SCIS 2002 The 2002 Symposium on Cryptography and Information Security
Shirahama,Japan, Jan.29-Feb.1,2002
The Institute of Electronics, Information and Communication Engineers

Instance Revocation of Digital Signature and Its Applications

Rui Zhang * Michiharu Kudoh † Kanta Matsuura ‡ Hideki Imai §

Abstract – This work is to have significant instance-management functions regarding digital signature. While a wide variety of the certificate revocation technology has been studied, an idea for instance revocation of digital signature without revoking the certificate has not been investigated. We show that the problem of the instance signature revocation is equivalent to the problem when and how the non-repudiatable property can be removed from the signature. We present a signature revocation model based on the signature revocation token and its revocation protocols, which enables the notion of instance revocation of digital signature on top of the existing signature schemes. Using our proposed model and protocols, the signer can revoke his previously signed message under the mutual consent with the signature receiver. We also present a construction for the privacy-enhanced digital signature that allows the signer to control the universal verifiability of the signature.

Keywords: Instance management, signature revocation

1 Introduction

1.1 Overview

Digital signature is a very important primitive in cryptographic protocols. Basically any legal receiver of the signature can verify the signature, thus the signature authenticates a certain will or duty of the signer. However, since digital contents are easy to copy, it will be very difficult to control where the signature goes. Suppose that a customer buys a book in a bookstore with a credit card. Later in the permitted period, the customer may want to return the book and have his money back. In the paper world, it will be very easy that if customer returns the book, he will get the cash or the shop assistant will tear the slip of receipt for the credit card, thus the bargain is cancelled. But in a digital world, since there is no face-to-face communication, and everyone is behind the network and only signs the contract with digital signatures. There will be not visible proof that the signed signature is really destroyed even after one party says so. In the future if the shop should ask the customer to execute the buying contract, the customer would have no way to protect himself. For he did sign such a signature in the past, and if there is no expiring date in the signature he will have no way to prove it to a judge that the signature has been revoked.

Thus there appear one problem that in some occasion the signer wants to take back his previously singed signature, in other words, to revoke his legal signature in a proper way, so that after certain procedures, an existed signature can be regarded as invalidate, which leads to a new study of instance management regarding digital signature.

1.2 Related work

Ordinary digital signature has one prominent property that every legal receiver can have the right to verify the signature. In contrast to this, after a signature has been distributed, the signer will have no way to prevent others from verifying the signature or re-distributing the signature. To solve the unauthorized re-distribution problem, Chaum and van Antwerpen proposed the "Undeniable Signatures" and "Zero-knowledge Undeniable Signatures" ([7],[5]), based on zero-knowledge proof that any receiver of the undeniable signature should get the help from the signer to verify the signature he has received. However, another problem occurs when in some case the signer should refuse to cooperate or be not available. Then the "Designated Confirmer Signature" ([6]) was proposed that lets some third party help the receiver to verify the received signature.

Up to now, there is some work with respect to "Revocation". We have "Private Key Traitor Trace Scheme with Revocation" ([12]) in the broadcast encryption, "Certificate Revocation" ([10],[8],[9],[1]) in PKI, and " Group Member Signing Key Revocation" in group signature management ([3]). To clarify the difference of these "Revocation" from our signature revocation, we briefly compare them in Table 1.

Fair exchange of digital contents is another important issue in network communication. It ensures that the participating parties can get what they negotiated to exchange in the end. Usually the participants don't trust each other, so there often exists a Trusted Third Party (TTP), that helps to complete the protocol. An obviously efficient approach will be the TTP only appears when needed i.e. the optimistic fair exchange. This can help the two parties to reach mutual consent.

