Proc. Compsec '91, 8th world conference on computer security, audit and control, Elsevier, Oxford, 125-134.

# Fail-stop Signatures; Principles and Applications

Birgit Pfitzmann
Institut für Informatik
Universität Hildesheim
Samelsonplatz 1, W-3200 Hildesheim, FRG

## Abstract

Digital signatures are necessary wherever legal certainty is to be achieved for digital message exchange. However, the unforgeability of conventional digital signatures is necessarily based on complexity theoretic assumptions. That is, even the most secure schemes can be broken by an adversary with unexpected computing abilities, e.g., one who can factor unexpectedly large numbers.

Fail-stop signatures improve upon this: They are as unforgeable as the best conventional signatures; but if someone nevertheless succeeds in forging a signature, this can be proved by the supposed signer. Thus one can relieve him from the responsibility for this signature. Additionally, one should stop the scheme or increase the security parameters.

As applications, mainly digital payment systems are discussed. The social and legal advantages of such a scheme are discussed, and a sketch of the construction of practical fail-stop signatures for this case is given (roughly three times the expenditure of a conventional signature scheme).

## 1. Overview

To avoid misunderstanding, Section 2 contains a short explanation of important terms in their most common usage. Section 3 explains digital signatures in general. In Section 4, the problem with conventional digital signatures (i.e., those which are not fail-stop) is shown. Fail-stop signatures are introduced, and their general advantages discussed, in Section 5. Important applications are treated in Section 6, and general remarks on practicability are made in Section 7. Section 8 is an optional section with an overview over the construction of concrete fail-stop signatures. The paper ends with a recommendation for the near future (Section 9).

## 2. Classification

Most cryptographic systems for data protection and security belong to exactly one of the following two classes:

- *Secrecy systems*, i.e., systems to keep message contents secret.
- *Authentication systems*, i.e., systems to ensure the authenticity of message contents and of the sender's address.

("Cryptosystem" often means any cryptographic system, sometimes secrecy system. "Encryption scheme" normally means secrecy system, sometimes any cryptographic system. One should not be confused by the fact that basic functions like RSA and DES can serve for both secrecy and authentication: For the two cases, they must be employed in different ways.)

Within each of the two classes, one distinguishes two types according to the key distribution:

- *Symmetric* systems, where sender and recipient of a message must have the same (secret) key.
- *Asymmetric* systems, where sender and recipient have different keys. One key is secret, the other one public.

## 3. Digital Signature Schemes

Digital signature schemes, both conventional ones and fail-stop ones, are asymmetric authentication systems.

Here it is the sender, called signer, who must keep his key *S* secret. All other participants only have a public test key *T*. The signer has generated *T* together with *S* and then published *T* (for remarks on actual key distribution see Section 8); one must not be able to sign with *T*. Everybody who receives a signed message, i.e., one which passes the test with the test key, knows that only the owner of the corresponding secret key could have produced this message.

Thus a digital signature can have the same effect for a digital messages as a handwritten signature ought to have for a normal message: It guarantees that the supposed sender really is the sender of exactly this message (or has at least authorized it). The recipient can also prove this to third parties, e.g., in the case of dispute, to a court: The third party also knows the public test key *T* and knows that only its owner can be the sender of the message.

(This is the difference to symmetric authentication systems, e.g., derived from DES (see, e.g., [4]): There, sender and recipient have a common key, which they keep secret from everybody else. Now each of the two can send authenticated messages to the other one. However, even if a court knew which key they are using, it would be unable to decide which of the two sent a particular message.)

Also note that digital signatures must be highly dependent on the message contents, so that, if someone changes a bit of a signed message, the signature does not fit any more.

## 4. Problem with Conventional Digital Signatures

### Not Provably Secure

The problem with conventional digital signature schemes such as RSA [12] and ElGamal [5] is that in a strict sense, they are not provably secure. With all currently known signature schemes, one needs a "cryptographic assumption", e.g., the "factoring assumption" which says that factoring numbers whose prime factors are large is difficult. However, on principle, factoring numbers into primes can be done in a unique way. Thus the security only relies on the fact that a would-be forger has not enough time and no sufficiently good algorithm to carry out the factorization. The same holds for the "discrete logarithm assumption" of the ElGamal scheme.

