



BUSINESS GUIDE



VeriSign® Code Signing for Microsoft®
Authenticode® Technology
Realizing the Possibilities of Internet
Software Distribution



Where it all comes together.™



CONTENTS

+ What is Authenticode?	3
+ Who Needs Code Signing Digital Certificates?	4
+ What Does Authenticode Look Like to Consumers?	4
+ Technical Overview: (Optional Reading)	6
What is a Digital Certificate?	6
CAs	7
How Does Object Signing Work with VeriSign Code Signing Digital Certificate?	7
Timestamping	8
The six Steps to Signing Code	9
+ Conclusion	10



What Is Code Signing?

When customers buy software in a store, the source of that software is obvious. Customers can tell who published the software, and they can see whether the package has been opened. These factors enable customers to make decisions about which software to purchase and how much to trust that software.

When customers download software from the Internet, the most they see is a message warning them about the dangers of using the software. The Internet lacks the subtle information provided by packaging, shelf space, shrink wrap, and the like. Without an assurance of the software's integrity, and without knowing who published the software, it is difficult for customers to know how much to trust software. It is difficult to make the choice to download the software from the Internet.

The solution to these issues is Microsoft Authenticode coupled with VeriSign® Digital Certificate. VeriSign is Microsoft's preferred provider of digital certificate services. Authenticode, through the use of digital signatures, enables software developers to include information about themselves and their code with their programs.

When customers download software signed with Authenticode and verified by VeriSign, they can be assured of

- **Content Source**—The software really comes from the publisher who signed it.
- **Content Integrity**—The software has not been altered or corrupted since it was signed.

Users benefit from this software accountability because they know who published the software, and they know the code has not been tampered with. In the extreme case that software performs unacceptable or malicious activity on their computers, users can also pursue recourse against the publisher. This accountability and potential recourse serve as a strong deterrent to the distribution of harmful code.

Developers and Web masters benefit from Authenticode because it puts trust in their name and makes their products harder to falsify. By signing code, developers build a trusted relationship with users, who then learn to confidently download signed software from that publisher or Web site. With Microsoft Authenticode, developers can create exciting Web pages using signed ActiveX® controls, signed Java™ applets, or other signed executables. And users can make educated decisions about which software to download.

Who Needs Code Signing Digital IDs?

Any publisher who plans to distribute code or content over the Internet or over corporate extranets risks impersonation and tampering. VeriSign Code Signing Digital Certificate for Authenticode Technology protects against these hazards. Authenticode is currently used to sign 32-bit .exe files (Windows® Preinstallation Environment [PE] files), .cab files, .ocx files, and .class files. In particular, if you are distributing active content (such as ActiveX controls) for use with such Microsoft end-user applications as Internet Explorer, Exchange, Outlook,® or Outlook Express, you will want to sign your code using Authenticode.

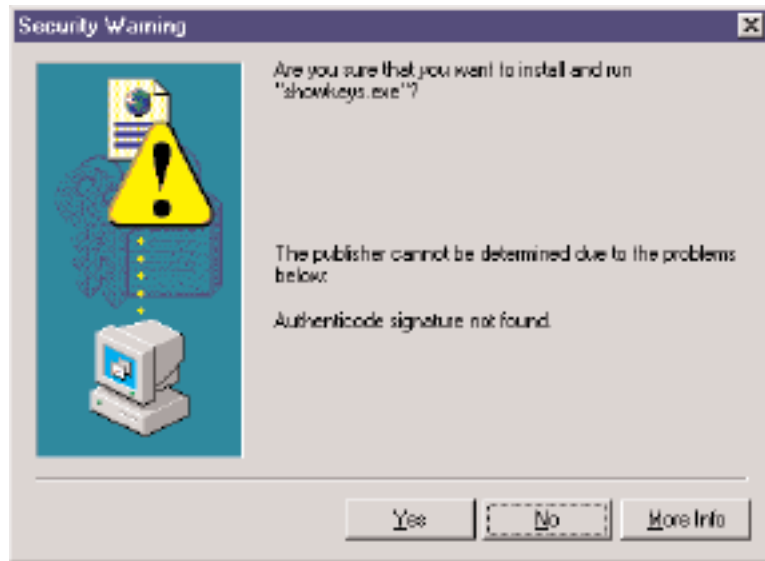
VeriSign offers a Class 3 certificate designed for commercial software publishers. These are companies and other organizations that publish software. This class of certificate provides greater assurance about the identity of a publishing organization and is designed to represent the level of assurance provided today by retail channels for software.

What Does Authenticode Look Like to Consumers?

