Validation of Electronic Signatures

White Paper

Hans Nilsson, iD2 Technologies, Hans.Nilsson@iD2tech.com

Denis Pinkas, Bull, Denis.Pinkas@bull.net

January 27, 1999

Disclaimer

This paper represents the opinion of the authors. It may or may not represent the opinion of their respective companies.

Purpose of this paper

The purpose of this paper is to serve as a tutorial, and as input for the discussion of a common validation model for electronic signatures.

The paper describes the technology to be applied and records needed to support signature validation at different times after the creation of the signature. It is focusing on the situation where the receiver of a document later needs to provide evidence that a certain individual has signed a document, and this individual denies this. This security service is known as "non-repudiation".

In this paper, we are using the term **signer** for the person who creates and signs an electronic document. The signed document is intended for a **recipient** to validate and act upon.

Validate or verify?

The PKI community uses words inconsistently when describing what a certificate user does to make certain that a digital certificate or a signature can be trusted. Usually, we say "validate the certificate" but say "verify the signature." Too often, however, verify and validate are used interchangeably. Here, we recommend a rule to make usage consistent and also align it with both generally accepted PKI community practice and the dictionary.

Use **validate** when referring to a process intended to establish the soundness or correctness of a construct, like a public key certificate or a certification path. Use **verify** when referring to a process intended to test or prove the truth or accuracy of a fact or value.

In other words, we verify atomic truths, but we validate data structures, relationships, and systems that are composed of or depend on verified items.

Validation of signatures

The retrospective validation of digital signatures can be broken down into distinct problems, or regimes, depending upon the use of the digital signature mechanism and the time that has elapsed since the signatures were created.

A digital signature may be used either for peer entity or data authentication purposes, or as an electronic signature for non-repudiation purposes. Usage in the context of peer entity or data authentication is not further discussed in this paper.

Usage of digital signatures in the context of creating electronic signatures for non-repudiation can be seen in several time frames:

1. near term: the validation is performed soon after the generation of the signature and while all the certificates and CRLs required to validate the various signatures are current and generally available. At that time the recipient must make sure that he has obtained all the data he will need later on for long term validation. If the signer has not supplied all the information needed, the receiver will need to collect it at the time of validation.

2. long term: the validation is performed after expiration of the certificate used at the time of generation of the signature, and either:
    - after one change of the certification key originally used to issue that certificate;
    - after several changes of the certification keys from the chain of certificates to be used to validate the signature;
    - after several changes of one of the self-signed certificates used to validate the certification path.
3. archival: the validation is performed after the time when the cryptography used is no longer secure. At this time is may be possible to derive private keys from public keys, or to generate hash collisions.

This paper describes the technology to be applied and records needed to support signature validation in each of the regimes in the context of non-repudiation.

Near term validation of electronic signatures

The need for identifying the certificate

It is the responsibility of the CA to make available in repositories all the information needed to validate any signature by any unexpired certificate it has issued. This includes all unexpired certificates and all CRLs on which any current certificate might have appeared.

In order to validate an electronic signature, the recipient must obtain the **certificate intended to be used by signer** at the time of the signature, and must also be sure that the user was the only one allowed to use the private key associated with that certificate.

Some signers will be able to get different certificates containing the same *public key* from different CAs or even from the same CA. The prime advantage is that a single private key can be used with all of them, which is an advantage when a smart card is used to protect the private key, since the storage of a smart card is always limited. When several CAs are involved, each certificate may contain a different identity, e.g. as a private individual or as an employee from a company. When the same CA is involved, additional attributes like roles may be added to the identifier in order to explicitly mention a role carried by the person. In this way it becomes necessary to find out which of the certificates that was intended to be used.

In order to identify unambiguously the certificate to be used for the validation of the signature, an **identifier of the certificate from the signer** must be part of the signed data. Many current schemes simply add the certificate after the signed data and thus are subject to substitution attacks.

A further advantage of including the identifier of the certificate is to counter a threat that could come from a "false" CA, which could issue a certificate to someone with the public key

of someone else. If the certificate from the signer was simply appended to the signature, and thus not protected by the signature, anyone could substitute one certificate with another and the message would appear to be signed by someone else.

