

Entrust Certificate Services

MS Office VBA Signing

User Guide

Document issue: Draft

Date of Issue: February 2009



Copyright © 2009 Entrust. All rights reserved.

Entrust is a trademark or a registered trademark of Entrust, Inc. in certain countries. All Entrust product names and logos are trademarks or registered trademarks of Entrust, Inc. in certain countries. All other company and product names and logos are trademarks or registered trademarks of their respective owners in certain countries.

This information is subject to change as Entrust reserves the right to, without notice, make changes to its products as progress in engineering or manufacturing methods or circumstances may warrant.

Export and/or import of cryptographic products may be restricted by various regulations in various countries. Export and/or import permits may be required.

Obtaining technical support

For support assistance by telephone call one of the numbers below:

- 1-877-754-7878 in North America
- 1-613-270-3700 outside North America

You can also email Customer Support at:

- support@entrust.com

This guide contains information about signing Microsoft® Office files. Sections in this guide include:

- [“The code signing process for MS Office/VBA”](#)
- [“Verifying the authenticity of the software”](#)
- [“Obtaining and using an Entrust Office/VBA signing certificate”](#)

Entrust certificates for Microsoft® Office and Visual Basic Applications (VBA) can be used to sign DOC, DOT, XLS, XLT, XLA, PPT, PPS, and PPA files. The signature is used by the browser to provide some confidence to the end user that the code is from a legitimate source and has not been subjected to tampering. Entrust offers PKCS#12 (Public Key Cryptography Standard # 12) certificates for use with MS Office and VBA files.

- For more information about code signing, certificates and keys, see the *Entrust Certificate Services Code Signing General Information Guide* available from *****
- For information about using an Entrust certificate with Microsoft Authenticode see the *Entrust Certificate Services Microsoft Authenticode Code Signing Guide* available from *****
- For information about using an Entrust certificate with Microsoft Authenticode see the *Entrust Certificate Services Java Code Signing Guide* available from *****

The code signing process for MS Office/VBA

When the code is signed, several pieces of information are added to the file. This information is used when the code is run by the end-user's browser to authenticate the author of the code and to check for tampering.

The bundle that is used to verify the authenticity of the code is created during two sequences of events:

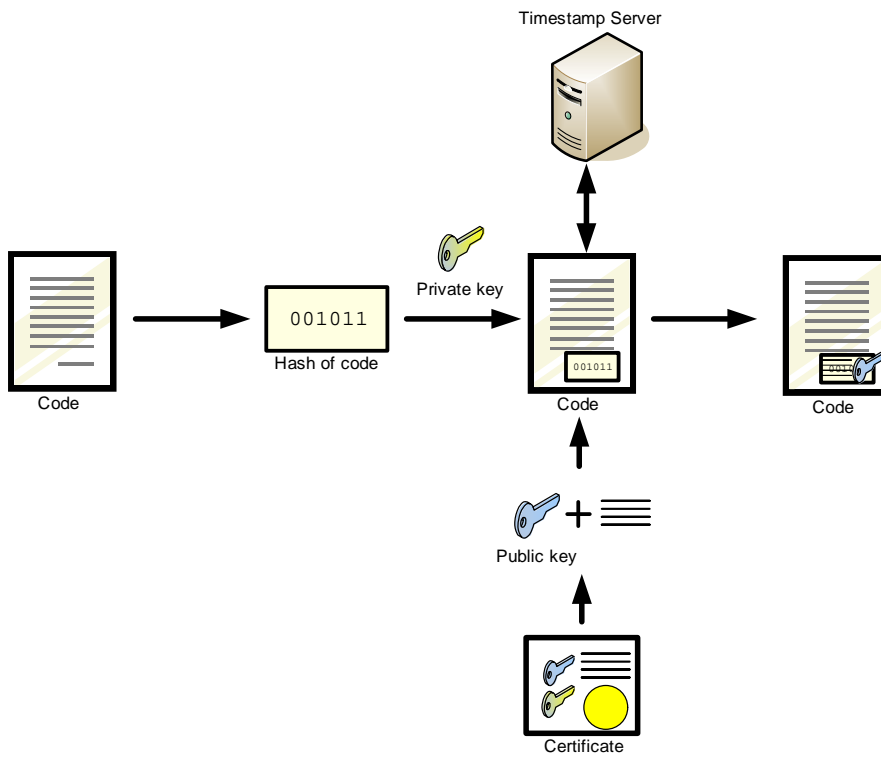
- First a mathematical representation of the code, called a hash, is created and signed. At that point the hash and signature are timestamped, hashed (with the timestamp) and signed again.
- The timestamp and second signature are applied by a timestamping authority (TSA). Timestamping Authorities are usually maintained by a third party (such as Entrust) to insure the validity of the timestamp.

