Designated Verifier Signature Schemes - An Overview

Liina Kamm

October 10, 2005

Structure

- The Jakobsson-Sako-Impagliazzo Scheme
- Security notions for DVS schemes
- DVS Scheme with tight reduction to DDH in NPRO
- Universal Designated Verifier Signature Scheme Without RO
- Security notions for UDVS

Problem statement

- Alice wants to prove Θ to Bob.
- Alice wants to prove Θ only to Bob.
- Cindy cannot know what was proved by Alice.
- Alice will prove $\Theta \vee \Phi_{Bob}$ to Bob.
- Cindy?

The Jakobsson-Sako-Impagliazzo (JSI) DVS scheme

- Undeniable signatures
- Trap-door commitment schemes
- Interactive and non-interactive designated verifier proofs
- Extension to multiple designated verifiers
- Strong designated verifier

Undeniable signatures

The challenge/response protocol:

- x, g^x, $z = m^x$
- Initial challenge $z^a (g^x)^b$
- Response equals $m^a g^b$?
- Choose c and d.
- Response $(r_1 g^{-b})^c = (r_2 g^{-d})^a$
- Probability p^{-1}

Trap-door commitment schemes

Definition 1. Let c be a function with input (y_i, w, r), where y_i is the public key of the user who will be able to invert c. The secret key corresponding to y_i is x_i, $w \in W$ is the value committed to and r a random string. We say that c is a *trap-door commitment scheme* if and only if

1. no polynomial-time machine can, given y_i, find a collision (w_1, r_1), (w_2, r_2) such that $c(y_i, w_1, r_1) = c(y_i, w_2, r_2)$
2. no polynomial-time machine can, given y_i and $c(y_i, w, r)$, output w.
3. there is a polynomial-time machine that given any quadruple (x_i, w_1, r_1, w_2) in the set of possible quadruples finds r_2 such that $c(y_i, w_1, r_1) = c(y_i, w_2, r_2)$ for the public key y_i corresponding to the secret key x_i.

Designated Verifier

Definition 2. Let (P_A, P_B) be a protocol for Alice to prove the truth of the statement Θ to Bob. We say that Bob is a *designated verifier* if the following is true: For any protocol (P_A, P'_B, P_C) involving Alice, Bob and Cindy, in which Bob proves the truth of ϑ to Cindy, there is another protocol (P''_B, P_C) such that Bob can perform the calculations of P''_B, and Cindy cannot distinguish transcripts of (P_A, P'_B, P_C) from those of (P''_B, P_C).

Interactive designated verifier proof of undeniable signatures

- Based on the generalisation of the confirmation scheme for undeniable signatures
- p, g generator of G_q, participant i's secret key x_i, public key $y_i = g^{x_i}$ mod p. m, participant i's signature on m: $s = m^{x_i} mod p$.

- The used confirmation scheme is the following:

  1. Bob uniformly at random selects two numbers a and b from $\mathbb{Z}_q$ and calculates $v = m^a g^b mod p$. Bob sends Alice v.
  2. Alice calculates $w = v^{x_A} mod p$. She calculates a commitment c to w and sends c to Bob.
  3. Bob sends (m, s, a, b) to Alice, who verifies that v is of the right form.
  4. Alice decommits to c by sending w and any possible random string r used for the commitment to Bob. Bob verifies that $w = s^a y_A^b mod p$ and that the commitment c was correctly formed.