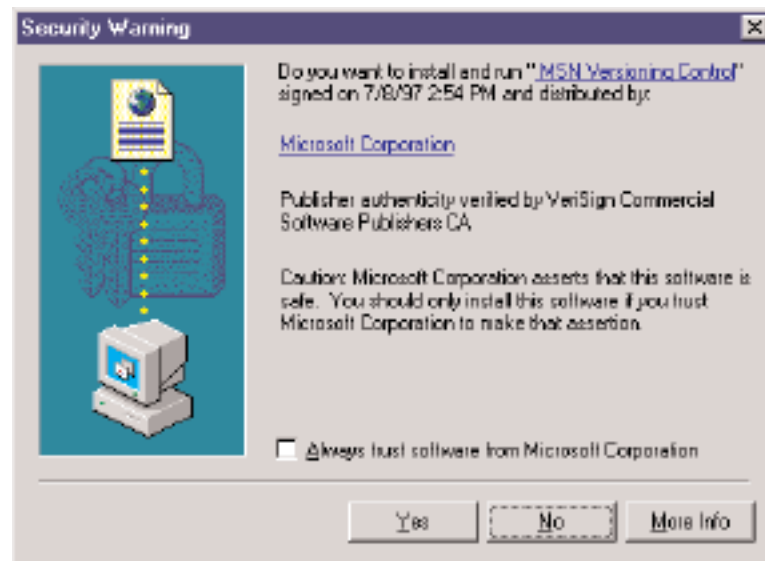
Microsoft client applications, such as Internet Explorer, Exchange, Outlook, and Outlook Express, come with security features that incorporate Authenticode. These applications are often used to obtain other pieces of software. In a component model such as ActiveX or Java this happens frequently, often without the end user being aware of it. For example, when a user visits a Web page that uses executable files to provide animation or sound, code is often downloaded to the end user's machine to achieve the effects. While this may provide substantial value, users risk downloading viruses or code from a disreputable publisher.

If an end user of one of these applications encounters an unsigned component distributed via the Internet, the following will occur:

- If the application's security settings are set on "High," the client application will not permit the unsigned code to load.
- If the application's security settings are set on "Medium," the client application will display a warning like this screen:



By contrast, if a user encounters a signed applet or other code, the client application will display a screen like the following:

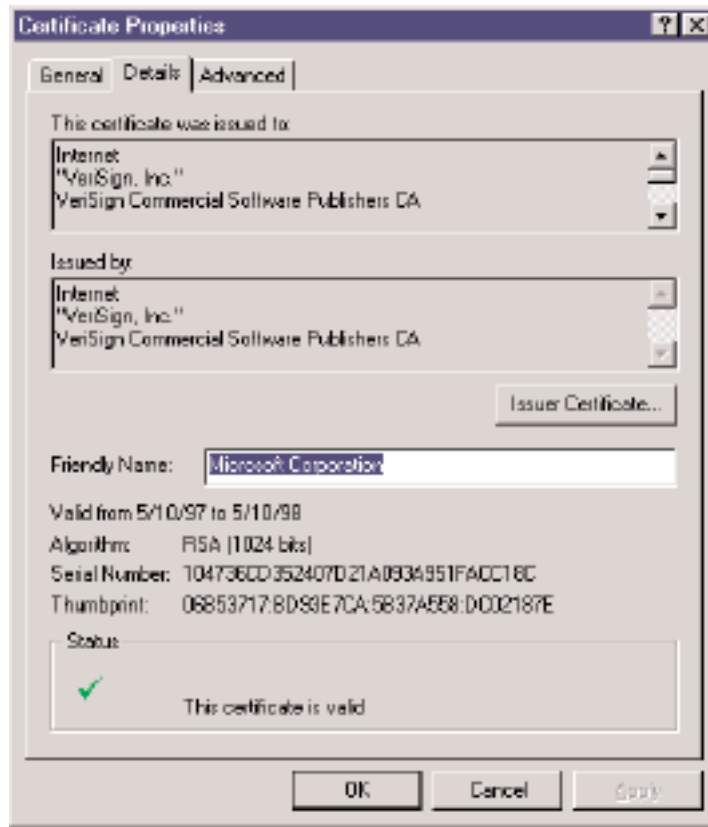


Through Authenticode, the user is informed

- Of the true identity of the publisher (in this case Microsoft Corporation)
- Of a place to find out more about the control (in this case MSN Versioning Control)
- That the authenticity of the above information is provided by VeriSign

Users can choose to trust all subsequent downloads of software from the same publisher. They can also choose to trust all software published by commercial publishers (see above) that have been certified by VeriSign.

Simply by clicking the “More Info” button, users can inspect the certificate and verify its validity:



Technical Overview

+ What Is a Digital Certificate?

A digital certificate is a form of electronic credential for the Internet. Similar to a driver’s license, employee ID card, or business license, a digital certificate is issued by a trusted third party to establish the identity of the certificate holder. The third party who issues certificates is known as a Certificate Authority (CA).