The entire sequence takes place as follows:

- The code is passed through a hashing algorithm creating a hash of the file. The hash is an exact numerical representation of the file. The hash is only reproducible using the unaltered file and the hashing algorithm that was used to create the hash. The hash is bundled with the file.
- The hash is signed using the signer's private key.
 - Information identifying the creator of the signature is drawn from the signer's certificate and incorporated into the signature.
 - Information about the CA or CAs that signed the signer's certificate is drawn from the signer's certificate and incorporated into the signature.
- The signer's public key is added to the bundle.
- The signature is sent to the timestamping authority (TSA).
 - The TSA adds a timestamp to the to the bundled information and computes a new hash.
 - The TSA signs the new hash with its private key creating a new bundle of information.
 - The timestamped bundle, original bundle that was sent to the TSA and the time stamp are re-bundled with the original code.

Figure 1: The code signing process for MS Office/VBA

MS Office/VBA code signing process



Verifying the authenticity of the software

When the end user's browser loads the code, it checks the authenticity of the software using the signer's public key, signature and the hash of the file. The timestamp is checked using a similar process.

If both the timestamp and the signature are verified successfully, the browser accepts the code as valid. If either the timestamp or signature are not successfully verified, the browser will react by warning the user or rejecting the code, according to the level of security being used.

Verifying the timestamp

The following sequence of events is used to verify the timestamp.

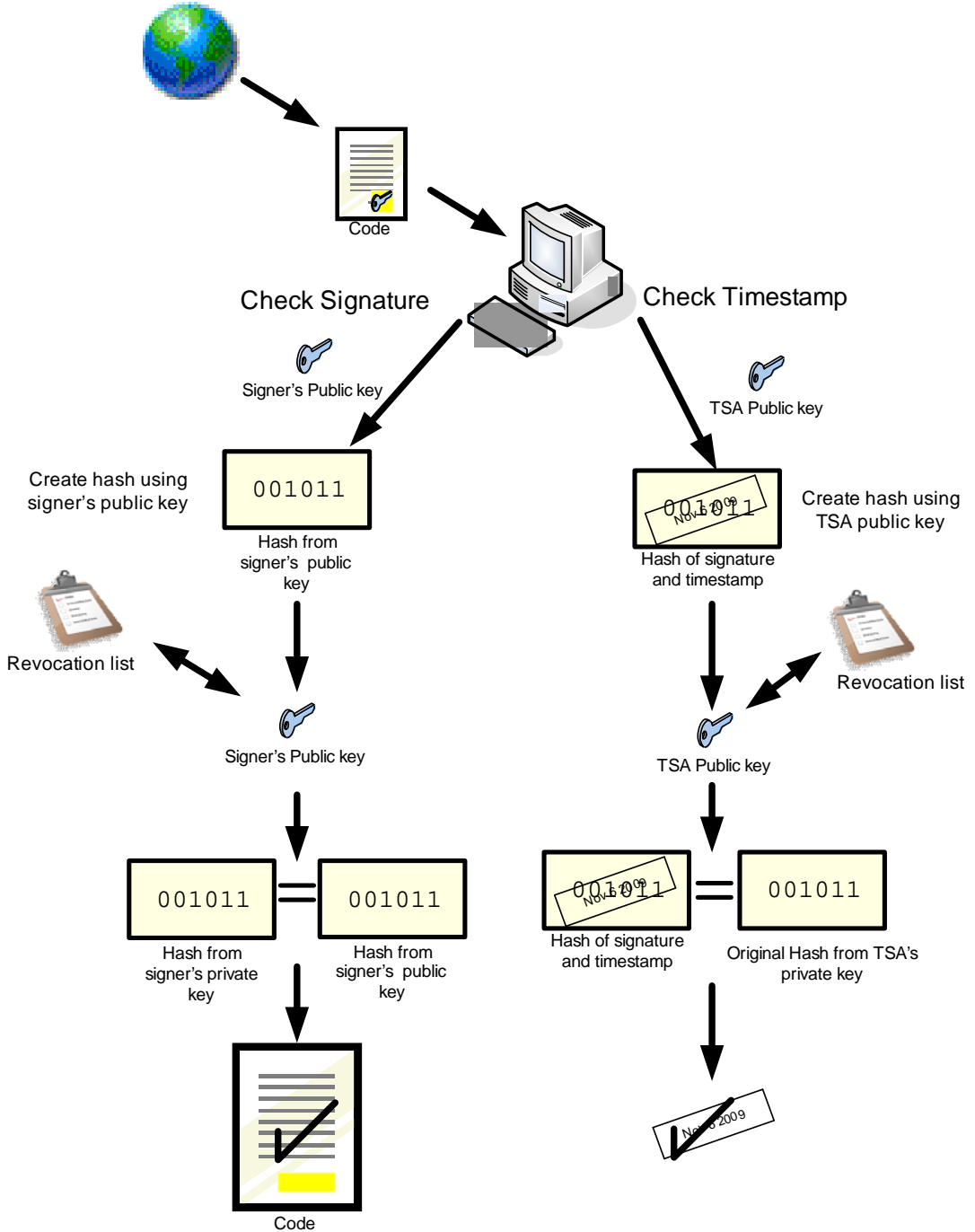
- The timestamp is added to the bundled signature information and the combined signature and timestamp are hashed.
- The Timestamping Authority's public key is applied to the timestamped signature block revealing the hash calculated by the TSA.
- The TSA's public key is verified by checking its expiry date and consulting the revocation lists to be sure that it has not been revoked.
- The two hashes are compared. If equal, the timestamp is considered to be valid.

