WATCH WHAT YOU SIGN!

This article was published in the
Digital Evidence and Electronic Signature Law Review Volume 3 Number 2

I was brought up in the belief that you should not sign anything you had not read. (After some years in legal practice I realised that this was a layman's misunderstanding of the original legal maxim, that you should not sign anything that your lawyer had not read.) The layman's version of the maxim is certainly sound, and for paper documents it is fairly simple to know whether you have complied with it or not. Things are not so straightforward when it comes to electronic documents and electronic signatures: this article is about some of the pitfalls which may confound the unwary. The electronic documents that I have in mind are mainly those produced by word-processing software, and the electronic signatures are the digital signatures made using public key cryptography.

Analogy between electronic and holograph signatures

It is worth noticing at the outset some of the ways in which the analogy between a handwritten signature and a digital signature is less than perfect. A handwritten signature is strongly bound to its maker. The utility of such signatures rests on a general acceptance that they are very hard to forge undetectably (given the availability of time and skill to examine the forgery). Unless a whole document is handwritten, however, the signature's resistance to forgery does not by itself provide any very strong assurance that changes to the rest of the document will be detected if made after the signature. A digital signature, by contrast, gives a very strong assurance of integrity – the slightest change to the document will prevent the signature from being verified. But the digital signature is only weakly bound to the signatory, since the binding depends on the ability of the signatory (a) to exclude others from obtaining access to the signature key and (b) to ensure that the signature creation device uses the key only to sign documents the signatory intends to sign. In a world of insecure devices and prevalent malicious software, neither result may be easy to achieve.

Any system of signing documents with digital signatures must satisfy two distinct requirements. The first is that a signature on a document must continue to be capable of verification for as long as its validity may be material to anyone with an interest in it. The second is that what the signatory appears to have signed is the same as what the signatory intended to sign.

In-line and detached signatures

Subsection 7(2) of the Electronic Communications Act 2000 (UK) provides that:

> (2) For the purposes of this section an electronic signature is so much of anything in electronic form as–

(a) is incorporated into or otherwise logically associated with any electronic communication or electronic data; and

(b) purports to be so incorporated or associated for the purpose of being used in establishing the authenticity of the communication or data, the integrity of the communication or data, or both.

This elegant definition reflects alternative forms of signature, the in-line signature (where the text to be signed and the signature itself appear as text in the same document), and the detached signature (where the file containing the document is signed and the signature takes the form of a separate file).

An example of an in-line signature is the following:

-----BEGIN PGP SIGNED MESSAGE-----

Specimen of signed text

-----BEGIN PGP SIGNATURE-----
Version: PGP 7.0

iQA/AwUBRB6O1AkSrmaJndf/EQKLpACfU8cFAg1QI2SP/vQbj1BvVmQQ1GUAn3cd
1XfDSSLvTCR07DoiLCwAV2CW
=iqNY
-----END PGP SIGNATURE-----

A detached signature will appear in the list of files:

Mutable.doc	36,352	20/03/2006	11:03
Mutable.doc.sig	66	20/03/2006	11:03

In this case a document called "Mutable.doc" has been signed, the signature being contained in the document "Mutable.doc.sig". Nothing in Mutable.doc gives any indication that it has been signed, because it is not changed by the making of the signature.

In each case the software used to create the signature can also be used to verify it.

Reliability and vulnerabilities

- *false documents that verify correctly*

There is a significant difference between the two methods of signature. The in-line signature contains and reveals the whole of the text that has been signed. In the case of the software used to generate the specimen above, PGP 7.0, it operates only on text, eliminating formatting such as colour or italics. This may seem a little crude, but it has

an important advantage. An example might be a document presented to you for signature containing hidden text, displayed in white on a white background. You might be satisfied with the visible text, and create an in-line signature; but this would convert the hidden text from white to black, revealing what you had really signed before you had released it.