Non- interactive designated verifier proofs

Constructing a proof

1. Alice, selects w, r, $t \in_u \mathbb{Z}_q$
2. Alice calculates

   $c = g^w y_B^r modp$

   $G = g^t modp$

   $M = m^t modp$

   $h = hash_q(c, G, M)$ (a hashed value in $\mathbb{Z}_q$)

   $d = t + x_A(h + w)modq$
3. Alice sends (w, r, G, M, d) to Bob

Verifying a proof

1. Bob calculates

$$c = g^w y_B^r mod p$$

$$h = hash_q(c, G, M)$$

2. Bob verifies that

$$G_{y_A}^{h+w} = g^d mod p$$

$$M_s^{h+w} = m^d mod p$$

Simulating transcripts

1. Bob selects $d, \alpha, \beta \in_u \mathbb{Z}_q$
2. Bob calculates

$$c = g^{\alpha} mod p$$

$$G = g^d y_A^{-\beta} mod p$$

$$M = m^d s^{-\beta} mod p$$

$$h = hash_q(c, G, M)$$

$$w = \beta - h \text{ mod } q$$

$$r = (\alpha - w) x_B^{-1} mod q.$$

Extension to Multiple Designated Verifiers

- Convincing a set of verifiers $\{Bob_i\}_{i=1}^{n}$
- Convince each individual Bob_i?
- Proposed solution: c is is one-way to each coalition of less than n of the designated verifiers, but invertible if they all cooperate.
- Distributing the secret key among the n designated verifiers.
- Cindy?

Strong designated verifier

Definition 3. Let (P_A, P_B) be a protocol for Alice to prove the truth of the statement Θ to Bob. We say that Bob is a *strong designated verifier* if the following is true: For any protocol (P_A, P_B, P_D, P_C) involving Alice, Bob, Dave and Cindy in which Dave proves the truth of ϑ to Cindy, there is another protocol (P'_D, P_C) such that Dave can perform calculations of P'_D and Cindy cannot distinguish transcripts of (P_A, P_B, P_D, P_C) from those of (P'_D, P_C).

- An honest Bob
- Transcripts can be probabilistically encrypted using the public key of the intended verifier
- Dave will not be able to present the decrypted transcripts to Cindy
- Cindy cannot distinguish encrypted transcripts from random strings of the same length and distribution

Security notions for DVS schemes

- Secure disavowability
- Unforgeability
- Non-delegatability
- Non-transferability

Secure disavowability and unforgeability

- Secure disavowability
  - Alice can prove that the signature was not simulated by Bob
  - Alice cannot disavow her own signatures
- Unforgeability
  - Signatures are verifiable by the designated verifier Bob
  - Bob rejects a signature when it was not signed by himself or Alice

Non-delegatability

Let $\kappa \in [0, 1]$ be the knowledge error. We say that Δ is (τ, κ)**-non-delegatable** if there exists a black-box knowledge extractor K that, for every algorithm F and for every valid signature σ, satisfies the following condition:

For every $(pk_A, sk_A) \leftarrow Generate$, $(pk_B, sk_B) \leftarrow Generate$ and message m, if F produces a valid signature on m with probability $\varepsilon > \kappa$ then, on input m and on access to the oracle F_m, K produces either sk_A or sk_B in expected time $\tau/(\varepsilon - \kappa)$

Non-transferability

- For an accepted message-signature pair (m, σ), and without access to the secret key of the signer, it is computationally infeasible to determine whether the message was signed by the signer, or the signature was simulated by the designated verifier.
- Let $\Delta = (Generate, Sign, Simulate, Verify)$ be a designated-verifier signature scheme with the message space M. We say that Δ is **perfectly non-transferable** if $Sign_{sk_A, pk_B}(m) = Simulate_{sk_B, pk_A}(m)$ as distributions for every $(pk_A, sk_A) \leftarrow Generate$, $(pk_B, sk_B) \leftarrow Generate$, $H_q \leftarrow \Omega$ ($\Omega = \Omega_{npro}$ or $\Omega = \Omega_{ro}$) and $m \leftarrow M$.
- Analogously defined: statistically non-transferable and computationally non-transferable schemes.