In addition, with RSA and ElGamal, it is not even clear if they are secure on their respective assumption. In particular, the security of fast hash functions, which are often used to speed up signing, is even more doubtful than that of the basic schemes.

However, there exists an improvement: The signature scheme GMR is at least provably secure on the factoring assumption [7, 6].

### Assessment of Cryptographic Assumptions

Of course, it is to be hoped that the cryptographic assumptions on which important conventional digital signature schemes rely are true. They are the more trustworthy the more intensively they have been examined, i.e., the longer scientists have tried in vain to find fast algorithms to break them. Especially the factoring assumption, but also the discrete logarithm assumption, is sufficiently interesting for mathematicians so that many studies take place. (In this respect, cryptographic systems resemble violins: If they are still unbroken at all, the oldest ones are the best ones. But of course, in order to deal with increasing computing power and better breaking algorithms, one must increase the so-called security parameter often enough, e.g., the size of the numbers used which hopefully cannot be factored. For instance, a prudent choice of an RSA modulus would be at least 1000 bits long, and similar choices should be made for the discrete logarithm assumption in any group; see [9] for an outlook on some new or forthcoming algorithms.)

### Consequences in the Case of Forgery

If a signature scheme is broken although one had reasons to hope it would not, the supposed signer of a message with a forged signature is defenceless: The forged digital signature looks exactly like an authentic one. If it is presented to a court, the court decides, still with the public test key $T$, that the signature is valid. Thus the supposed signer will be held responsible. (The recipient of a signed message, however, is absolutely secure: If he has checked that the signature passes the test with the public test key, he knows that it will also do so in court, no matter if it is authentic or forged.)

At least this is the technical view of what a court should do. A real court is not obliged to comply with this view. (At least German courts are not obliged to acknowledge a handwritten signature either.) It could believe the protestation by the supposed signer that the signature is a forgery, the more so since the signer can adduce that the signature scheme relies on an unproven assumption, no matter how long it has been examined.

However, if this were the case, recipients would not be secure at all any more: Even if the assumption is perfectly correct and no signature is ever forged, the real signers could now deny their signatures in court, just like the supposed signer above. It is impossible for the court to distinguish the two cases. Hence, if recipients had to reckon with signatures being rejected in court, nobody would want to accept digital signatures from anybody whom they did not trust. But the possibility of protecting mutually suspicious senders and recipients, e.g., when signing a contract of purchase, was just the main advantage of digital signatures over symmetric authentication systems (besides the simpler key distribution).

## 5. Fail-stop Signatures

**Distinguishing Feature**

With fail-stop signatures, too, unforgeability relies on an assumption. In the concrete scheme sketched in Section 8, called hiding scheme, this is the factoring assumption. (There exists an alternative based on the discrete logarithm assumption.) It has been proved that forgery is impossible on this assumption [11]. (This was the property that only GMR had among the signature schemes mentioned above.)

The new feature of fail-stop signatures is: If a signature is forged nevertheless, the supposed signer can *prove* that it is a forgery. More precisely: He can prove that the underlying assumption has been broken, e.g., by showing the factors of a number which he could not have factored if the assumption were true. Thus one can relieve him from the responsibility for this signature. For details of how this is achieved, see Section 8.

**Error Probability versus Cryptographic Assumption**

It should be mentioned that the proof of forgery fails with a very small probability, which can be chosen arbitrarily, e.g., as $2^{-100}$. However, this does not matter much:

First, a small error probability is unavoidable with digital authentication systems (including symmetric ones): The signer only uses a finite number of bits of secret information. If a forger guesses them all correctly, nothing can be done. Thus one calls an authentication system "unconditionally secure" if it has been proved that a forger

- cannot forge better than by blind guessing, and
- cannot check whether he has guessed correctly (except by going to court or asking the true signer).

