Designated Verifier Proofs and Their Applications

Markus Jakobsson[1] *, Kazue Sako[2], and Russell Impagliazzo[1]**

[1] Department of Computer Science and Engineering,
University of California, San Diego, La Jolla, CA 92093.
Email: {markus,russell}@cs.ucsd.edu.
[2] NEC Corporation, 4-1-1 Miyazaki Miyamae, Kawasaki 216, JAPAN
Email: sako@sbl.cl.nec.co.jp.

Abstract. For many proofs of knowledge it is important that only *the verifier designated by the confirmer* can obtain any conviction of the correctness of the proof. A good example of such a situation is for undeniable signatures, where the confirmer of a signature wants to make sure that only the intended verifier(s) in fact can be convinced about the validity or invalidity of the signature.
Generally, authentication of messages and off-the-record messages are in conflict with each other. We show how, using designation of verifiers, these notions can be combined, allowing authenticated but private conversations to take place. Our solution guarantees that *only* the specified verifier can be convinced by the proof, even if he shares all his secret information with entities that want to get convinced.
Our solution is based on *trap-door commitments* [4], allowing the designated verifier to open up commitments in any way he wants. We demonstrate how a trap-door commitment scheme can be used to construct designated verifier proofs, both interactive and non-interactive. We examplify the verifier designation method for the confirmation protocol for undeniable signatures.

1 Introduction

Background. When undeniable signatures [5] were introduced, the following scenario served as a motivation: A software vendor puts digital signatures 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 these signatures. One property of undeniable signatures is that their validity or invalidity cannot be verified without interaction with a prover. This in itself, however, only allows the prover to decide *when* a signature is verified, and not *by whom* (or even by

* Research supported by NSF YI Award CCR-92-570979, Sloan Research Fellowship BR-3311, and The Royal Swedish Academy of Sciences.
** Research supported by NSF YI Award CCR-92-570979 and Sloan Research Fellowship BR-3311.

how many), because of blackmailing [11, 22] and mafia[3] attacks [10]. In order to avoid these types of attacks, in both of which the conviction is transferred to one or several hidden co-verifiers, the prover must be able to designate *who* will be convinced by a proof.

Another situation where it is important for the prover to designate who can be convinced by the validity of a proof is when a voting center wants only a voter himself to be convinced that the vote he cast was counted. Here, it is important for the voter Bob to be convinced by the proof, but we want to prevent an armed coercer, Cindy, to be able to force Bob to prove to her how he voted. Other methods – not clearly stating designation of verifiers – of avoiding such attacks in order to obtain receipt-free electronic voting schemes were studied in [3, 34].

RESULTS. This paper suggests a solution, *designation of verifiers*, that resolves the conflict between authenticity and privacy, and dodges the described attacks by limiting who can be convinced by a proof. We say that we *designate a verifier* in a proof when we acertain that nobody but this participant can be convinced by the proof. The intuition behind the solution can be described in one sentence: *Instead of proving Θ, Alice will prove the statement "Either Θ is true, or I am Bob."* Bob will certainly trust that Θ is true upon seeing such a proof[4], but if Bob diverts the proof to Cindy, Cindy will have no reason at all to believe that Θ is true, as Bob is fully capable of proving himself to be Bob. We show how, with only small changes in the confirmation protocol for undeniable signatures, the confirmer can designate verifiers. Also we demonstrate that we can have designated verifier non-interactive undeniable signatures, which combines desireable properties of ordinary signatures and undeniable signatures.

In order to solve the problem, we will use trap-door commitment schemes, also known as chameleon commitment schemes, introduced by Brassard, Chaum and Crépeau [4]. Assume the trap-door information is known to Bob only. If Alice uses the scheme to commit to a value, Bob will trust her commitment as Alice cannot find collisions, but if the proof is diverted to Cindy by Bob, Cindy has no reason to trust the commitment, as Bob can decommit as he wishes. It is important to point out that the verifier designation holds even if the designated verifier chooses to reveal secret information to cooperating co-verifiers, since Bob can *still* find collisions after revealing secret information.