* 4-6-1, Komaba, Meguro-ku, Tokyo, 153-8505 Japan Information and Systems, Institute of Industrial Science, University of Tokyo, zhang@imailab.iis.u-tokyo.ac.jp
† Internet Technology Tokyo Research Laboratory IBM Japan Ltd. kudo@jp.ibm.com
‡ * kanta@iis.u-tokyo.ac.jp
§ * imai@iis.u-tokyo.ac.jp

	Private Key Revocation*	Certificate Revocation	Signature Revocation
Things to revoke	Group signing key	Certificate of public key	Message-wise signature
Implementation approach	Enable block	CRL or CRT or CRS	Revocation Token
Key reusability	No	No	Yes

*: In Broadcast Encryption

Table 1: Different revocation

2 Our contribution

While a lot of research has been done on certificate revocation and key revocation in PKI, there is still not study concerns the instance revocation of digital signature. In a large amount of applications, the signer of the signature needs to invalidate the signature legally. Another big issue is how to control the revocation in a flexible way, so that the signer can revoke the signature whenever he wants and no matter how long it lasts, he can still prove to a third party, say, a judge, that the signature has been revoked in a legal way.

In this paper, for the first time we investigate the problem of signature revocation, a matter of instance management regarding digital signatures. We formulate the categories of different applications and propose an efficient scheme based on the hash function. Using our models of signature revocation, a digital signature can be revoked by the signer and respective receiver at any time legally. We would also like to point it out that our scheme enjoys semantic security in the context of random oracle model and public key assumption. We also give another construction of signature revocation based on the Designated Confirmer Signature (DCS). To further explain our work, those solved in this work are listed in Table 2.

3 Preliminary

3.1 notations

A digital signature scheme e.g. [11], is a 3-tuple algorithm containing such elements:

1. Generation algorithm: $\mathsf{Gen}()$, in which a signing key and a verification key pair is generated according to a user Alice, where the signing key will only known by the signer and verifiers will have the verification key.

2. Signing algorithm: $\mathsf{Sign}()$, for an input message M, a Signer Alice using the signing algorithm outputs $\mathsf{Sign}(\mathsf{M})$, where $\mathsf{Sign}(\mathsf{M})$ will be called the signature on M. Especially, $\mathsf{Sign}_{\mathsf{A}}(\mathsf{M}_1)$ is the signature on M_1 of a signer Alice, $\mathsf{Sign}_{\mathsf{B}}(\mathsf{M}_1)$ is the signature on M_2 of a signer Bob, etc..

3. Verification algorithm: $\mathsf{Ver}()$, given a message M, a certain verification key, K_{v} of a signer Alice, and a respective signature Sign(M) on message M, $\mathsf{Ver}(\mathsf{M}, \mathsf{Sign}(\mathsf{M}))$ outputs a boolean value $\mathsf{B} \in (0,1)$ that either rejects or accepts that $\mathsf{Sign}_{\mathsf{A}}(\mathsf{M})$ is a valid signature on M.

For the convenience later, we also list some other tools as follows:

$\mathsf{H}()$ the collision-free hash function, for a random number R of any length, the hash function output a fixed length value $\mathsf{H}(\mathsf{R})$, so that it is computationally hard to find different values R_1 and R_2, where $\mathsf{H}(\mathsf{R}_1)$ equals $\mathsf{H}(\mathsf{R}_2)$. We also assume that given a hashed value $\mathsf{H}(\mathsf{R})$, it is computationally hard to find R' so that $\mathsf{H}(\mathsf{R}')$ equals to $\mathsf{H}(\mathsf{R})$. Typically examples can be MD5, SHA-1 etc..

$\|$: string concatenation. For an arbitrary string M, and another string R, $\mathsf{M}\|\mathsf{S}$, means the concatenation of the two strings. The new string will have the length of $\mathsf{M} + \mathsf{S}$, and respective digits of M and S.

3.2 Sketch of the idea

Definition 1 Universal security of signature revocation: once the signature is revoked, then no one can prove the original signature to anyone the validity of the signature, then this kind of revocation will be called universally secure signature revocation.

Remark 2 In fact the universally secure revocation deprives the signature of the verifiability.

We have noticed that for the normal public key system it is hard to have universally secure revocation. However, like the certificate revocation list, we introduce other kind of method that can prove some proof to the players that in a prospective dispute, the players can protect themselves by submitting the proof to a judge.

Evidently for a revoked signature the signer should be able to deny it, hence for any verifier of the signature, the original the non-repudiatable property of the the signature was no longer available. Oppositely if non-repudiatable property of the the signature was removed, then then signature is in fact revoked. Consequently, the problem of the revocation regarding the digital signature can be equivalently expressed as removing the non-repudiatable property from the signature.