Verifying the signature

The signature is verified as follows:

- The original code is passed through a hashing algorithm creating a hash.
- The public key of the designer or publisher is applied to the signature information revealing the hash that was calculated when it was signed.
- The expiry date of the public key is checked and it is checked against the revocation lists to be sure that it is valid.
- The two hashes are compared. If the hashes are equal, the signature is considered to be valid.
- If the file is considered to be valid it is accepted by the browser. If the file is not considered to be valid the browser takes the security measure appropriate to its current level of security.

Figure 2: Verifying the authenticity of the code



Obtaining and using an Entrust Office/VBA signing certificate

Microsoft Office/VBA code is signed using the Microsoft office application that produced it. For example, if you create a spreadsheet in Microsoft Excel, the Excel application offers built-in tools that enable you to sign it using your Entrust certificate and private key.

Obtaining a certificate from Entrust

To obtain a code signing certificate from the Entrust, log into the Entrust Web site URL <https://buy.entrust.net/buy>. Code signing certificates are available to users who have registered for the Entrust Certificate Management System (CMS). For information about enrolling in the CMS see the *Entrust Certificate Management System Enrollment Guide*. For information about buying and managing code signing certificates see the *Entrust Certificate Management System User Guide*.

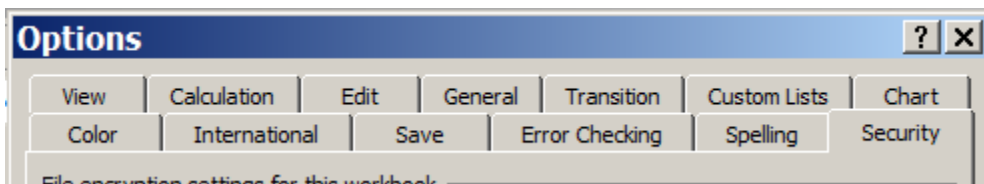
Using your Entrust certificate to sign a Microsoft Excel spreadsheet

Several Microsoft applications include built-in tools that allow the user to sign documents that they create. The signing process used by Microsoft Excel is used as an example in this procedure.