RELATED WORK. A large variety of digital signature schemes (e.g., [12, 23, 33]) has been introduced, in which the validity of the signatures is publicly verifiable. On the other hand, there also are more authentication-like schemes, such as schemes built on zero-knowledge protocols, e.g., the Feige Fiat Shamir identification protocol [16], where the authenticity of a message cannot be verified by *anyone* once the interactive verification session is terminated. In between

[3] Also known as the man-in-the-middle attack.

[4] Technically, this is not a *proof* of Θ, but an argument, as a powerful prover could cheat by calculating the verifier's secret key from his or her public key.

these two schemes are undeniable signatures [5], which can be freely distributed and verified any number of times, but only with the cooperation of a confirmer. However, several cooperating verifiers can each be convinced about the validity of the signature in these interactive protocols, as is shown in [11, 22]. (Here, the verifiers jointly set the challenge in a way so that none of them can determine the outcome.) This attack shows that designation of messages is a necessary tool for authentication of private messages. A well-known method of doing this is for the sender of a message to encrypt it using a secret key that only he and the receiver knows, but this only applies to plain messages and not to proofs. Still, it is a form of verifier designation, as it allows the receiver of a message to simulate identical transcripts, thereby making it impossible to prove that the transcript indeed originated with the claimed sender.

In [13], Dolev, Dwork and Naor presented a solution, non-malleable cryptography, avoiding some mafia-attacks. In their scenario, researcher A has proven that $P \neq NP$, and wants to convince researcher B about this in a zero-knowledge fashion, but without allowing this person to convince researcher C and claim credit for the discovery himself. However, researcher C *will* be convinced that the statement proven is true, although she will know that A, not B, is the prover. Thus, conviction is transferable, but credits not. However, in the cases we presented, where the sender *does not* want the conversation to be possible to trace back to him or her, this is not sufficient. If a *designated verifier proof* is used, it will be impossible for a third party to be convinced about the validity of a proof sent from A to B. This holds *even if B should cooperate with C.*

Outline. We start by explaining our model in Section 2, followed by definitions and examples in Section 3. We examplify our method in Section 4 by showing how to designate verifiers in the verification protocol for undeniable signatures. A designated verifier proof can be made heuristically non-interacative, and we give examples and applications for this in Section 5. In Section 6, we show how the result can be extended to multiple designated verifiers. We discuss practical issues in Section 7, and an alternative (and stronger) definition of designated verifier in the Appendix.

2 Approach and Model

Approach. Alice wants to prove to Bob – but only to him – that the statement Θ is true. Let Φ_{Bob} be the statement "I know Bob's secret key." Using the methods soon to be introduced, Alice will prove $\Theta \vee \Phi_{Bob}$ to Bob, who will be convinced that Θ is true (or that his secret key has been compromised.) Given that Bob knows his secret key, Cindy will not be convinced that Θ is true after seeing a proof of $\Theta \vee \Phi_{Bob}$, which holds even if Bob cooperates with her and shares secret information with her. This is so since Bob can produce such a proof himself, independently of whether Θ is true or not.

Participants. All participants are assumed to be polynomial-time limited. Partipants must only be able to be convinced by a proof designated to them if they

are able to simulate transcripts of the same distribution themselves, i.e., they have to know their secret key. (We discuss methods to ensure this in section 7.)

THREAT MODEL. It is important to remember that we want to reach our goal of designating who will be convinced by a proof for pragmatic reasons, in order for the use of authentication of data not to be possible to abuse with a purpose of damaging the interests of the legal participants *in the real, physical world.* Should an attacker be in total control of Bob's physical and mental self, then we mean that the attacker has already achieved his goal. Also, in a sense, the attacker has *become* Bob. We therefore mean that it is realistic assumption to make that the verifier has *not* totally lost control of his ability to access his data, perform calculations and freely communicate.