Disavowability attack on the JSI DVS scheme

- A malicious Alice can generate signatures exactly from the same distribution as Bob
- Alice computes a signature $(\overline{s}; w, t, G, \overline{M}, z)$ for a message m, with $\overline{s} \neq m^{x_A}$, as follows

  1. She uniformly elects four random numbers $w, t, r, \overline{r} \in \mathbb{Z}_q$
  2. She sets $c = g^w y_B^t mod p$

     $G = g^r mod p$

     $\overline{M} = m^{\overline{r}} mod p$

     $h = H_q(c, G, \overline{M})$

     $z = r + (h + w)x_A mod q$

     $\overline{s} = m^{x_A} \cdot m^{(r-\overline{r})/(h+w) mod q} mod p$

3. She sends a message-signature pair $(m, \overline{s})$ with $\overline{\sigma} = (\overline{s}, P = (w, t, G, \overline{M}, z))$ to Bob
4. Bob will believe that $\overline{s}$ is Alice's signature for message m
5. In later disputes, Alice can convince a third party that $\overline{s}$ was simulated by Bob, by using a standard disavowal protocol to show that $log_g y_A \neq log_m \overline{s}$.

Corrected JSI Scheme

- Solution 1
  - Alice must provide an additional proof of knowledge that $log_m M = log_g G$
  - This, however, increases the signature length
- Solution 2
  - Alice includes s (together with pk_A and pk_B) to the input of the hash function.
- The scheme is now unforgeable, non-delegatable, computationally non-transferable and securely disavowable

The DVS scheme with tight reduction to the DDH problem in the NPRO

- The Decisional Diffie-Hellman (DDH) assumption
- Random Oracles
- The DVS scheme (DVS-KW)

The Decisional Diffie-Hellman assumption

- A **group family** G is a set of finite cyclic groups $G = \{G_p\}$ where p ranges over an infinite index set.
- An **instance generator**, IG, for G is a randomised algorithm that given an integer n (in unary), runs in polynomial time in n and outputs some random index p and a generator g of G_p

Definition 4. Let $G = \{G_p\}$ be a group family. A Decisional Diffie Hellman (DDH) algorithm A for G is a probabilistic polynomial time algorithm satisfying, for some fixed $\alpha > 0$ and sufficiently large n:

$$|Pr[A(p, g, g^a, g^b, g^{ab}) = "true"] - Pr[A(p, g, g^a, g^b, g^c) = "true"]| > \frac{1}{n^\alpha}$$

where g is a generator of G_p. The probability is over the random choice of $\langle p, g \rangle$ according to the distribution induced by $IG(n)$, the random coice of a, b, c in the range $[1, |G_p|]$ and the random bits used by A. The group family G satisfies the **Decisional Diffie Hellman assumption** if there is no DDH algorithm for G.

Random Oracles

- The random-oracle model for a hash-function h is the model where h is replaced by a uniformly random function

- When a random oracle is given a query x it does the following:

  1. If the oracle has been given the query x before, it responds with the same value it gave the last time.

  2. If the oracle hasn't been given the query x before, it generates a random response which has uniform probability of being chosen from anywhere in the oracle's output domain.

NPRO and RO

- Random Oracle
  - The adversary does not know the secret key
  - The adversary is forced to program the random oracle to be able to answer successfully to the signature queries
- Non-programmable random oracle
  - The adversary knows Alice’s secret key
  - The adversary can answer successfully to the signature and simulation queries without a need to program the random oracle.

- The NPRO is known to be strictly weaker than the RO model
- Proofs in the RO model work for the "best case" (showing that for every forger there exists a function H_q such that the signature scheme is unforgeable)
- Proofs in the NPRO model work for the "average case" (showing that the signature scheme is unforgeable for a randomly chosen function $H_q \to \Omega_{npro}$, independent of the forger)

DVS-KW

- The signer presents a designated verifier proof that his public key is a Decisional Diffie-Hellman (DDH) tuple
- The unforgeability of this scheme is proved by providing a tight reduction to the underlying cryptographic problem (DDH) in the non-programmable random oracle (NPRO) model.
- This scheme is non-delegatable, correct and perfectly non-transferable, unforgeable in the non-programmable random oracle model.
- Proof of concept: has a tight reduction in the unforgeability proof and isstill non-delegatable
- More efficient than the JSI scheme