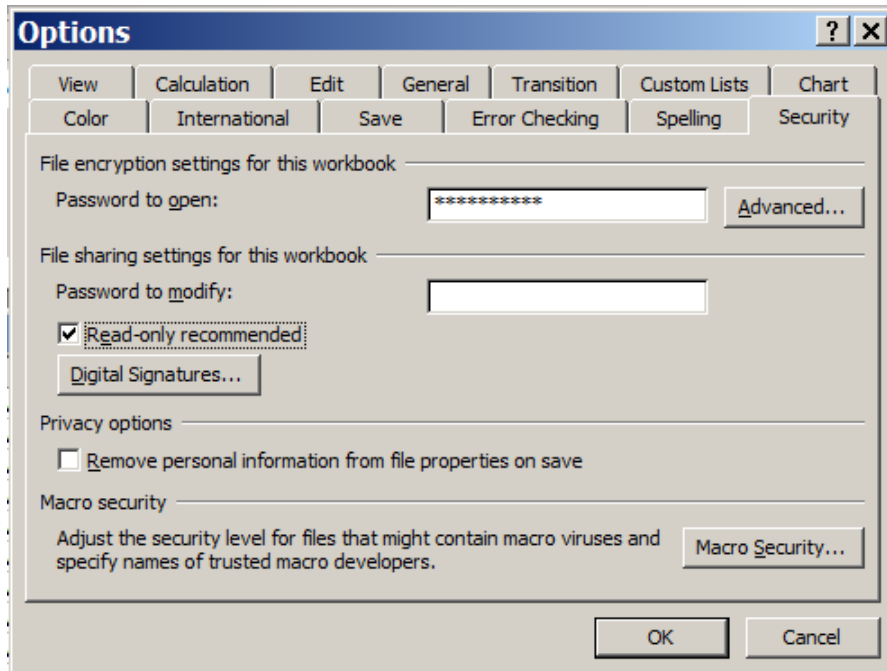
This procedure assumes that you have downloaded and installed an Entrust PKCS#12 certificate.

To sign a Microsoft Excel spreadsheet

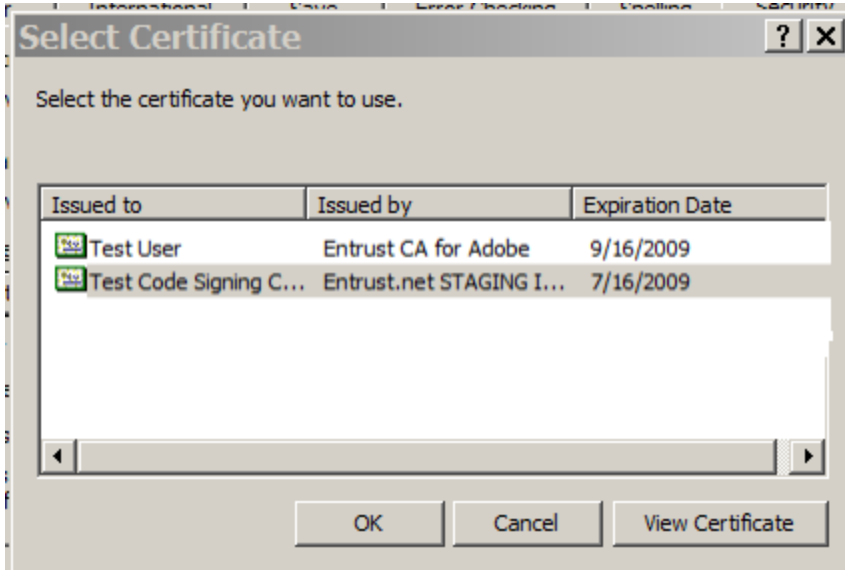
- 1 Save the spreadsheet.
- 2 From the Excel toolbar select **Tools > Options**.
- 3 In the **Options** Window, select the **Security** tab.