As this example shows, the security properties of a procedure can only be analysed in the context of conjectural attacks on it. A more sophisticated attack might be to produce a document (the primary document) displaying text acceptable for signature, part of which is not in fact contained in the primary document in the form of text, but is displayed there as the result of the placing in the primary document of a link to a secondary document file which contains the text. Once again the comparative crudity of the in-line signature is valuable. Because the link does not consist of text, the in-line signature procedure simply discards it, and the text in the linked secondary file is not included in the signature.

(One might ask what advantage the attacker might hope to gain from including the text through a link in this indirect way: the answer is that if the attacker has control of the secondary file to which the link connects the primary document, he can alter the displayed content of the primary document by altering the text in the secondary document without making any change to the primary document.)

A detached signature operates differently. Instead of operating purely on text, it operates on the whole file, treating it as an electronic object, a collection of bits. As an example, consider the case of the primary document, part of which consists of a link to the secondary document. The text displayed in the primary document depends on the text contained in the secondary document. Changes to the text of the secondary document are reflected faithfully in the text displayed by primary document. But those changes do not change the primary document at all: it consists of its own text and a link to the secondary document. It is in part a mere conduit, which does not itself change merely because of changes in what the conduit conveys.

The result is that a detached signature on the primary document continues to verify correctly despite changes to the text it displays, if those changes reflect changes not to the primary document but to the secondary document.

This is an elementary example, and I hope not well adapted to fraud; but the sophistication of modern word-processing software, with its built-in scripting engines and other facilities, is such that signing a file with a detached signature can give surprisingly little assurance about the text to which the signature appears to relate.

- *genuine documents that fail to verify*

From "false" documents that verify correctly, consider genuine ones that fail. Most users will have had the experience of opening a document to read it, making no changes, clicking the button to close it and being met with a dialogue asking, "Do you

want to save the changes you made to Mutable.doc?" Those who unwarily click "Yes" will find that while the text content may not have changed, system changes have in fact occurred – the document may record the date on which it was last opened, for example, which may have been changed by the mere fact of your having opened it to read it. The result will be that Mutable.doc.sig can no longer be used to verify Mutable.doc, because although the text has not changed, not all the bits in the file have remained identical, and a change to any one of them is enough to prevent signature verification. In the example given, the user would no longer be able to use the signature to prove the genuineness of the signed document.

The solution in this case is to change the status of the signed file to "read only" immediately after signature; and it would be a desirable property of signature software that by default it should make such a change to protect the unwary user.

Security advantages of detached signatures

In the face of these examples of the perils of using detached signatures, what good reason is there to use them in preference to in-line signatures? There are in fact two different reasons for doing so.

The first is that in-line signatures can be applied only to text. This is satisfactory for documents and e-mail messages; but to sign a spreadsheet or a voice message or a video, only a detached signature will work. Anyone who contemplates signing files of that kind should give careful thought to just what they intend their signature to mean, and should make sure that they have a common understanding with those who will rely on the signature.

- the untrustworthiness of general purpose computers

The second reason for using detached signatures is more complex. Reference was made above to the importance of the ability of the signatory (a) to exclude others from obtaining access to his signature key and (b) to ensure that his signature creation device uses the key only to sign documents he intends to sign. Signature keys may be held in secure tokens (USB devices or smartcards) and protected by passwords to enhance the user's ability to exclude others from obtaining access to them. (Such devices may be vulnerable to a sophisticated attack by a well-equipped adversary, but the user should be alert to the loss of the token and able to revoke the validity of the key so as to frustrate such an attack.) The problem of ensuring that the user signs only what he intends to sign is more difficult, because in order to make a signature, the user depends on a general purpose computer, and general purpose computers are widely acknowledged to be untrustworthy devices. In particular, they are vulnerable to malicious software, which may give a third party the power to carry out operations on the user's computer without his knowledge. Such operations could include signing documents that are not presented to the user on the screen.

- the lack of secure signature-creation devices

This problem could be solved by the deployment of secure signature-creation devices of such limited functionality as to be practically immune to such attacks; or by the development and deployment of general purpose computers rendered trustworthy by technical means. It remains uncertain whether or when any such deployment will take place. Digital signatures have not proved to be the essential foundation for the success of electronic commerce in quite the way their promoters claimed ten years ago, and the market for secure devices may not justify the investment required (and may indeed have been distorted by the demands of the entertainment industry for the deployment of computers that satisfied the security requirements of that industry instead of those of the users).

