



Ten Things to Look for in Digital Signing Software

1. Handling dynamic content
2. Signing all content
3. Content taken direct from the application
4. No ActiveX controls or macros
5. Validation of certificates on signing
6. Countersigning
7. Spoof resistance
8. Support range of file formats
9. Support for visible signatures with images
10. Time-stamping

1. Is Dynamic content handled correctly?

Dynamic content such as fields and VBA scripts in Office documents can be a source of problems for digital signatures. For example a date/time field may be set to update whenever a document is opened. This change in value could invalidate digital signatures for the document.

It has also been pointed out that fields in Office documents can be used to maliciously change the content of a document after it has been signed, without invalidating the signature¹. This would happen if the signing software did not include the content of fields when calculating the hash of the content.

secure2sign locks fields before signing, preventing both invalidation of the signature and changes to content, and so ensuring content integrity. Users are also notified of any dynamic content before a document is signed.

2. What aspects of the document are included in the signature?

Digital signature software must interpret the document that the user sees on-screen as a stream of binary information. Unfortunately, it is difficult to realise all of this information unambiguously as binary data, which may lead to some information being omitted when calculating the hash to create the digital signature.

¹ K. Kain, S.W. Smith, R. Ashokan, *Digital Signature and Electronic Documents: A Cautionary Tale*, 6th IFIP Conference on Multimedia and Security, (2002).
(<http://www.dartmouth.edu/~pkilab/slides/IFIPSecurity26Sep02.pdf>).



For example, one approach would be to *only* include the text, omitting fonts and other formatting information. Other aspects of the document that are commonly omitted include embedded or linked objects (charts, spreadsheets, etc.).

However, omitting this information is potentially dangerous, as changes to font size, colour, etc., may be used to change the content visibly (and hence its meaning) with the signature apparently showing that the content is unchanged from when it was signed. This type of attack has been demonstrated². A similar attack on digital signatures that do not include data from linked or embedded content has also been demonstrated¹.

Note that signing the entire document file is not a viable solution as, for example, Microsoft Office files contain substantial control information that is subject to change independent to the content.

secure2sign includes all salient content data, including font settings and linked and embedded objects, when calculating the hash for a signature. This eliminates the attacks, noted above, on signature validity.

3. Does the content being signed come directly from the application?

It has been recommended³ that the data to be signed is obtained directly from the application that is used to generate and view the document, both when signing and verifying signatures.

Most signing applications work in this way but some use an indirect approach. For example, one method is based on printing the content to an image file and signing the image. To verify the signature it is the *image* that is viewed and checked, not the original document. This method resolves many of the issues noted above (dynamic content and content interpretation) but at the cost of usability and flexibility. Also, sending such images by email will often require greater bandwidth, and their reception may be blocked by some organisations. Furthermore, serious attacks on this method have been demonstrated⁴.

secure2sign is closely integrated with the applications that it supports for digital signing and obtains the content data for both signing and verifying signatures directly from the relevant application.

4. Does the software require an ActiveX control or macros?

Most digital signature programs that integrate with Microsoft Office applications require macros and / or ActiveX controls to be inserted into signed documents. There are a number of problems with this approach:

² Jøsang, A., Povey, D. and Ho, A. *What you see is not always what you sign*, The Proceedings of the Australian UNIX User Group, Melbourne (2002) (<http://sky.fit.qut.edu.au/~josang/papers/JPH2002-AUUG.pdf>).

³ Adil Alsaid and Chris J. Mitchell, *Dynamic content attacks on digital signatures*, Information Management & Computer Security **13** (4), 328 (2005).