In this sense, it is unconditionally secure that forgery of fail-stop signatures can be proved.

(A similar situation with handwritten signatures would be that a forger engages a horde of monkeys to produce scribbles. He must select some of those as signatures, without even looking if they resemble the authentic signature.)

Secondly, probabilities of other failures are larger, in particular the probability that a wrong signature is accepted due to a hardware fault, which has the same effect, and indeed the probabilities of most risks of daily life.

The difference between a scheme with a small error probability and one that relies on an unproven assumption, like conventional digital signatures, is that with the latter it is not *known* if the probability of forgery is so small; instead of engaging a horde of monkeys, the forger *might* have constructed an automatic signing machine.

One should also mention that the fail-stop feature does not help if the key is physically stolen. This, too, is unavoidable with all digital authentication systems: Once a forger has the signer's complete secret information, it is impossible to distinguish him from the signer by digital means. Thus this problem must still be dealt with by physical measures.

**Advantages**

With fail-stop signatures, a court can always decide if a signature is a forgery or not (at least in the digital world), in contrast to conventional digital signatures.

Also, already after the first forgery, all the other participants in the system and the system operator (if there is one, e.g., a bank using the signatures in a payment system) know that the signature scheme has been broken. Hence one can stop it before much further damage arises, or at least increase the security parameter. This explains the name "fail-stop": In fault tolerance, it is used for systems which, in the case of failure, at least end up in a fixed state, e.g., railway signals which drop into the stop!-position when a wire breaks.

Of course, however nice it is to be able to recognize forgeries, it does not solve all problems: If the recipient of a forged signature is not the forger himself, he has relied on the signature. If the signature becomes invalid because it is recognized as a forgery, he will lose something, depending on the application. However, if the damage is only financial, an insurance might pay it: In contrast to conventional signatures, the insurance can now incontestably check if the case of damage has arisen, just like a court; and the damage only occurs if the assumption is broken.

## 6. Application in Payment Systems

**Use**

Digital payment systems might become the most important application for fail-stop signatures. More generally, the following holds for any system where a large organization like a bank or a PTT, wants to use digital signatures for message exchange with its clients.

There, clients would use fail-stop signatures, the bank conventional digital signatures. Then the clients are unconditionally secure, i.e., they need not trust the cryptographic assumption (because they are signers of fail-stop signatures and recipients of conventional signatures, and in both roles one is unconditionally secure, as described).

**Why Like This?**

Why doesn't one make the bank unconditionally secure instead?

First, because the bank is the stronger partner: It can choose the conventional signature scheme and the security parameter and thus provide for its own security. Also, it can inform itself about how trustworthy the cryptographic assumption is, whereas many clients will already be discouraged by the name of a factoring or discrete logarithm assumption.

Secondly, the bank doesn't even need a proof of forgery if its own signature is forged: It is the system operator, thus it can stop the system by itself.

Thirdly, the bank might even have an advantage over a digital payment system using conventional digital signatures only (besides the possibility of advertising with the higher security for clients): The fact that forgeries can be incontestably recognized might increase the legal acceptance of the system, i.e., in the more likely case that the cryptographic assumption is *not* broken, the bank might be more secure that the clients' signatures are indeed recognized in court.

# 7. Usable in Practice?

Fail-stop signatures have not been implemented yet. Also, they are not the kind of scheme that anybody would buy, e.g., to make their local networks more secure, but they are meant for systems of legal relevance and high security requirements such as payment systems. For this particular case, the theoretical development is so far advanced that one can consider an implementation. Using fail-stop signatures as described in Section 6, and with the concrete scheme sketched in Section 8, the expenditure (time, storage, and communication) is not more than three times that of a system where the clients, too, would use the conventional signature scheme chosen by the bank, even if that is RSA with a fast hash function.