The scheme

- p, q ($q|(p-1)$)
- G_q is a multiplicative subgroup of $\mathbb{Z}_p^*$
- $g_1, g_2 \in G_q$
- Alice proves to Bob that $(g_1, g_2, y_{1A}, y_{2A})$ is a Decisional Diffie-Hellman tuple
- $x_i \leftarrow_r \mathbb{Z}_q$ is i's private key, $pk_i = (g_1, g_2, y_{1i}, y_{2i})$ is i's public key with $y_{1i} = g_1^{x_i}$ and $y_{2i} = g_2^{x_i}$.
- Making the scheme designated-verifier
- Non-interactive - using a non-programmable random oracle H_q with outputs from $\mathbb{Z}_q$

Generating a proof $Sign_{sk_A,pk_B}(m)$:

1. Alice generates random $r, w, t \leftarrow \mathbb{Z}_q$
2. She sets

   $$a_1 = g_1^r mod p$$

   $$a_2 = g_2^r mod p$$

   $$c = g_1^w y_{1B}^t mod p$$

   $$h = H_q(pk_A, pk_B, a_1, a_2, c, m)$$

   $$z = r + (h + w)x_A mod q$$

3. She outputs the signature $\sigma = (w, t, h, z)$.

Simulating a signature $Simulate_{sk_B, pk_A}(m)$:

1. Bob selects three random numbers $z, \alpha, \beta \leftarrow_r \mathbb{Z}_q$
2. Bob calculates

   $(a_1, a_2) = (g_1^z y_{1A}^{-\beta} mod p, g_2^z y_{2A}^{-\beta} mod p)$

   $h = H_q(pk_A, pk_B, a_1, a_2, g_1^{\alpha} mod p, m)$

   $w = \beta - h$ mod q

   $t = (\alpha - w) x_B^{-1} mod q$

Verifying a proof $Verify_{pk_A, pk_B}(m; w, t, h, z)$:

1. Bob checks whether

   $h = H_q(pk_A, pk_B, g_1^z y_{1A}^{-(h+w)}$ mod $p, g_2^z y_{2A}^{-(h+w)}$ mod $p, g_1^w y_{1B}^t mod p, m)$.

Universal designated verifier signature without random oracles

- Bilinear groups
- A short signature scheme without random oracles
- Model of UDVS
- Security Notions for UDVS
- Model of UDVS without Random Oracles

Bilinear groups

Definition 5. Let V and W be vector spaces over the same field F. A *linear transformation* is a function $T : V \to W$ such that

1. $T(v + w) = T(v) + T(w)$ for all $v, w \in V$
2. $T(\lambda v) = \lambda T(v)$ for all $v \in V$ and $\lambda \in F$.

Definition 6. Let S and U be vector spaces over a field K. A function $B : S \times U \to K$ is called a *bilinear map* if

1. $x \mapsto B(x, y)$ is linear for each $y \in U$
2. $y \mapsto B(x, y)$ is linear for each $x \in S$

That is, B is bilinear if it is linear in each parameter taken separately.

Short signature scheme without random oracles

- Let (G_1, G_2) be bilinear groups, $|G_1| = |G_2| = p$ for some large prime p
- $m \in \mathbb{Z}_p^*$ is the message

Generating the keys

1. Pick a random generator $g_2 \in G_2$ and set $g_1 = \psi(g_2)$, pick $x, y \leftarrow \mathbb{Z}_p^*$
2. Compute $u = g_2^x$
   $v = g_2^y$
3. For fast verification, also compute $z = e(g_1, g_2) \in G_T$

The public key is (g_1, g_2, u, v, z) and the secret key is (x, y).

Signing

1. Pick $r = \mathbb{Z}_p^*$
2. If $x + r + ym = 0$ mod p, try again with a different random r
3. Compute $\sigma = g_1^{1/(x+r+ym)} \in G_1$

The signature is (σ, r).

Verifying

1. Given the public key (g_1, g_2, u, v, z), a message $m \in \mathbb{Z}_p^*$, and a signature (σ, r) accept if $e(\sigma, u \cdot g_2^r \cdot v^m) = z$, otherwise, reject.