This, in particular, excludes two possible attacks:

1. *The Suicide Attack*
   Bob kills all his aliases, provides the attacker, Cindy, with his secret data, and then self-destructs. (This effectively transfers the notion of "being Bob" from Bob to Cindy.)
2. *The Demon Attack*
   Cindy locks Bob (and all his aliases) up, preventing him from communicating with anybody but her, and forces him to perform certain calculations and prove them correct. (This corresponds to taking total command of somebody's mental functions.)

TRUST MODEL. Let us call a person who wants to act as a verifier in a proof, but is not designated by the prover, a *hidden verifier.* We assume that a hidden verifier, Cindy, will not trust the designated verifier, Bob, that he did not produce the transcript for the proof of $\Theta \vee \Phi_{Bob}$, where Φ_{Bob} is a proof of knowledge of Bob's secret key, or she could just as well take his word for that Θ is true in the first place, without seeing a proof of it.
In Appendix A, we introduce a weaker trust model and a correspondingly stronger notion of designated verifier, *strong designated verifier*, in which Cindy trusts Bob to be perfectly honest, i.e., never to engage in a protocol that is not part of the system. (Here, trying to convince Cindy that $\Theta \vee \Phi_{Bob}$ is true is *not* one of the prescribed protocols.) However, Cindy can try to trick Bob to convince her by interacting in "legal" protocols. The reason why this is a *weaker* trust model and a stronger security notion is that a protocol that is *designated verifier* is not necessarily *strong designated verifier*, whereas the converse holds.

3 Definitions

In this section, we give informal and intuitive definitions.

Definition 1 Designated Verifier.
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*[5] 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).

Definition 2 Trap-Door Commitment. (also see [4])
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. Here, 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.

Let us give two examples of trap-door schemes:

Example Trap-door commitment scheme 1. [4]
Secret key of the receiver: $x_B \in_u Z_q$.
Public key of the receiver: $y_B = g^{x_B} \bmod p$. Here, $p = q * k + 1$ for two primes p, q and $k \in Z$; g is a generator of the subgroup G_q of Z_p^*, of order q.
Value to commit to: $w \in Z_q$.
Commitment: Alice selects $r \in_u Z_q$. The commitment is $c = g^w {y_B}^r \bmod p$.
Decommitment: Alice sends Bob (w, r).

Example Trap-door commitment scheme 2. [8]
Secret decryption scheme of the receiver: $D_B(.)$
Public encryption scheme of the receiver: $E_B(.)$
Value to commit to: $w \in Range(E_B)$.
Commitment: Alice will uniformly at random select $r \in Range(E_B)$. She calculates a commitment $c = E_B(w) \oplus E_B(r)$, where $\oplus$ is a combiner such as XOR.
Decommitment: Alice sends Bob (w, r).

Scheme 2 is a trap-door commitment scheme if arbitrary collisions (w_1, r_1), (w_2, r_2) such that $E_B(w_1) \oplus E_B(r_1) = E_B(w_2) \oplus E_B(r_2)$ can be found if and only if $D_B(.)$ is known. The use of e.g. RSA [33] seems plausible.

4 Interactive Designated Verifier Proof of Undeniable Signatures

We show how to change the normal verification protocol for undeniable signatures to make it designated verifier. We will base our scheme on the confirmation

[5] Occasionally, we will also refer to the (prover's part of the) corresponding protocol as designated verifier.

scheme for undeniable signatures[6][6]. We use denotation similar to that in the original scheme: We will let p be a large prime, g a generator of G_q, participant i's secret key is x_i and his public key is $y_i = g^{x_i} \bmod p$. If m is a message, participant i's signature on m will be $s = m^{x_i} \bmod p$.

The following scheme is the confirmation scheme for undeniable signatures, given in [6]:

1. Bob uniformly at random selects two numbers a and b from Z_q and calculates $v = m^a g^b \bmod p$. Bob sends Alice v.
2. Alice calculates $w = v^{x_A} \bmod p$. 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 $w = s^a {y_A}^b \bmod p$ and that the commitment c was correctly formed.