# 8. Overview over the Construction of Concrete Fail-stop Signatures

### Basic Idea for the Construction of Fail-stop Signatures

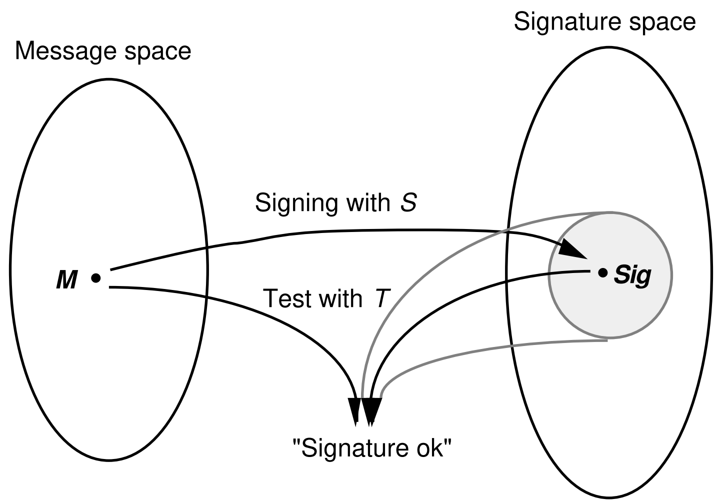


**Figure 1** Basic idea for the construction of fail-stop signatures

Like with most conventional signature schemes, the signer, using his secret key *S*, obtains exactly one signature *Sig* for each message *M*; and if one applies the test with *T* to the pair of *M* and *Sig*, the result is "ok".

The new idea is that, apart from *Sig*, there is a set of other values (the grey circle in Fig. 1) which pass the same test and are therefore acceptable as signatures on *M*. The construction guarantees: If the cryptographic assumption is true, nobody except for the signer can find any acceptable signature, and the signer just this single one. (In particular, the grey circle is still small in comparison with the whole signature space.)

Now, if a forger breaks the assumption, he may be able to compute the grey circle by inverting the test function, but he cannot find out which of the values in there is the authentic signature *Sig*. (For this to be true, two more conditions must be fulfilled: A forger might try to find out *S* first. Thus there must also be a set of other possible secret keys which fit *T*, and these yield exactly the other acceptable signatures. Also, other signed messages must not reveal information about the new signature.) A forger must therefore guess one out of all these values correctly in order to use the authentic signature for his forgery.

Hence, if one is willing to tolerate an error probability of, say, $2^{-100}$, one must choose the size of the grey circle as $2^{100}$.

If the forger guesses wrong, the signer can present his authentic signature and show that it looks different. Since even the signer can only compute one signature if the cryptographic assumption is true, the fact that there are now two different signatures on the same message is proof that the assumption has been broken. In the scheme sketched in the following sections, the signer can also use the two signatures to compute the factorization of a number whose factors he did not know before. Thus the connection with the factoring assumption is even more obvious.

**Frame Construction for Payment Systems: 3-Phase Protocol**

The problem with current concrete fail-stop signatures in the general case is that they grow quickly with the length of the message. However, especially for payment systems where fail-stop signatures are to be used as in Section 6, the following frame protocol ensures that the clients with their fail-stop signatures only sign 1-bit messages (leaving most of the work to the bank with its more efficient conventional signatures). This makes the system practical.

When a client $C$ wants to give the bank his $i$-th order, e.g., a remittance, this is done in 3 phases:

1. *Instruction*: The client tells the bank his $i$-th order, e.g., "Remit 17 £ to client $D$".
2. *Confirmation*: The bank confirms the instruction with a signed message (in the conventional signature scheme), e.g., "Client $C$'s $i$-th order is: Remit 17 £ to $D$."
3. *Ok*: If the confirmation is correct, the client signs one single bit, which means "ok", with his $i$-th fail-stop signature.

About Phase 3, one should note that current fail-stop signatures are automatically numbered. Phase 1 may be augmented by some weak form of authentication.

If there is a dispute about if the bank has carried out the $i$-th order wrong, a court can decide as follows: First the bank must present the client's ok, which allowed the bank to carry out the $i$-th order at all. The client is given the opportunity to prove that this is a forgery. Otherwise, he must present the signed confirmation of the bank. The content of this confirmation is regarded as the correct order.