In the concrete we shall introduce a new tool named Revocation Token to help to achieve removing the non-repudiatable property from the signature. We notice that a single signer usually don't have the right to revoke his signature, if he commits something or promises some duty. So usually we provide a negotiation phase for both the signer and the receiver. So that the revocation can only take place if both of them agree

	Public Key based signature scheme	Designated confirmer Signature
Approaches	Revocation Token	An additional revocation protocol
Universal Security	No	Yes

Table 2: What will be solved in this work

to the revocation. Of course, for certain applications, e.g. some free service provider's contract, the provider should have the right to invalidate the contract any time. In such occasions, only the signer's consent will be enough for the revocation. So we should also include it into our scheme.

Definition 3 Revocation Token is such a matter that by proving the approval and authentication of the related players it convinces a third party such facts that the signature has become invalid.

We can base the signature on the invalidity (invisibility) of the Revocation Token in the signing phase. As long as the Revocation Token is invalid (invisible), the signature will be valid. For revocation, just uncover the Revocation Token, which will equivalently invalidate the signature.

Another approach in reference to Undeniable Signature or Designated Confirmer Signature is in a different category. If the respective signer or confirmer should refuse to help receivers to verify the signature, the signature would be factually revoked. We also consider this specific type of signature and give a complete formalized protocol to realize the revocation. According to the nature of Designated Confirmer Signature, this approach is of universal security.

4 Revocation Token

4.1 Basic logic

We can make such a construction utilizing the hash function the hash chain. We can also extend it to multilayer signature revocation, which can revoke a number of signatures at the same time. Here we first give the construction, then we prove it to be semantically secure under proper assumptions.

Suppose a signer Simon signs the message together with the concatenation of the hash value of a randomly selected number, and he will keep it secret until the signature is revoked: such that, the signature will be

$$\mathsf{Sign}_S(M \| H(R))$$

Simon sends it to a receiver Rata. Rata can verify Simon's signature on $M \| H(R)$ as usual after she receives it, therefore, authenticates M in normal way verification of digital signature, however, she can't know any information about R, for the un-invertibility of the hash function.

When there is a need to revoke the signature, Simon and Rata need to perform the following protocol together:

1. They negotiate on the revocation of a certain signature of Simon $\mathsf{Sign}_S(M \| R)$, and
2. Simon sends the random number R to Rata.

If later there arises a dispute, the court will call on Simon and Rata can send the random number R to the judge, judge will compute the hash function on R, If $H(R)$ corresponds with the one in the signature, then the signature is revoked. Or the signature is still valid if there is no extra time limit in the signature.

However, as we have mentioned before, in the occasions that the signer commits some duty to the receiver, sometimes he shouldn't be allowed to revoke the signature if only he wants. In other words, the revocation of signature management shouldn't be controlled only by the signer. The receiver should also have the power to control the revocation. But if we merely use the strategy stated above, the receiver will never be able to call on a revocation.

So as a formal approach, we propose such a construction that combines two hash values independently generated by Signer and Receiver respectively. As including the hash value of their secret numbers in the signatures, it can be regarded for both of them as an agreement to the revocation option in advance. They keep the number secretly until the revocation phase. Thus not only the Signer but also the Receiver has part of the right in the revocation process. In the revocation process they just exchange the two random number. In case of dispute, the judge will call on them to give out the two random numbers. If either of them can submit the number, then the signature will be regarded as revoked. We will give the formal construction of the Revocation Token later as well as security discussion.

However, another problem occurs that one player may get advantages in the exchange phase. Here a fair exchange sub-protocol is needed in the exchange phase, so that in the end either each player gets the other's secret number, or no party gets the other's number, and if both players are honest they must result in a fair exchange. We will define the security of a fair exchange as this:

Fairness After the exchange, if both the players get the other's digital contents or neither of them gets the other's digital contents, we shall call the exchange fair.

Completeness If both of the players are honest, they must get fair exchange.