Making it Designated Verifier: The above scheme can be made designated verifier by letting c be a trap-door commitment scheme, using the public key of the designated verifier.

5 Non-interactive Designated Verifier Proofs and Their Applications

In this section we present a *non-interactive* designated verifier proof of an undeniable signature. Such a scheme bridges the gap between publicly verifiable digital signatures and undeniable signatures, in that it limits who can verify it *without help from the prover*, but does not necessitate interaction. Still, and in contrast to publicly verifiable signatures that are made designated verifier, they can be used for contracts, etc., as their validity can be verified when the prover agrees to this. We believe that this property makes them a very useful tool in balancing the need for privacy against that of authenticity.

First, we discuss a general method to transform ordinary three-move zero-knowledge protocols to non-interactive designated verifier proofs. Then, we examplify this method by showing a non-interactive undeniable signature verification scheme with designation of verifiers. Our technique can be used to obtain non-interactive, non-transitive signatures [28].

Non-Interactive Designated Verifier Proofs

An ordinary three-move zero-knowledge protocol can be described as a commit - challenge - response protocol. The Fiat-Shamir technique [18] is a famous trick for making such a protocol non-interactive, while preserving the security of the protocol in a practical manner [17]. By generating challenges from a hashed value

[6] We show the "folklore generalization" of this scheme, in which the commitment scheme is not specified.

of multiple commitments, the three moves can be collapsed into one single move. However, this resulting transcript is in itself a transitive proof whose correctness can be verified by anyone.

In order to construct a designated verifier proof, we modify the scheme and use a trap-door commitment in the commitment stage. It becomes a designated verifier proof, as Bob, the designated verifier, can always use his trap-door to simulate a transcript for any statements. Cindy cannot distinguish a valid proof of a true statement from an invalid proof forged by Bob.

A Non-interactive Undeniable Signature Scheme

We use the same denotation as in section 4, where the confirmation scheme consists of a three-move zero-knowledge protocol for proving that Alice's public key y_A and the signature s have a common exponent x_A with respect to a public generator g and the message m [7]. This is a corresponding non-interactive designated verifier proof:

Constructing a proof:
The prover, Alice, selects $w, r, t \in_u Z_q$ and calculates

$$\begin{cases} c = g^w {y_B}^r \bmod p \\ G = g^t \bmod p \\ M = m^t \bmod p \\ h = hash_q(c, G, M) \\ d = t + x_A(h + w) \bmod q \end{cases}$$

where $hash_q$ gives you a hashed value in Z_q. The prover sends (w, r, G, M, d) to the verifier, Bob.

Verifying a proof:
The designated verifier can verify a proof by calculating

$$\begin{cases} c = g^w {y_B}^r \bmod p \\ h = hash_q(c, G, M). \end{cases}$$

and verifying that

$$\begin{cases} G\, {y_A}^{h+w} = g^d \bmod p \\ M\, s^{h+w} = m^d \bmod p. \end{cases}$$

Simulating transcripts:
The designated verifier can simulate correct transcripts by selecting $d, \alpha, \beta \in_u Z_q$ and calculate

$$\begin{cases} c = g^\alpha \bmod p \\ G = g^d {y_A}^{-\beta} \bmod p \\ M = m^d s^{-\beta} \bmod p \\ h = hash_q(c, G, M) \\ w = \beta - h \bmod q \\ r = (\alpha - w){x_B}^{-1} \bmod q. \end{cases}$$

[7] A disavowal scheme can also be described as a series of 3-move protocols[9] which can be merged in a non-interactive designated verifier proof.

6 Extension to Multiple Designated Verifiers

If Alice in a proof of knowledge wants to convince a set of n verifiers, $\{\text{Bob}_i\}_{i=1}^n$, but only these, the trivial approach is for Alice to convince each individual verifier, Bob_i, in an individual proof.