As promised, the clients are unconditionally secure: Their security only depends on their fail-stop signatures and that they store the correct confirmation. The bank is secure on the cryptographic assumption, which it could choose itself: In order to cheat the bank, someone must either forge the bank's confirmation or present a proof of forgery for a fail-stop signature.

More generally, any organization can use this 3-phase protocol to exchange signed messages with its clients. The limitation is that a client of several organizations needs a different key with each one (so that it is clear for whom an "ok" was). Thus individuals should only be clients of a limited number of organizations. With payment systems, this should be no problem.

One can also extend the protocol to a 5-phase protocol (using the organization as an intermediary) where clients can exchange signed messages so that both are unconditionally secure, and the organization does not see the message contents. This is particularly useful for untraceable payment systems.

**Simple Fail-stop Signatures: Hiding Scheme for "Ok"-Messages**

The hiding scheme is based on a so-called *bundling function* $g$. This is a one-way function (i.e., $g$ can be computed efficiently, but on the cryptographic assumption, its inverse cannot) with two additional properties:

A. In each case, $g$ maps a certain number of values ($2^{100}$ in the example) to the same image.
B. However, on the cryptographic assumption it is impossible to find two values with the same image.

If the client wanted to sign a bit "ok" just once, one would use a bundling functions as follows: (Remember that in the 3-phase protocol, one need not even distinguish bits "0" and "1".) In advance, the client chooses an arbitrary value *Sig* as the signature and, at the same time, as the secret key $S$. He publishes the image $T = g(Sig)$ as his public key. Later, when he gives his ok by revealing *Sig*, everybody can test it $g(Sig) = T$. This ensures the properties depicted in Fig. 1: As long as the cryptographic assumption holds, nobody can forge a signature because of the one-way property of $g$; and because of Property B, the client knows just the single signature *Sig*. And even if the assumption has been broken, Property A ensures that a forger must guess *Sig* among the $2^{100}$ preimages of $T$.

If this idea is used in the 3-phase protocol, each time a new key $T_i$ for the $i$-th order can be exchanged within the previous order: The client chooses $Sig_i$ before the ($i$–1)-th order. He tells the bank $T_i$, and gets it confirmed by the bank, as a part of his ($i$–1)-th order.

**Concrete Bundling Functions**

One obtains concrete bundling functions for the factoring assumption and the error probability $2^{-100}$ as follows: The bank chooses two large prime numbers $p$, $q$ with $p \equiv 3$, $q \equiv 7 \bmod 8$ and publishes $n = p \bullet q$.

The domain consists of pairs $(z, d)$, where $z < 2^{100}$ is an integer and $d$ a number modulo $n$ with two additional properties:

(1) $d < n/2$.
(2) The Jacobi symbol of $d$ is 1 (see, e.g., [8]).

An auxiliary function $g'_n$ is defined by

$$g'_n(z, d) = 4^z \bullet d^{2^{100}} \bmod n.$$

If $g'_n(z, d) < n/2$, then $g_n(z, d) = g'_n(z, d)$, else $-g'_n(z, d) \bmod n$.

For a more detailed description see [11], for formal proofs [1] and [10].

(Specialists may have noticed that the bank, who knows $p$ and $q$, can invert the function $g$. This doesn't matter, although the bank could thereby forge the clients' signatures, because the clients could prove the forgery. Thus, according to the 3-phase protocol, the bank would only cheat itself and stop its own payment system.)

Similarly efficient bundling functions exist for the discrete logarithm assumption [2]. (Additionally, in this case no way is known how the bank could forge the clients' signatures.)

The efficiency depends on which conventional signature scheme the bank trusts. The function $g'$ itself can be evaluated with 100 multiplications, i.e. faster than an RSA operation. Thus, even if the bank uses RSA with a fast hash function, the 3-phase protocol is only by a factor of about 3 less efficient than if the clients, too, would use RSA.