Definition 4 If the exchange protocol satisfies such requirements(fairness and completeness), we shall call the exchange protocol secure.

In [2], there are ready solutions for most public key systems. Here we just make a small change to it that the digital signature there will be replaced by a random

hash value R. For details readers are encouraged to refer the original paper.

4.2 Detailed protocol

In this section we give a protocol with the construction of hash values. For the receiver should have part of the right in the revocation phase, we also include the receiver in the signing phase to submit a hashed value of his secret number.

Definition 5 We define that $(H(R_S), H(R_R))$ is the Revocation Token of the signature

$$Sign(M \| H(R_S) \| H(R_R))$$

The signing phase:

1. Rata selects a random number R_R from a specific set $\{S_R\}$, and hashes it to get the value $H(R_R)$, then together with the signing request, she sends $Req(M, H(R_R))$ to Simon.

2. Simon receives the request $Req(M, H(R_R))$ from Rata, extracts $H(R_R)$ from the request, selects a random number R_S from $\{S_S\}$, calculates the hashed value $H(R_S)$, then he signs the message M together with $H(R_S)$ and $H(R_R)$. He forwards the second extra value $H(R_S)$ and the signature $Sign(M \| H(R_S) \| H(R_R))$ to Rata, then quits.

3. Rata receives the respective hashed value $H(R_S)$ and the signature $Sign(M \| H(R_S) \| H(R_R))$ on the message M from Simon. She verifies the signature with the verification key K_P and verification algorithm, if

$$Ver(K_P, Sign(M \| H(R_S) \| H(R_R))) = 1$$

then she accepts $Sign(M \| H(R_S) \| H(R_R))$ is a valid signature on message M, then quits.

The revocation negotiation phase:

In this phase, Simon and Rata discuss to reach the agreement for a revocation. Since the signature will be no longer valid after the negotiation, all the decisions will be made on behalf of the two players. If each of them agrees to invoke the revocation, then terminates. The revocation will only take place after this phase.

The Revocation Token construction phase:

The signer who wants to revoke his past generated signature initiates the signature revocation. Here according to the fair exchange logic, the random number will be first converted into a verifiable encryption ([4],[2]). Since there is only two players, and they don't trust each other will do the exchange honestly, they ask the Trusted Third Party (TTP) to assist in the exchange. Without losing generality, we suppose Rata moves first.

1. Rata encrypts her secret random number R_R in a verifiable encryption way that with only negligible probability that she will success in cheating Simon that with a number different from R_R, Simon will accept the encryption.

2. Simon receives the verifiable encrypted number R_R, then he sends Rata his secret random number R_S. To avoid possible eavesdroppers in the middle, he may encrypt it in a normal way, i.e., no need verifiable. If he changes his mind at this very point, he can call on the abort protocol and quits.

3. Rata receives Simon's number R_S, and verifies whether this value accords with the one she receive in the exited signature. If she succeeds, she stores the value together with her secret value, and sends her secret value to Simon. If she fails in verifying the value Simon sent her in the second step, then she contacts the TTP for an abortion. Otherwise she may call on a resume sub-protocol.

4. Simon receives Rata's number, he does the same: first verify the correctness of the value, if it corresponds to the one Rata sent him in the signing phase. If the verification succeeds, then he quits. Otherwise he informs the TTP call on the resume sub-protocol.

Protocol Abort:

1. A player sends the request of abortion to the TTP $Req(Abort)$.

2. TTP receives from the player the request of abortion $Req(Abort)$, he searches his database for the record of abortion $Record(Abort)$, if it has been aborted before, and if there is no record of resume, either, then he sends to the requester the approval $Appr(Abort_{Rata})$, writes a record to his database of abortion $Record(Abort)$ and quits. If the protocol has already been resumed before, then he will send to the player the Revocation Token $(H(R_S), H(R_R))$.

Protocol Resume:

1. Simon sends to the TTP his secret number R_S, the original signature $Sign(M \| H(R_S) \| H(R_R))$, the verifiable encryption of Rata's secret number $VE_R(R_R)$ together with the request of resume $Req(Resume)$.