A more appealing approach is the following solution: We will use exactly the same protocol as for only one designated verifier, but let c be a function that is one-way to each coalition of less than n of the designated verifiers, but invertible if they all cooperate. This can easily be done by letting the secret key corresponding to the public key used be distributed among all the n designated verifiers so that they all need to cooperate to calculate it. It is, however, not necessary for the designated verifiers to share a secret in advance. For example, using the DL-based commitment scheme (trap-door commitment scheme number one,) the following modified commitment scheme can be used to extend to multiple verifiers, Bob_1 to Bob_n:

Trapdoor commitment scheme 1, modified for multiple verifiers:
Individual secret keys of the receivers: $\{x_{B_i}\}_{i=1}^n$, $x_{B_i} \in_u Z_q$.
Shared secret key of the receivers: $x_B = \sum_{i=1}^n x_{B_i} \bmod q$.
Individual public keys of the receivers: $\{y_{B_i}\}_{i=1}^n$, $y_{B_i} \in Z_p$.
Shared public key of the receivers: $y_B = \prod_{i=1}^n y_{B_i} \bmod p$.
Value to commit to: $w \in Z_q$.
Commitment: *Alice selects* $r \in_u Z_q$. *The commitment is* $c = g^w {y_B}^r \bmod p$.
Decommitment: *Alice sends all the Bobs* (w, r).

Conviction. Each designated verifier would be convinced by the proof as long as he knows that his share of the secret key has not been compromised. However, no "outsider", Cindy, would be able to receive conviction, as the set of designated verifiers, $\{\text{Bob}\}_{i=1}^n$, could have cooperated to cheat her. This they can do without revealing their personal shares of the secret key to each other.

7 Practical Issues

In order to assure that a designated verifier who can be convinced by a proof indeed also is able to simulate identically distributed transcripts, we have to require that he can only be convinced of a proof designated to him if he knows his secret key.

Depending on the situation, different relationships between logical and computational entities will have to be enforced. Sometimes, it is sufficient to define a logical entity as the set of computational entities who must cooperate in order to output the secret key corresponding to the public key of the logical entity and not enforce any particular relationship. One example where this may be sufficient is for a company being the logical entity and its employees being the computational entities. It is in the best interest of the company not to allow a competitor to be part of the set of computational entities, as this makes the company rely on its competitor for signing, decryption, etc. However, this way

of defining logical entities is not appropriate in all settings. We can find an example of such a setting in the motivation for undeniable signatures: A software company will prove validity of signatures, and thereby virus freeness of the corresponding programs, but only to clients who buy this service. Here, the company wants to make sure that it is not possible for several computational entities to be one logical entity, thereby letting them all be convinced by one proof of validity.

We see that it is of interest that Bob cannot convince Cindy that he does not know his own secret key, or she *would* trust that Θ is true upon seeing a proof of $\Theta \vee \Phi_{Bob}$. Bob can convince her that he does not know his secret key in one of two ways:

1. Bob shows Cindy a preimage over a one-way function of his public key, other than his secret key. If Cindy believes that Bob cannot calculate a triple (public key, secret key, other preimage), then she believes that Bob does not know his secret key, and consequently, cannot prove Φ_{Bob}.
2. Bob and Cindy secret shares his secret key, which they generated together, and Cindy has not released her share of it to Bob. Here, Cindy and Bob has to collude before the proof session started in order for Cindy to be convinced by the proof of Θ.

We suggest two alternative approaches for ensuring that Bob cannot transfer the conviction using the former technique:

1. Before the proof of knowledge of $\Theta \vee \Phi_{Bob}$ from Alice to Bob, Bob has to prove knowledge of Φ_{Bob} to Alice.
2. When proving $\Theta \vee \Phi_{Bob}$, Alice probablistically encrypts the transcripts she sends (or parts thereof) using an encryption function for which decryption abilities enables arbitrary collision finding for the trap-door commitment scheme used. This is easily done for our second example of trap-door commitment schemes.