Another technique to counter this threat has been identified, although not as reliable as the previous technique of including an identifier of the certificate in the signed data. This technique mandates all CAs to perform a Proof Of Possession (POP) of the private key at the time of registration. The problem with that technique is that it does not provide any guarantee at the time of validation and only some proof "after the event" may be obtained, if and only if the CA keeps the POP in an audit trail.

It should be noted that the technique of including the identifier of the certificate in the signed data also handles the situation where the CA key actually gets revoked (even compromised). Otherwise, after a CA certificate revocation, all signers might immediately try to repudiate their signatures using the following line of arguments:

- The recipient has himself created and signed the documents earlier with his own private key, and then sent them for time-stamping, in the hope that the CA key may be compromised later.

- When the CA key then was compromised, the recipient used the compromised key to create a false certificate in the name of the signer, but with a key pair known to the recipient.

The recipient can oppose this line of argument by showing that the signer's certificate, as indicated in the signed data, actually existed and was not revoked at the time of signing.

Note: There are other means to support roles, like the use of Attribute Certificates. In order to keep this paper reasonably short and focused, they are however not further discussed.

The need for the signing policy

The recipient may extract the identifier (name) of the signer from the certificate. In order to validate that certificate, he must also be in possession of an appropriate self-signed certificate from a CA, previously obtained in a trusted manner. That self-signed certificate will contain, in particular, the name of a CA trusted to issue certificates containing given forms of identifiers (names), a certification key and a validity period. Since there may exist multiple self-signed certificates from the same CA, used for different certificate policies and/or for different naming constraints, it is necessary to be able to know unambiguously which one that was intended to be selected by the signer and thus to be used by the recipient.

In order to identify unambiguously the non-repudiation policy to be used to validate the signature, an **identifier of the signing policy** from the signer must be part of the signed data. That signing policy, among other information, indicates to the recipient the self-signed certificates to be used.

The need for time stamping

One important property in the context of non-repudiation is the following: if a signature has been once found to be valid, the same result over the same data shall be obtained months or years later.

In order to perform the validation, the certificate used by signer at the time of the signature must be obtained and its validity at the time of the signature must be proven. It might happen that a certificate was valid at the time of the signature but revoked some time after. Should the certificate be revoked, it must be proven that the document was signed before the signing key was revoked.

The use of a time stamp performed over some data by a TSA (Time Stamping Authority) is able to provide a solution, as it will now be explained.

A time stamp is obtained by sending a hash value of some data to the TSA. The returned «time-stamp» is a signed document, containing the hash value, the identity of the TSA and the time of stamping. This allows proving that some data existed *before* the time of stamping.

If the hash of a *digital* signature is sent to a TSA and is time stamped before the revocation of the private key used to generate that signature, this will allow proving that the *digital* signature was formed before the revocation of the public key certificate.

If a recipient wants to hold a valid *electronic* signature he will have to ensure that he has obtained a valid time stamp for it, before that key (and any key involved in the validation) is revoked. The sooner after the signature time, the better.

It is important to notice that signatures may be generated "off-line" and be time-stamped later on by anyone, e.g. the signer or any recipient interested in the value of the signature. The time stamp may thus be provided by the signer together with the signed document, or obtained by the recipient after receiving the signed document.

If the recipient can not show a time stamp of the signed document, the signer may try to repudiate his signature, and thus withdraw the signed document, in the following way:

- The signer revokes his certificate as soon as he changes his mind. The certificate will then be entered into the CRL.

- The signer then claims that someone else has created the signature after the certificate was revoked, and that the recipient has not checked the CRL. Alternatively the signer can claim that the recipient has had access to the private key and created the signature himself.

In order for a signed message to be valid under a signing policy for non-repudiation purposes, and in case the certificate used for signing will be revoked, the recipient needs to obtain a **time stamp from a TSA** before the date of revocation of the signer’s certificate.

Frequently Asked Question (FAQ)

Question: Since the time stamp from the TSA is a trusted time, why is a time stamp over the signed data, excluding the signature from the signer, not able to replace the time indicated by the signer?

Answer: In such a case, it would be possible to mount the following planned attack: an attacker identifies in advance his victim and prepares a message that would apply to the victim and requests a time stamp over that prepared message. A few weeks or months later, the attacker manages to get the private key of the victim and then signs the time stamped message. The signed message from the attacker would thus appear to be signed during the validity period of the certificate and would be declared as valid!