Digital-certification technology is based on the theory of public key cryptography. In public key cryptography systems, every entity has two complementary keys—a public key and private key—which function only when they are held together. Public keys are widely distributed to users, while private keys are kept safe and only used by their owner. Any code digitally signed with the publisher’s private key, can only be successfully verified using the complementary public key. Another way to look at this is that code successfully verified using the publisher’s public key (which is sent along with the digital signature), can only have been digitally signed using the publisher’s private key (thus authenticating the source

of the code), and has not been tampered with. For more information on public keys and private keys, please see the VeriSign White Paper, *Introduction to Public Key Cryptography*.

The purpose of a digital certificate is to reliably link a public/private key pair with its owner. When a CA, such as VeriSign, issues a certificate, it verifies that the owner is not claiming a false identity. Just as when a government issues you a passport, it is officially vouching for the fact that you are who you say you are, when a CA issues you a digital certificate, it is putting its name behind the statement that you are the rightful owner of your public/private key pair.

+ CAs

CAs, such as VeriSign, are organizations that issue digital certificates to applicants whose identity they are willing to vouch for. Each certificate is linked to the certificate of the CA that signed it.

As the Internet's leading CA, VeriSign has the following responsibilities:

- Publishing the criteria for granting, revoking, and managing certificates
- Granting certificates to applicants who meet the published criteria
- Managing certificates (for example, enrolling, renewing, and revoking them)
- Storing VeriSign root keys in an exceptionally secure manner
- Verifying evidence submitted by applicants
- Providing tools for enrollment
- Accepting the liability associated with these responsibilities
- Timestamping digital signatures

+ How Does Authenticode Work with VeriSign Digital Certificates?

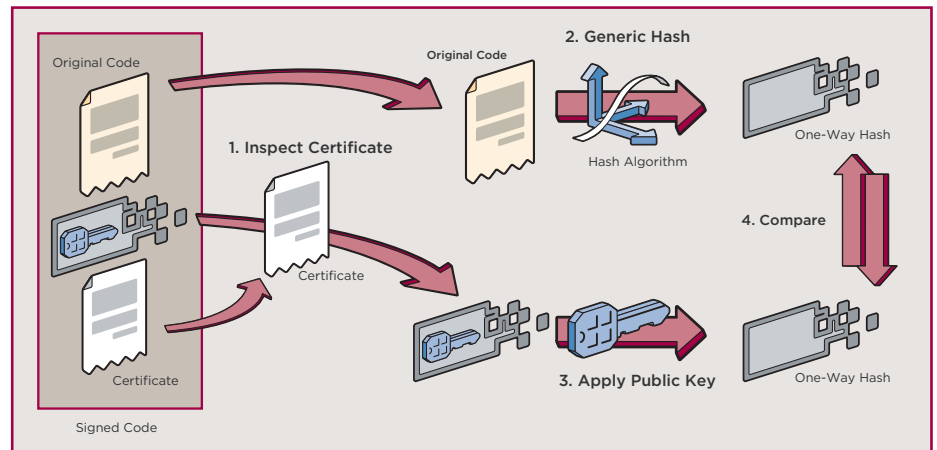
Authenticode relies on industry-standard cryptography techniques such as X.509 v3 certificates and Public Key Cryptography Standard (PKCS) #7 and #10. These are well-proven cryptography protocols, which ensure a robust implementation of code signing technology. Developers can use the WinVerifyTrust application programming interface (API), upon which Authenticode is based, to verify signed code in their own Win32 applications.

Authenticode uses digital signature technology to assure users of the origin and integrity of software. In digital signatures, the private key generates the signature, and the corresponding public key validates it. To save time, the Authenticode protocols use a cryptographic digest, which is a one-way hash of the document.

The process is outlined below:

1. The publisher obtains a VeriSign Code Signing Digital Certificate.
2. The publisher creates code.
3. Using the SIGNCODE.EXE utility, the publisher:
 - Creates a hash of the code, using an algorithm such as message digest 5 (MD5) or secure hash algorithm (SHA),
 - Encrypts the hash using his/her private key,
 - Creates a package containing the code, the encrypted hash, and the publisher's certificate.
4. The end user encounters the package.

5. The end user's Microsoft browser examines the publisher's certificate. Using the VeriSign root public key, which is already embedded in Authenticode enabled applications, the end user browser verifies the authenticity of the Code Signing Digital Certificate (which is itself signed by the VeriSign root private key).
6. Using the publisher's public key contained within the publisher's certificate, the end user browser decrypts the signed hash.
7. The end-user browser runs the code through the same hashing algorithm as the publisher, creating a new hash.
8. The end-user browser compares the two hashes. If they are identical, the browser conveys the message that VeriSign has verified the content. The end user then has confidence that the code was signed by the publisher identified in the certificate and that the code has not been altered since it was signed.