**Key Distribution**

Key distribution for a digital signature scheme is not trivial, but easier than for symmetric systems. In practice, one usually assumes a trusted key distribution center. However, in systems of legal relevance, it is better not to have any single instance whom citizens must trust. (For instance, in constitutional states with separation of powers, even the decisions of courts are usually publicly verifiable and can be appealed against.)

Luckily, this situation can be improved upon without much overhead: The signer still gives his $T$ to a center (note that everybody should definitely generate their own keys), but the center must sign which $T$ it is to distribute. For this, the center can already use its conventional digital signature; the bank's public key, as the starting point of all digital authenticity, must be distributed reliably outside the network anyway. Everybody who asks the center for a certain $T$ also demands a signed response. In the case of a dispute about a public key, all parties must show the center's signature on their version. If two different versions have been signed, the center is held responsible. Again, the clients are unconditionally secure, and the center on the cryptographic assumption.

## 9. Recommendation

This description that more security can be achieved than with conventional digital signatures is not meant to give the impression that those were useless. At the current state of the photocopying technique, they are probably even much more secure than handwritten signatures.

Of course, one should continue to develop more secure signature schemes. (Even a scheme very similar to signatures where everybody is unconditionally secure from forgeries has already been proposed; unfortunately, it is not practical at present [3].) However, the planning stage of applications of the size of payment systems is long, as experience with the development of applications of RSA shows. Thus one should better quickly start using, e.g., RSA-based applications, which are almost ready, than to be completely insecure or to use symmetric authentication only. However, one should remember to choose security parameters prudently, see Section 4.

## Acknowledgements

It is a pleasure to thank Manfred Böttger, David Chaum, Andreas Ort, Andreas Pfitzmann, and Michael Waidner for many helpful comments.

## References

1. Gerrit Bleumer, Birgit Pfitzmann, Michael Waidner: A remark on a signature scheme where forgery can be proved; *Eurocrypt '90*, LNCS 473, Springer-Verlag, Berlin 1991, 441-445.
2. David Chaum, Eugène van Heijst, Birgit Pfitzmann: Cryptographically Strong Undeniable Signatures, Unconditionally Secure for the Signer; Interner Bericht 1/91, Fakultät für Informatik, Universität Karlsruhe 1991 (Revision appears at Crypto '91).
3. David Chaum, Sandra Roijakkers: Unconditionally Secure Digital Signatures; *Crypto '90*, Santa Barbara 1990, Abstracts, 209-217.
4. Donald Davies, Wyn Price: *Security for Computer Networks, An Introduction to Data Security in Teleprocessing and Electronic Funds Transfer*; (2nd ed.) John Wiley & Sons, New York 1989.
5. Taher ElGamal: A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms; *IEEE Transactions on Information Theory* 31/4 (1985) 469-472.
6. Dirk Fox, Birgit Pfitzmann: Effiziente Software-Implementierung des GMR-Signatursystems; *Verläßliche Informationssysteme (VIS'91)*, Informatik-Fachberichte 271, Springer-Verlag, Heidelberg 1991, 329-345.
7. Shafi Goldwasser, Silvio Micali, Ronald Rivest: A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks; *SIAM J. Comput.* 17/2 (1988) 281-308.
8. Evangelos Kranakis: *Primality and Cryptography*; Wiley-Teubner Series in Computer Science, B. G. Teubner, Stuttgart 1986.
9. Brian A. LaMacchia, Andrew M. Odlyzko: Computation of Discrete Logarithms in Prime Fields; *Designs, Codes, and Cryptography* 1/1 (1991) 47-62.
10. Birgit Pfitzmann, Michael Waidner: Formal Aspects of Fail-stop Signatures; Interner Bericht 22/90, Fakultät für Informatik, Universität Karlsruhe 1990.
11. Birgit Pfitzmann, Michael Waidner: Fail-stop Signatures and their Application; *Securicom 91; 9th Worldwide Congress on Computer and Communications Security and Protection*, Paris 1991, 145-160.
12. Ronald Rivest, Adi Shamir, Leonard Adleman: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems; *Communications of the ACM* 21/2 (1978) 120-126.