature (and not that it is originated by Alice) as follows. Bob presents  $(m, s, t, c)$  to Cindy. Then Cindy computes  $r = h(m, c)$  and asks to Bob to prove the knowledge of  $x_b$  as the discrete logarithm of  $y_b$  in one hand and  $c$  on the other hand with respect to  $g$  and  $(g^s y_a^r)^t \pmod{p}$  as the bases, respectively.

Because of lack of space, we do not give a security proof of our scheme and leave it for an extended version of this paper.

**COMPARISONS**

In this section we give a performance comparison of our new scheme and the two existing schemes, namely Jakobsson et al's one (JSI for short) and Rivest

	JSI	Strong JSI	RST	Strong RST	New scheme
Generation	1,200	1,200	768	768	240
Verification	1,200	1,968	0	768	480
Total	2,400	3,168	768	1,536	720
Size (bits)	2,368	2,880	1,536	2,048	480

Table 1: Performance and size comparison

et al.’s one (RST for short). For this comparison we choose an implementation, setting  $p = 512$  bits and  $q = 160$  bits for the JSI and our scheme. In order to have comparable security, we set the RSA modulus to 512 bits in the RST scheme.

For the comparison to be effective, we only counted the number of modular exponentiations, which are the most significant operations, and neglect other operations such as hashing, modular multiplications and symmetric encryptions. We also suppose that when using RSA, the public-key is set to 3, which allows one exponentiation to be replaced by two modular multiplications that are considered negligible.

In table 1, we indicate the complexity—in terms of modular multiplications resulting from modular exponentiations—of the two existing schemes in their strong and not strong flavours, as well as our new scheme. We assume that an exponentiation is equivalent to  $1.5 \times \log(m)$  modular multiplications, where  $m$  is the exponent. In order for the JSI and the RST schemes to provide the strong designated verifier property, they need to be encrypted. We assume that a session key is encrypted using 512 bit RSA public-key encryption. This session key can then be used to cipher the transcripts. We can see in table 1 that our scheme is much more efficient than the JSI scheme for both generation as well as verification. One may also see that the verification in the RST scheme is the most efficient one. However this is not crucial for designated verifier signatures. In traditional signature schemes, efficient verification is a desirable design issue, motivated by the fact that a signature is generated once, but may be verified many times. In designated verifier schemes, there exists only one verifier, which implies only one verification. Therefore we argue that, for designated verifier schemes, only the total computational cost, regrouping signature generation and verification, is significant. When considering the total computing amount our scheme is slightly more efficient than the RST scheme in the normal case, and more than twice more efficient in the case of strong designated verifier.

We also compared the size of the respective signatures, assuming that the hash function's output is of size 160 bits. Table 1 shows that our new scheme provides significantly smaller signatures than both existing ones.

## CONCLUSION

In this paper, we proposed a new designated verifier signature scheme. To the best of our knowledge, it is the first scheme providing directly the strong designated verifier property, without any additional encryption. Moreover, our scheme is more efficient than the existing ones in terms of both computation and communication complexity.

## REFERENCES

- [1] David Chaum. Private signature and proof systems. United States Patent 5,493,614, 1996.
- [2] Markus Jakobsson, Kazue Sako, and Russell Impagliazzo. Designated verifier proofs and their applications. In Ueli Maurer, editor, *Advances in Cryptology—EUROCRYPT 1996*, volume 1070 of *Lecture Notes in Computer Science*, pages 143–154. Springer-Verlag, May 1996.
- [3] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In Colin Boyd, editor, *Advances in Cryptology—ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*. Springer-Verlag, December 2001.
- [4] Claus P. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, 1991.