The entire process is seamless and transparent to end users, who see only a message that the content was signed by its publisher and verified by VeriSign.

+ Timestamping

Because key pairs are based on mathematical relationships, which can theoretically be “cracked” with a great deal of time and effort, it is a well-established security principle that digital certificates should expire. Your VeriSign Digital Certificate will expire one year after it is issued. However, most software is intended to have a lifetime longer than one year. To avoid having to resign software every time your certificate expires, VeriSign and Microsoft introduced a timestamping service. Now, when you sign code, a hash of your code will be sent to VeriSign to be timestamped. As a result, when your code is downloaded, clients will be able to distinguish between:

- Code signed with a revoked certificate, which should NOT be trusted
- Code signed with a certificate that was valid at the time the code was signed but has subsequently expired, which SHOULD be trusted

This means that you will not need to worry about resigning code when your VeriSign Digital Certificate expires. VeriSign is the only CA offering the timestamping service. This service is free to all VeriSign commercial and individual Code Signing Digital Certificate customers.

+ The Six Steps to Signing Code

Signing Code is an easy six-step process. By following the instructions below, you will be signing code in no time.

Step 1: Make sure you are running the correct versions of all tools.

The tools include:

- Internet Explorer 5.0 or later
- Internet Client Software Developer's Kit (SDK)

These tools are all available from Microsoft.

Step 2: Apply for a Code Signing Digital Certificate for Authenticode from VeriSign.

[Click here](#) for instructions on obtaining a code signing Digital Certificate.

In the process of applying for a VeriSign Digital Certificate, your browser will generate a private key. You should store this private key (called MyPrivateKey.pvk) on a floppy disk that you store in a safety deposit box or other secure location. Please make a backup copy of this private key, as you will need this key to sign code. This key is never sent to VeriSign, so if you lose this private key, you will be unable to sign code. If this key is lost or stolen, please contact VeriSign immediately.

Step 3: Pick up your Certificate.

Once you have completed the application process, VeriSign will take a number of steps to verify your identity. For commercial publishers, VeriSign does a considerable amount of background checking. As a result, it will take approximately 3-5 days to verify your information and issue a certificate.

At the end of this process, VeriSign will send you an email containing a personal identification number (PIN). Follow the instructions in this email to pick up your certificate. Save your certificate as a file (e.g. MyCredentials.spc).

Please note that you must use the same machine to apply for and obtain your certificate. You can then use the private key and certificate to sign files on a different machine.

Step 4: Prepare your files to be signed.

If you are building any PE file (*.exe*, *.ocx*, *.dll*, or other), you need not do anything special. For *.cab* files, you need to add the following entry to your *.ddf* file before creating the *.cab* file: Set ReservePerCabinetSize=6144

Step 5. Sign your files.

You can now sign your *.exe*, *.cab*, *.ocx*, or *.dll* file. To sign, you use the SIGNCODE.EXE utility included in the ActiveX SDK. You will also need your certificate file (generally called MyCredentials.spc) and the diskette containing your private key (MyPrivateKey.pvk).

SIGNCODE.EXE should be used from the MS-DOS prompt. Here is an example of how to sign:

```
C:\>ActiveX\INETSdk\ signcode -prog myfilename -name displayname  
-info http://www.mycompany.com -spc mycredentials.spc -pvk  
a:myprivatekey.pvk -timeStamper  
http://timestamp.verisign.com/scripts/timestamp.dll
```

In the above command:

- myfilename is the name of the file that needs to be signed.
- displayname is the description of the file that will show up in the certificate.
- http://www.mycompany.com is a uniform resource locator (URL) where the user can find more information about the file being downloaded.
- mycredentials.spc is the certificate file that was obtained in Step 3.
- myprivatekey.pvk is the private key generated during Step 2 that is securely stored on a diskette.
- http://timestamp.verisign.com/scripts/timestamp.dll is the URL for the VeriSign timestamping service. Please note that “timestamp.dll” does not contain the letter “e.”

Step 6: Test your signature.

The Microsoft SDK contains a utility called `chktrust.exe`. This may be used to check your signature before distributing your file.

- To test a signed `.exe`, `.dll`, or `.ocx` file, run `chktrust filename`
- To test a signed `.cab` file, run `chktrust -c cabfilename.cab`

If your signing process was successful, this will bring up a certificate. Congratulations, you have just digitally signed your file. When a user downloads this file from a Web site, Internet Explorer will display the same certificate to the user. If the file is tampered with in any way after it has been signed, the user will be notified and given the option of refusing installation.

Conclusion

Microsoft and VeriSign are committed to making the Internet a secure and viable platform for commerce and the distribution of content. With Authenticode and VeriSignCode Signing Digital Certificate, your code will be as safe and trustworthy to your customers as it would be if you shrink-wrapped it and sold it off a store shelf.

Visit us at www.Verisign.com for more information.