⁴ Spalka, A., Cremers, A.B., und H. Langweg, *The Fairy Tale of "What You See Is What You Sign" –Trojan Horse Attacks on Software for Digital Signatures*, Proceedings of IFIP WG 9.6/11.7 Working Conference on Security and Control of IT in society-II. Ed. Fischer-Hübner, S., Olejar, D., Rannenber, K., Bratislava (2001) (<http://www2.hig.no/~hannol/research/scits01p.pdf>)



- Security settings in Microsoft Office may interfere with the operation of such applications.
- Many organisations do not allow the use of documents with embedded ActiveX controls or macros due to security considerations.
- Some Office file formats do not support the use of embedded macros or ActiveX controls (e.g. .rtf, .docm in Office 2007, etc).
- Incorrectly written or insufficiently tested ActiveX controls pose a security risk, according to Microsoft⁵.

secure2sign is integrated into Microsoft Office applications without ActiveX controls, Visual Basic code or macros, providing a robust and secure signing environment.

5. Is the signer's certificate validated at signing time?

When someone signs a paper document one of the requirements is that the signer was competent to sign *at the time of signing*. So, any future change to the signer's competence will have no effect on the validity of their signature on existing documents. For example, if a person signs a document and then, six months later, dies, their signature on the document is still valid. The fact that this person can no longer sign documents does not affect this.

Similarly, with digital signatures it is important for the signing software to validate the user's digital certificate at the time of signing. It is irrelevant as to whether the certificate is subsequently revoked or expired – it only matters that the certificate is valid when used to sign – like the paper signature.

Most digital signing software does not check for certificate validity at signing time; the only check on certificate validity occurs when the signature is subsequently checked. This has two unfortunate consequences:

- A signature can be made with an invalid certificate without the user knowing until the signature is checked.
- A signature made with a valid certificate that subsequently expires will then be reported as an invalid signature which it clearly isn't.

secure2sign validates the user's certificate at signing time, to prevent the use of an invalid certificate.

6. Is countersigning supported?

Countersigning is a useful option when digitally signing documents. In effect a countersignature is used to say that both the document content and all previous signers are valid at the time of the countersigning.

Countersigning works by signing the document and also validating the certificates of all previous signers of the document. (These certificates could have expired or been revoked since they were used to sign the document.) It can be used as a form of notarisation for document signatures or as a means of stating that the existing signatories are approved.

secure2sign supports countersigning.

7. Is the software resistant to spoofing?

One method of attacking digital signatures is to intercept the window that displays signature verification details and substitute either the entire window or

⁵ <http://office.microsoft.com/en-gb/assistance/ha011403101033.aspx>



parts of it so that it then appears to display the results the attacker wants. For example, that all signatures are valid when in fact the document content has been altered.

With active documents (e.g. Word, Excel or PowerPoint native files) this can even be achieved without requiring external Trojan horse programs, by utilising Visual Basic commands built into the application and stored within the document.

This type of attack has been demonstrated against commercial signing software⁴.

secure2sign incorporates techniques to resist this type of attack, to help ensure that correct signing information is always delivered to the user.

8. Does the software support a range of file formats?

Many digital signing programs only permit signing of native file formats within Microsoft Office. For example with Microsoft Word prior to Office version 2007, this would mean only supporting signing of .doc files.

Although most Office documents are generated and used in native format, sometimes documents in other formats are used, for example .rtf and it would be useful to apply signing to these documents as well.

secure2sign supports signing of a range of file formats within Microsoft Office applications.

9. Are visible signatures with images supported?

Although visible signatures confer no security benefits, users often like to see something visible when a document is signed. Furthermore such visible signatures can incorporate images of the user's handwritten signature or a company logo to further enhance the signature.

secure2sign supports visible signature with user supplied images.

10. Support for Timestamping

Timestamping a document involves sending the document hash to a certified timestamp server. The current time, digitally signed by the time stamp provider, is returned. This provides proof that the document existed at the time of the timestamp, with the signature enabling it to be shown that the document has not been modified since the timestamp was made.

secure2sign supports timestamping based on RFC 3161.

 info@avocosecure.com

US: +1 415 839 9433

International: +44 207 851 6070

www.avocosecure.com