Model of UDVS

UDVS = (CPG, SKG,VKG, S, PV,DS,DV, P_{KR}).

1. **Common Parameter Generation** CPG
2. **Signer Key Generation** SKG
3. **Verifier Key Generation** VKG
4. **Signing** S
5. **Public Verification** PV
6. **Designation** DS
7. **Designated Verification** DV
8. **Verifier Key-Registration** P_{KR}(KR,V)

Security notions for UDVS

- Strong DV-unforgeability
  - Public Verifiable signature unforgeability - security of the signer
  - Designated Verifier signature unforgeability - security for the designated verifier
- Non-transferability
  - Unconditionally non-transferable against adaptive chosen public key attack and chosen message attack (NT-CPKMA)
  - $\exists S$: for every A, every computationally unbounded D distinguishes outputs of A and S on any challenge message m^* with only probability $negl(k)$
  - A is able to access to Designation oracle with respect to any message before the challenge message is determined
  - This helps the adversary adaptively choose the challenge message

Model of UDVS without random oracles

1. **Common Parameter Generation** CPG
   - Str_D : (G_1, G_2) of prime order $|G_1| = |G_2| = p$
   - Bilinear map e : $G_1 \times G_2 \rightarrow G_T$
   - Isomorphism ψ : $G_2 \rightarrow G_1$
   - Choose a random generator $g_2 \in G_2$
   - Compute $g_1 = \psi(g_2) \in G_1$.
   - The common parameter is $cp = (Str_D, g_1, g_2)$.

2. **Signer Key Generation** SKG

- Given cp, pick random $x_1, y_1 \leftarrow \mathbb{Z}_p^*$
- Compute $u_1 = g_2^{x_1}$ and $v_1 = g_2^{y_1}$
- For speeding up the verification, compute $z \leftarrow e(g_1, g_2) \in G_T$
- The public key is $pk_a = (cp, u_1, v_1, z)$
- The secret key is $sk_a = (x_1, y_1)$

3. **Verifier Key Generation** VKG

- Given cp, pick random $x_3, y_3 \leftarrow \mathbb{Z}_p^*$
- Compute $u_3 = g_2^{x_3}$ and $v_3 = g_2^{y_3}$
- The public key is $pk_b = (cp, u_3, v_3)$
- The secret key is $sk_b = (x_3, y_3)$

4. **Signing S**

- Given the signer's secret key (cp, x_1, y_1) and a message m, select $r \leftarrow \mathbb{Z}_p^*$
- If $x_1 + r + my_1 = 0$ mod p, restart
- Compute $\sigma = g_1^{1/(x_1+r+my_1)}$
- Output $s = (\sigma, r)$ as the PV-signature

5. **Public Verification** PV

- Given the signer's public key (cp, u_1, v_1, z), and a message/PV-signature pair (m, s)
- Accept only if $e(\sigma, u_1 \cdot g_2^r \cdot v_1^m) = z$
- Otherwise reject

6. **Designation** DS

   - Given the signer's public key (cp, u_1, v_1), a verifier's public key (cp, u_3, v_3) and a message/PV-signature pair (m, s), where $s = (\sigma, r)$, let $h = g_2^r$
   - Compute $d = e(\psi(u_3), v_3^r) \in G_T$
   - The DV-signature is $\overline{s} = (\sigma, h, d)$.

7. **Designated Verification** DV

   - Given a signer's public key (cp, u_1, v_1), a verifier's secret key (x_3, y_3), and message/DV-signature pair $(m, \overline{s})$
   - Accept only if the following two equations hold simultaneously:

     $z = e(\sigma, u_1 \cdot h \cdot v_1^m)$

     $d = e(\psi(u_3), h^{y_3})$
   - Otherwise reject

Properties of UDVS

The scheme is

- Correct
- Unforgeable against adaptive chosen public key attack and chosen message attack for designated verifier
- Unconditionally non-transferable