- a helpful protocol

In the absence of secure devices to solve this problem, it may be possible in some cases to reduce the risks by procedural means. Imagine an on-line banking system in which important documents (instructions for the payment of large amounts, for example) were signed by the customer using a digital signature, perhaps after some interaction between bank and customer through "conventional" means such as a secure website and the use of passwords. The risk in contemplation is that while the customer thinks he is signing the bank's document in accordance with his intentions, a malicious third party has unknown to him caused his computer to replace the document he intended to sign with one in favour of the third party. The bank could guard against this risk by creating (and retaining) a detached signature verifying its own document before sending it to the customer. If the customer's computer signs a substituted document, the customer's signature will verify correctly; but the bank's will not, since what it gets back is not what it sent out. The third party would have to compromise not just the customer's computer but also that of the bank, and would have to co-ordinate activity between them. Such a compromise would be substantially more difficult to achieve, and the procedure would provide a corresponding measure of protection.

- in-line signatures hard to use with the protocaol

But in practice the procedure depends on the use of detached signatures, and could not comfortably be achieved with in-line signatures. If the bank creates a detached signature in order to verify that the returned document has not been changed, then the customer must use a detached signature to sign it, because if the customer uses an in-line signature, the document will necessarily change by the inclusion of the signature, and the bank's check will fail. It is true that the bank could send the document to the customer with an in-line signature. The sophisticated customer may not object to the appearance of the document (although banks may not be confident of their customers' sophistication in the matter); but in order to preserve the bank's ability to verify that the instruction was the same as in the document it sent out, the customer would have to sign the bank's signature as well as the text it signed. This is not impossible – the text within a signature can be any text, including another signature – but the effect is strange

and the procedure might easily be prone to errors.

To illustrate this, assume a payment instruction as follows:

[Payment Instruction]

The bank sends this out signed:

-----BEGIN PGP SIGNED MESSAGE-----

[Payment Instruction]

-----BEGIN PGP SIGNATURE-----
Version: PGP 7.0

iQA/AwUBRB6vrgkSrmaJndf/EQKDmwCeOMz4oyyJJVZXa05qO72f0KSb/PIAnA69
QsCgiwyUdpM+oHqlGsBAefRG
=m8dV
-----END PGP SIGNATURE-----

The customer sends this back signed in turn, with both the payment instruction and the signature included within his own signature:

-----BEGIN PGP SIGNED MESSAGE-----

-----BEGIN PGP SIGNED MESSAGE-----

[Payment Instruction]

-----BEGIN PGP SIGNATURE-----
Version: PGP 7.0

iQA/AwUBRB6vrgkSrmaJndf/EQKDmwCeOMz4oyyJJVZXa05qO72f0KSb/PIAnA69
QsCgiwyUdpM+oHqlGsBAefRG
=m8dV
-----END PGP SIGNATURE-----

-----BEGIN PGP SIGNATURE-----
Version: PGP 7.0

iQA/AwUBRB6v7QkSrmaJndf/EQIEjACg+CynG4PWRldOt8U7zsvsyWteWg0AoLZf
phtHcObcDaO7KCX/U8QSYM0v
=lfUv
-----END PGP SIGNATURE-----

This a confusing procedure, to put it no higher, and its commercial attractiveness must be doubted.

Conclusion

Regrettably, there is no satisfactory solution that can be offered at present. Detached signatures lend themselves to advantageous use in a secure context, such as between bank and customer where there is mutual trust. (Although even there a dishonest employee of the bank might try to use the linked file technique to manipulate the

customer's instructions and thereby defraud the bank.) But outside a trusted environment, the use of detached signatures introduces serious security vulnerabilities, and some risks to reliability through mishandling of documents. Methods of in-line signature which apply only to text, and offer protection against invisible text, linked documents or scripting techniques, provide users with significant benefits.

Handwritten signatures have worked well for several centuries. Perhaps it was always too much to hope that satisfactory electronic replacements would emerge in just a few years.