The signing time as indicated by the signer

As described above, the time stamp serves as a proof for the recipient that a signature was applied before the date of a possible revocation or expiration of the signer's certificate. It does not indicate the signing time.

For certain applications however, the signing time as indicated by the signer may be important in order to show when the event or action was indeed recognized by the signer as being valid. In order to achieve this, with more or less accuracy, two different approaches may be used, as described below. After comparing the advantages and disadvantages of each of them, only one will be recommended.

Let us assume that the signer includes in his signature a signing time T1. As seen before, a good insurance for the recipient against the possible revocation of the signing key is to have the signature countersigned by a signing time T2 obtained from a TSA as soon as possible after the signature and then to make sure that the signing key was valid, i.e. not revoked and not expired at that time. Let us also assume that the recipient takes that insurance.

Let us now take the position of a recipient who receives a signature containing T1, countersigned by a TSA at T2. Different cases can be considered whether T1 is or is not a trusted time and whether T1 is greater or lower than T2. Let us first treat the less interesting case (T1 > T2) .

T1 > T2. If T1 is trusted (e.g. obtained from a TSA), this situation can simply not occur. If T1 is untrusted, this means that the signer has post-dated his signature.

- If such a signature is verified at a time T < T1, the signature will be declared as invalid. This is similar to a cheque effectively signed on January 30th but with the date of February 20th and which cannot be paid before February 20th. If a recipient accepts such a signature, he takes the risk of being unpaid if the signer's certificate is revoked before that date.
- If the signature is verified at a time T > T1 and if it can be proven that the signing key was still valid at T1, then the signature may now be declared as valid. Hence a new time stamp with a time greater than T1 must be obtained by the recipient from a valid TSA to form a valid signature.

T1 < T2. If T1 is untrusted, the best that can be said is that the signature was made before T2. If T1 is trusted (e.g. obtained from a TSA), it can now be said that the signature was made after T1 and before T2. However, the signer may capture T1 well in advance, so T1 is not an accurate minor margin for the signing time. In addition the recipient may also remove the time stamp which contains T2 and replace it by another one obtained later. In order to prevent the replacement of the time stamp from the TSA, another signature made by the signer over the first signature and the time stamp would be necessary. With this scheme, the signing policy would have to include a requirement for a double signature from the signer and a *real-time interaction* with the TSA. This would make off-line signatures impossible.

The alternative approach we will now describe is explained in annex D of ISO 10181-4 (Non Repudiation Framework). It has the advantage of allowing off-line signatures and mandating only one *asynchronous interaction* with a TSA. Such an interaction may be done at will by the signer or any recipient. Only two signatures are needed: one from the signer and one from the TSA. The main idea in this approach is to introduce an additional condition in the signing policy: the electronic signature will only be valid if (T2 - T1) is smaller than a *maximum* indicated in the signing policy.

It is now possible to say that the signature was made after (T2 - *maximum*) and before T2. Hence the accuracy of the signing time is equal to the maximum indicated in the signing policy. In practice T1 and T2 should be « close enough », e.g. a few minutes, hours or even days, depending upon the nature or the value of the transaction.

According to this signing policy, if a recipient holds a valid electronic signature he can not now change the time stamp at will, because he would then get an invalid electronic signature.

The signing time then serves as a reminder to the recipient to indicate when the time stamp should be obtained at the latest, according to the signing policy. It also forces the recipient to take the insurance that he will have no problem in case of later on revocation of the signer's key.

In order for the signing time, as indicated by the signer, to be valid, the **signing time** and the **TSA time** as indicated in the time-stamp from a valid TSA must be "**close enough**", according to a maximum indicated in the signing policy.

Frequently Asked Questions (FAQs)

Question: How can the signer's own indication of time be of any value in the signed data, since it is not a trusted time?

Answer: The signer's own time is neither accurate nor trusted, but its insecurity is limited by the signing policy to the time period allowed to get the time stamp. Pre-dating too far away from the current time is impossible, post-dating is however possible.

Question: Why not use in addition a trusted time for the signing time T1?

Response: We can already say that the signature was made after T1 and before T2 with T1 - T2 < maximum, according to the signing policy. There is no additional value in adding a trusted time T1 since the accuracy is not better. Requiring a trusted time would also make off-lines signatures impossible.