2. TTP receives from Simon $VE_R(R_R)$, the original signature $Sign(M \| H(R_S) \| H(R_R))$ and the request of resume $Req(Resume)$, he searches the record $Record(Abort)$ in his database, if there is an abortion record before, then he sends to Simon the abortion information $Appr(Abort_{Simon})$ and quits. Otherwise he decrypts the $VE_R(R_R)$ and gets the number R_R. Or he can get it from the database if the protocol has already been resumed before. Then he will sends the approval of Resume $Appr(Resume_{Simon})$ together with the Revocation Token $H(R_S), H(R_R)$ to Simon, saves $Record(Resume)$ in his data-base (if necessary the Revocation Token $(H(R_S), H(R_R))$, too) and quits.

In the case Rata calls on a resume, she must include her verifiable encryption of her number besides the same things Simon sends to the TTP.

After all the protocols, both Simon and Rata hold the other player's secret number, they can construct the Revocation Token separately.

5 Security analysis

Theorem 6 *The construction of Revocation Token using hash function i.e.* (R_S, R_R) *is semantically secure under the assumption of random oracle model and public key assumption.*

Proof. From our construction of Revocation Token, we know that For a signature

$$Sign(M \| H(R_S) \| H(R_R)),$$

a pair of secret numbers (R_S, R_R) that generate the hashed value which were concatenated with the message in the signature has the properties:

First, with the assumption of public key, we know that it is computationally hard for one to forge a signature of others, in other words, it is also impossible for one to forge another version of revoked signature within finite time, which means no one can forge the hashed value included in the signature, either.

Then we have assumed that the hash function is unable to invert. Under the random oracle model assumption, before one disclose the root number to the other player, he will know no information above it. This means that one can't get arbitrary access to the original number until the other player tells him. Only after one gets the other's number can he put it with his own to have one valid Revocation Token to the particular signature.

Last by secure fair exchange protocol, we can have two players exchange their secret number fairly. No one will gain any advantage in the exchanging phase. Even the one drops from the middle of the exchange protocol and terminates won't be able to leave the other waiting for ever. In such a situation, the other party can call the TTP to decrypt the verifiable encryption of the exchanging number sent in the first step of the exchange protocol.

Complete. ■

After the exchange protocol if there arises a dispute, either Simon or Rata can just show the Revocation Token to the judge. Then with enough convincingness the judge will trust the signature is revoked. Otherwise if none of them can provide the Revocation Token, then no matter one or both of them are cheating, the signature is still valid.

6 Extension and applications

Suppose we have such a hash chain generated by a random seed as root, whose elements are the hashed values of previous one. Then using the same construction as above, we can use one element once in a series of continuous related signatures. It will be very efficient for applications in which the contract (signature) are to be revoked very often and not desirable to return, if each revocation to the signature is related to the last change in the chain. Another type of applications can be branch routines revocation based on only one revocation root. An example of such kind is depicted in figure 1.

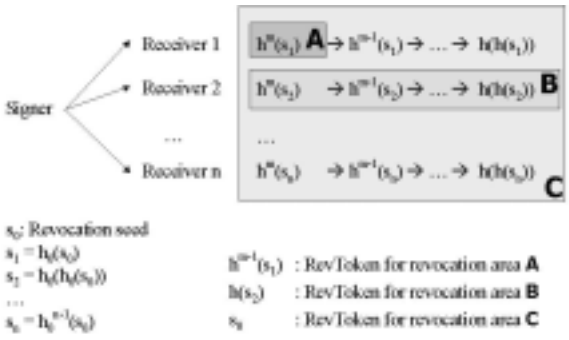

Figure 1: Applications

For a traveller to arrange his course of the trip, he may have to book the flight, the hotel, buy the ticket to the park, rent a car at the spot and etc.. Then with travel agencies, he can make all the reservation in one contract. Later, he may change his idea about part of the arrangement in his travel (e.g. the hotel and/or the date of the flight) or even cancel the whole course. So this time, we can utilize the hash chain value in all the reservation with priority. One revocation will not affect the upper ones, and will only partially revoke the contract (customer's digital signature).