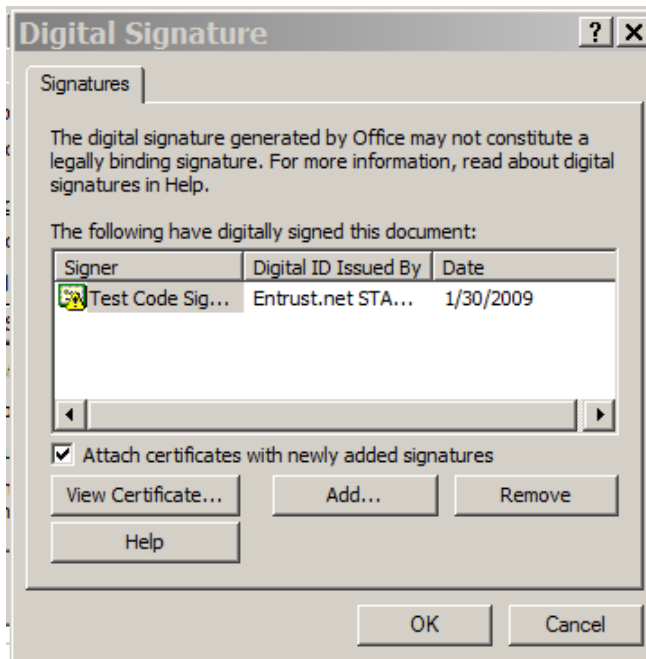
- 4 Select the security features that you want to implement for the file and click **Digital Signatures**.



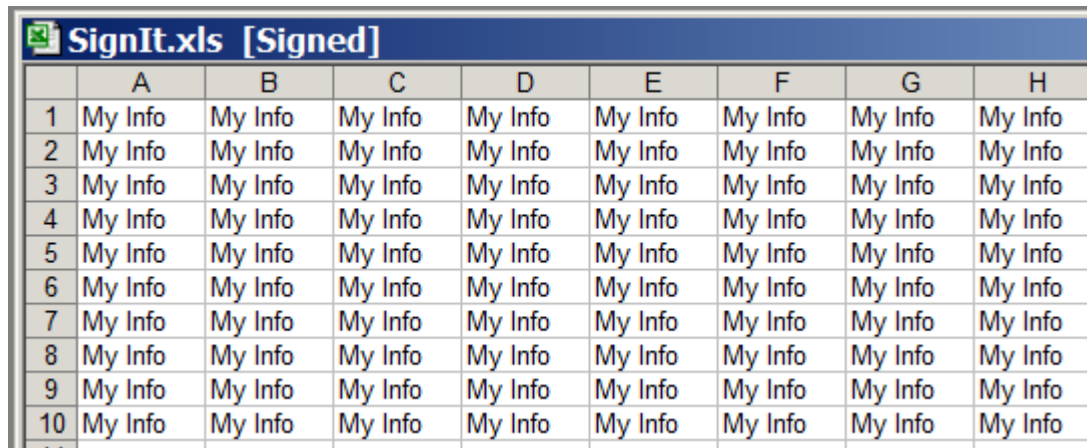
- 5 In the **Select Certificate** page, select your Entrust Microsoft Office/VBA signing certificate and click **Ok**.



- 6 In the Digital Signature window, click **Ok** to accept the signature.



The heading of the Excel spreadsheet indicates that it has been signed.



The image shows a screenshot of an Excel spreadsheet window. The title bar at the top reads "SignIt.xls [Signed]". The spreadsheet contains a grid of data with 10 rows and 8 columns (A through H). Each cell in the grid contains the text "My Info".

	A	B	C	D	E	F	G	H
1	My Info	My Info	My Info	My Info	My Info	My Info	My Info	My Info
2	My Info	My Info	My Info	My Info	My Info	My Info	My Info	My Info
3	My Info	My Info	My Info	My Info	My Info	My Info	My Info	My Info
4	My Info	My Info	My Info	My Info	My Info	My Info	My Info	My Info
5	My Info	My Info	My Info	My Info	My Info	My Info	My Info	My Info
6	My Info	My Info	My Info	My Info	My Info	My Info	My Info	My Info
7	My Info	My Info	My Info	My Info	My Info	My Info	My Info	My Info
8	My Info	My Info	My Info	My Info	My Info	My Info	My Info	My Info
9	My Info	My Info	My Info	My Info	My Info	My Info	My Info	My Info
10	My Info	My Info	My Info	My Info	My Info	My Info	My Info	My Info