Summary of required evidence for non-repudiation

If the signer later repudiates (denies) a signature, the recipient will need to produce the following records as evidence in court:

1. The description of the signing policy.
2. The signed document including the appropriate data (signing policy, type of event or action, signing time, identifier of the signer's certificate and attribute certificates, if any).
3. The certificate containing the signer's public key.
4. An attribute certificate, if any is necessary according to the signing policy.
5. A time stamp from an appropriate TSA over the signed document.
6. One of the following:
    a) A CRL from the time of signing, where the certificate is not included.
    b) A response carried by OCSP (Online Certificate Status Protocol) which shows that the certificate was not revoked at the time of signing.
7. A valid chain of unrevoked CA certificates (i.e. cross-certificates) at the time of the signature up to a trust point defined in the signing policy.

Thus, all this information needs to be collected, saved and stored by the recipient when he receives and validates the signature the first time, should there be a need to prove the validity of the electronic signature in a court a few years later. However, as will be explained later, this is not sufficient when one or more of the certification keys needed to validate the certification path might have been compromised at the time of validation.

Long term validation of electronic signatures

The long term validation begins when the certificate needed to validate a signature has expired at the time of validation, and when either:

- the certification key from the CA from the signer has changed.
- the certification key from a root CA has expired at the time of validation.
- one or more of the cross-certificates have expired.

Several changes of the certification key originally used to issue some certificates may even have occurred. Some CAs may even have ceased their activity and transferred it to another CA. Some CA keys may have been compromised.

In order to perform the validation, the recipient must use a certification path valid at the time of the signature.

All certificates, cross-certificates, CRLs or OCSP responses must be time stamped, in order to protect against the later compromising of a certification key.

At the time of the first validation of the signature, the recipient will need to gather the various cross-certificates together with either current corresponding CRLs, where the cross-certificates are not included, or OCSP responses, which show that each cross-certificate is not revoked, to perform the validation.

The public key certificate from the signer and the attribute certificate, if any is used, should also be time stamped as well as each component from the certification path. Each information could be individually time-stamped. Since anyway a time stamp has to be provided by either by the signer or a recipient, that time stamp can include the whole certificate from the signer and the whole attribute certificate from the signer, if any is used.

The whole certification path may be time stamped separately so that the same data can be used for validating several electronic signatures.

This leads to the following:

The recipient needs to obtain and save, close enough to the signing time, the public key certificate from the signer and the attribute certificate (if any is used by the signer). This will make the signed message valid under a **long term** signing policy, and protect it against the compromising of the certification key of the certification authority and the attribute authority of the signer.

The recipient needs to obtain and save, close enough to the signing time, a time stamp of a valid certification chain, together with either current CRLs, where each cross-certificate is not included, or OCSP responses which shows that each cross-certificate is not revoked. This will make the signed message valid under a **long term** signing policy, and protect it against the compromising of any certification key from the certification chain.

Archived electronic signatures

Archival storage is indefinite, and may extend long past the time where the cryptography used to sign documents is still secure. Future advances in computing and cryptography are likely to make it possible to generate private keys from public keys where this is unfeasible today.

Current cryptography soon broken

In the case where it would appear to be soon possible to break a given cryptographic algorithm, but where it can be expected that new algorithms (or old ones with greater key lengths) will be available, a sequence of time stamps will protect against forgery. Each time stamp needs to be affixed over the whole evidence before either the compromising of the signing key or of the breaking of the algorithm. TSAs should have long keys (e.g. 2048 bits).

It should be noticed that the mandatory time stamping, which must be obtained shortly after the signature from either the signer or the recipient, protects not only against the compromising of the signing key, but also against the breaking of the signature algorithm used by the signer. In this way, signer's keys not do need to be long and their resistance must only be as long as the validity of the certificate. If the certificate is valid for two years, its resistance should be more than two years but does not need to be twenty years, even if the signature must be validated twenty years later.

Current hash function soon broken

In the case where it would appear to be soon possible to create hash collisions for a given hash algorithm, the whole signed document, certification path and CRLs or OCSP responses will need to be time stamped using a stronger hash algorithm before the breaking of the hash function.

In summary: if it was impossible to forge the signature at the time the signed document was time stamped by a trusted TSA, the signature is valid and can be relied upon.

However, if the recipient only **himself** re-signs his archived material, without using a trusted TSA, it can only be used for authentication, not as evidence in court for non-repudiation.