7 Revocation protocol regarding DCS

As we have pointed out that we can utilize the power of DCS as possible approach regarding signature revocation of this category. First, the Confirmer (hereby we shall call her Carol) in DCS is assumed to act as a TTP that can do such things:

- Confirmer can be trusted as linking to exact time source.
- Confirmer can deal with signature revocation status.

We here emphasize that once Confirmer Carol receives a certified revocation request from the signer, Confirmer stops corresponding verification service at later time frame.

The message flow of the revised DCS protocol is given here:

1. Signature Generation: Simon generates signature and sends it to Rata just as the normal DCS signature does.
2. Signature Verification: Rata confirm the signature using the same protocol as DCS based on Zero-knowledge proof.
3. Revocation: As what we have done with the hash function construction, here we divide it into two situations:

Signer-only Revocation As the literality, only the signer Simon can revoke the signature and Simon himself is enough to have the revocation happen. He contacts Carol who will do these for him:

Records the time of his request and from this point every request for verify the signature will be refused, which means the signature is completely revoked.

Negotiation Revocation This requires Simon and Rata discuss the possibility of the revocation. If one party should not agree to revoke the signature, then it will still be valid as long as the time limit of the signature is not due.

Then after both of Simon and Rata agree to invalidate the signature, they will send their authentication to Carol. After Carol receive it from both, she will record the time and add a revocation record together with the proof of consent of both party in her database and any request on the verification of this signature will be neglected.

4. Post-Revocation: If someone attempts to verify a signature, Carol will first search her database for the revocation record. If there is such record in her database, she will send a "revoked" proof to the verifier. Otherwise she will help to verify the signature as usual.

8 Conclusion

In this work we investigate the problem of instance management regarding signature especially the revocation problem. For different types of signatures like the normal public key signature scheme and other verifier-based signature schemes e.g. DCS. One thing to point out is that by using the hash value and hash chain construction, it will be great efficiency in the signature revocation. Future work is likely to be other construction with more applications towards the universal security.

References

[1] W. Aiello, S. Lodha, and R. Ostrovsky, "Fast digital identity revocation, in Advances in Cryptology 佑RYPTO98, Lecture Notes in Computer Science. Berlin, Germany: Springer-Verlag, vol. 1462, pp. 137152. 1998.

[2] N. Asokan, V. Shoup and M. Waidner, "Fair exchange of digital singature," IEEE Journal ON Selected Areas In Communications, vol. 18, No. 4, April 2000.

[3] E. Bresson and J. Stern, "Efficient revocation in group signatures", K. Kim (Ed.): PKC 2001, LNCS 1992, pp. 207-224, 2001.

[4] Jan Camenisch and Ivan B. Damgärd. "Verifiable Encryption and Applications to Group Signatures and Signature Sharing," BRICS, pp. 18-37 December 1998.

[5] D. Chaum, "Zero-knowledge Undeniable Signatures," Proc. of Eurocypt'90, LNCS 473, Springer-Verlag, pp.348-464, 1991.

[6] D. Chaum, "Designated Confirmer Signatures," Proc. of Eurocrypt'94, LNCS 950, Springer-Verlag, pp. 86-91, 1995.

[7] D. Chaum and H. van Antwerpen, "Undeniable Signatures," Proc of CRYPTO'89, LNCS 435, Springer-Verlag, pp.212-216, 1990.

[8] P. Kocher, "On certificate revocation and validation, in Financial Cryptography FC98, Lecture Notes in Computer Science. Berlin: Springer-Verlag, vol. 1465, pp. 172177, 1998.

[9] S. Micali, "Efficient certificate revocation," Tech. Memo MIT/LCS/TM-542b, 1996.

[10] A public key infrastructure for U.S. government unclassified but sensitive applications, Sept. 1995.

[11] R. L. Rivest, A. Shamir, and L. M. Adleman, "A method for obtaining digital signatures and public-key cryptosystems, Commun. ACM, pp. 120126, 1978.

[12] W. Guey. Tzeng and Z. J. Tzeng, "A public-key traitor tracing scheme wit hrevocation using dynamic shares", K. Kim (Ed.): PKC 2001, LNCS 1992, pp.207-224, 2001.