In order to avoid the second "attack", the sharing of the secret key, we suggest two possible approaches:

1. When Bob is registering his public key to have it certified, he has to prove knowledge of his secret key to the Certification Agency, in a setting where he can *only* communicate with the CA (e.g., a smart-card setting.)
2. When registering his public key, Bob presents his secret key to the CA, who then has to be trusted to neither divulge it to someone else nor to prove knowledge of it.

In an implementation, a combination of the above approaches has to be taken to ensure that the receiver of a readable[8] proof indeed knows his secret key, and that the prover is convinced of this.

[8] Referring to the possible use of probabilistic encryption.

8 Conclusion

We have presented a method to designate verifiers, allowing authentication to be combined with privacy (in the sense that the authentication cannot be forwarded to a non-designated party.) We have shown how the verification protocol for undeniable signatures can be made designated verifier, and have demonstrated both an interactive and a non-interactive version. We have discussed how the results can be extended to multiple designated verifiers, and briefly treated practical issues.

Acknowledgements

The first author wishes to thank Mihir Bellare, David Chaum, Giovanni Di Crescenzo, Niels Ferguson and Moti Yung for interesting and helpful comments. Special thanks to Ivan Damgård for valuable comments and important guidance during the preparation of the final version. Finally, we wish to thank the anonymous referees for important feedback.

References

1. M. Bellare, S. Goldwasser, "New Paradigms for Digital Signatures and Message Authentication Based on Non-Interactive Zero Knowledge Proofs," Crypto '89, pp. 194-211.
2. M. Bellare, S. Micali, "How to Sign Given Any Trapdoor Function," 20th Annual STOC, 1988, pp. 32-42.
3. J.C. Benaloh, D. Tuinstra, "Receipt-Free Secret-Ballot Elections," 26th Annual STOC, 1994, pp. 544-553.
4. G. Brassard, D. Chaum, C. Crépeau, "Minimum Disclosure Proofs of Knowledge," Journal of Computer and System Sciences, Vol. 37, No. 2, Oct. 1988, pp. 156-189
5. D. Chaum, H. van Antwerpen, "Undeniable Signatures," Crypto '89, pp. 212-216
6. D. Chaum, "Zero-Knowledge Undeniable Signatures," Eurocrypt '90, pp. 458-464
7. D. Chaum, E. van Heijst, B. Pfitzmann, "Cryptographically Strong Undeniable Signatures, Unconditionally Secure for the Signer," Crypto '91, pp. 470-484
8. D. Chaum, personal communication
9. I. Damgård, personal communication
10. Y. Desmedt, C. Goutier, S. Bengio, "Special Uses and Abuses of the Fiat-Shamir Passport Protocol," Crypto '87, pp. 21-39
11. Y. Desmedt, M. Yung, "Weaknesses with Undeniable Signature Schemes," Eurocrypt '91, pp. 205-220
12. W. Diffie, M.E. Hellman, "New Directions in Cryptography," IEEE Transactions on Information Theory, v. IT-22, n. 6, Nov 1976, pp. 644-654
13. D. Dolev, C. Dwork, M. Naor, "Non-Malleable Cryptography," 23rd Annual STOC, 1991, pp. 542-552
14. T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithm," IEEE IT 31 (1985), pp. 469-472
15. S. Even, O. Goldreich, S. Micali, "On-Line/Off-Line Digital Signatures," Crypto '89, pp. 263-275

16. U. Feige, A. Fiat, A. Shamir, "Zero Knowledge Proofs of Identity," Proceedings of the 19th annual ACM Symposium on Theory of Computing, pp. 210-217
17. U. Feige, A. Shamir, "Witness Indistinguishable and Witness Hiding Protocols," 22nd Annual STOC, 1990, p. 416–426.
18. A. Fiat, A. Shamir, "How to prove yourself; practical solution to identification and signature problems," Crypto '86, pp. 186-194
19. Z. Galil, S. Haber, M. Yung, "Symmetric Public-Key Cryptosystems", submitted to J. of Cryptology
20. S. Goldwasser, S. Micali, "Probabilistic Encryption & How To Play Mental Poker Keeping Secret All Partial Information," Proceedings of the 18th ACM Symposium on the Theory of Computing, 1982, pp. 270-299
21. O. Goldreich, S. Micali, A. Widgerson, "Proofs that Yield Nothing but their Validity or All Languages in NP Have Zero-Knowledge Proof Systems," Journal of the ACM, vol. 38, n. 1, 1991, pp. 691-729
22. M. Jakobsson, "Blackmailing using Undeniable Signatures", Eurocrypt '94, pp. 425-427
23. R.C. Merkle, "Secure Communication over Insecure Channels," Communications of the ACM, v. 21, n. 4, 1978, pp. 294-299
24. R. Merkle, "A Certified Digital Signature," Crypto '89, pp. 218-238
25. S. Micali, A. Shamir, "An Improvement of the Fiat-Shamir Identification and Signature Scheme," Crypto '88, pp. 244-247
26. M. Naor, M. Yung, "Universal One-Way Hash Functions and their Cryptographic Application," 21st Annual STOC, 1989, pp. 33-43
27. T. Okamoto, K. Ohta, "Divertible Zero-Knowledge Interactive Proofs and Commutative Random Self-Reducibility," Eurocrypt '89, pp. 134-149
28. T.Okamoto, K. Ohta, "How to Utilize Randomness of Zero-Knowledge Proofs," Crypto '90, pp 456–475.
29. H. Ong, C. P. Schnorr, "Fast signature generation with a Fiat-Shamir like scheme," Eurocrypt 90, pp. 432-440
30. T. Pedersen, "Distributed Provers with Applications to Undeniable Signatures," Eurocrypt '91, pp. 221-238
31. J.-J. Quisquater, L.S. Guillou, "A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory," Eurocrypt '88, pp. 123-128
32. C. Rackoff, D. Simon, "Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack", Crypto '91, pp. 433-444
33. R. Rivest, A. Shamir, L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," Communications of the ACM, v. 21, n. 2, Feb 1978, pp. 120-126
34. K. Sako, J. Kilian, "Receipt-Free Mix-Type Voting Scheme," Eurocrypt'95, pp 393-403.
35. A. Yao, "Protocols for Secure Computations," Proceedings of the 23rd FOCS, 1982, pp. 160-164

Appendix: Strong Designated Verifier

Previously, we have assumed that Cindy, the third party, will not be convinced that Θ holds after seeing a proof of $\Theta \vee \Phi_{Bob}$, as she knows that Bob could

have produced such a transcript himself, Φ_{Bob} being the proof of knowledge of Bob's secret key. Let us now assume that it is widely believed that Bob would not engage in such "transcript forgery," being the thoroughly pure and honest person he is, and will only engage in the prescribed protocols of the system. Under such circumstances, Cindy would, indeed, be convinced that Θ is true after seeing a proof of $\Theta \vee \Phi_{Bob}$. If we want this not to be possible, we need a stronger notion of what it means to be a designated verifier:

Definition 3 Strong Designated Verifier.
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 the 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).

This definition captures the fact that Bob is honest (i.e., he engages only in the protocol P_B, where this may be a concatenation of any polynomial number of "legal" protocols.) In order to make protocols strong designated verifier, transcripts can be probabilistically encrypted using the public key of the intended verifier. No "pure and honest" participant will be agree to decrypt ciphertexts of this particular type. Thus, since Dave will not be able to present the decrypted transcripts to Cindy, and Cindy cannot (due to the probabilistic encryption) distinguish encrypted transcripts from random strings of the same length and distribution (which is sampleable,) Dave will be able to produce transcripts (P_B, P'_D, P_C) that Cindy cannot distinguish from transcripts of (P_A, P_B, P_D, P_C).