Driver Package Integrity during Plug and Play Device Installations in Windows Vista

December 17, 2007

Abstract

Windows Vista® now allows any Plug and Play driver to be signed by companies other than Microsoft. This change allows Windows® to verify the origin and integrity of driver packages while giving IT departments and vendors more flexibility in publishing their drivers. This paper gives device vendors and IT professionals a preview of the Windows Vista code-signing policies.

This information applies for the Windows Vista operating system.

For up-to-date information, see “Device Installation Design Guide” in the WDK documentation at

<http://msdn2.microsoft.com/en-us/library/ms790231.aspx>

The current version of this paper is maintained on the Web at:
 <http://www.microsoft.com/whdc/winlogo/drvsign/pnp-driver.mspx>

References and resources discussed here are listed at the end of this paper.

Contents

[Introduction 3](#_Toc173211162)

[Signing Drivers with a Code-Signing Certificate 4](#_Toc173211163)

[Driver Installation 4](#_Toc173211164)

[Ranking Multiple Drivers 6](#_Toc173211165)

[References 8](#_Toc173211166)

Disclaimer

This is a preliminary document and may be changed substantially prior to final commercial release of the software described herein.

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This White Paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, email address, logo, person, place or event is intended or should be inferred.

© 2006–2007 Microsoft Corporation. All rights reserved.

Microsoft, Windows, Windows Server, and Windows Vista are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

#### Revision History

|  |  |
| --- | --- |
| **Date** | **Change** |
| 12/17/2007 | Updated WDK documentation references |
| 4/20/2006 | Created |

# Introduction

Software integrity is a top priority for customers. Concerned by the increase in malicious software that is appearing on the Internet, customers want to be sure that their software has not been corrupted or tampered with. Code signing helps verify software integrity by identifying the publisher and allowing the system to confirm that the code has not been subsequently modified. However, the Windows® XP driver-signing policies make it difficult for device vendors and corporate information technology (IT) departments to deploy drivers.

Windows Vista® addresses these problems by allowing device vendors and IT departments to sign and publish drivers by using their own digital signature.

* IT departments can sign drivers—including in-box drivers—and configure Windows to treat those drivers as equivalent to drivers that are signed by Microsoft. This allows IT departments to silently deploy drivers across their corporation.
* Device vendors can sign drivers and quickly deploy emergency updates to their customers without having to wait for a signature from Microsoft.

Because these drivers are signed, IT departments and end users have the security of knowing that the drivers come from a trusted source and have not been altered in any way since they were published.

Third-party signatures complement rather than replace the signatures given by the Microsoft Windows Hardware Quality Lab (WHQL).

* Third-party signatures guarantee the publisher's identity and driver integrity, but do not provide any information about the quality of the driver.
* WHQL signatures are provided as part of the Windows Logo Program. They not only verify identity and integrity but also indicate that the driver has met or exceeded the quality standard that is defined by the WHQL tests.

Windows Vista contains several other changes and new features that are related to allowing third-party signatures:

* Administrators can control which driver publishers Windows Vista trusts. Windows Vista installs drivers from trusted publishers without prompting. It never installs drivers from publishers that the administrator has chosen not to trust.
* Driver-signing policy is always set to Warn, eliminating the Block and Ignore options that were available in earlier versions of Windows.
* All device setup classes are treated equally. Certclas.inf no longer exists.
* If there are several compatible drivers to choose from, the ranking algorithm Windows Vista uses to pick the best driver includes drivers with third-party signatures. By default, Microsoft signatures take priority over third-party signatures, but IT departments can configure them to be equivalent.

Windows Vista for x64 processors goes a step further. To ensure the integrity of all software that is running in kernel mode, Windows Vista for x64 processors loads only signed kernel-mode components. For more information on this feature, see the white paper titled "Digital Signatures for Kernel Modules on x64-based Systems Running Windows Vista."

# Signing Drivers with a Code-Signing Certificate

Vendors and IT departments that choose to sign and publish drivers must first obtain a code-signing certificate from a trusted certification authority (CA) such as GTE or VeriSign, Inc. A CA validates the identity and entitlement of an applicant and then issues a certificate that the applicant uses to sign its drivers. The signing process stamps the driver with the publisher’s identity and can be used to verify that the driver has not been modified since it was signed.

Windows uses the signature for two purposes:

* **Identification.** Windows uses the identity to allow users to choose whether to trust or not trust a driver's publisher.
* **Integrity.** Windows validates the driver's files against the signature to verify that they have not been modified since they were signed.

For more information about obtaining a certificate, see the white paper titled “Creating, Viewing, and Managing Certificates.”

Windows Vista Plug and Play device installation requires driver packages to have a signed catalog file. Driver packages commonly consist of multiple files; the catalog file contains the signature for the entire package. After a publisher has obtained a certificate, it uses the following procedure to create a signed catalog file for its driver package:

1. Run the Inf2Cat tool to generate a catalog file for the driver package.

2. Run SignTool to sign the catalog file. Note that SignTool can generate both embedded and catalog file signatures.

3. For boot-start drivers, run SignTool to embedded-sign the driver binaries.

SignTool is available in the Windows Driver Kit (WDK), and Inf2Cat is included with the Windows Quality Online Services (Winqual) submission tools. For more information on the signing process and tools, see the white paper titled “Kernel-Mode Code Signing Walkthrough.”

**Note:** Publishers should use the signing tools to timestamp drivers when they sign them. This is valuable if a publisher must revoke the signature later (for example, if the publisher’s certificate is stolen). By timestamping their drivers, publishers can revoke the signature for only those drivers that were signed after a particular date. In absence of time stamps, a publisher must revoke the signature for every driver the publisher has ever signed with that certificate. For more information on revoking signatures, see the white paper titled “Certificate Revocation and Status Checking.”

# Driver Installation

Before installing a driver, Windows analyzes the driver’s signature. If a signature is present, Windows validates all of the driver’s files against that signature. Based on the results of this analysis, Windows puts the driver in one of the following categories:

* Signed by a Windows signing authority.
These drivers are either in-box, signed by WHQL, or signed by Windows Sustained Engineering.
* Signed by a trusted publisher.
These drivers have been signed by a third party, and the user has explicitly chosen to always trust signed drivers from this publisher.
* Signed by an untrusted publisher.
These drivers have been signed by a third party, and the user has explicitly chosen to never trust drivers from this publisher.
* Signed by publisher of unknown trust.
These drivers have been signed by a third party, and the user has not indicated whether to trust this publisher.
* Altered.
These drivers are signed, but Windows has detected that at least one file in the driver package has been altered since the package was signed.
* Unsigned.
These drivers are either unsigned or have an invalid signature. Valid signatures must be created with a certificate that was issued by a trusted CA.

**Note:** Users choose which publishers are trusted or untrusted with the Microsoft Management Console (MMC) Certificate Manager.

After the driver has been categorized, Windows determines whether it should be installed. The process depends on the type of user. For nonadministrative and standard users, Windows does not prompt the user. It automatically installs drivers from trusted publishers and silently refuses to install all others. Administrative users have more flexibility:

* If a driver is signed by a Microsoft Windows signing authority or a trusted publisher, Windows installs the driver without prompting the user.
* If the driver is signed by an untrusted publisher, Windows does not install the driver. Windows does not prompt the user in this case, but logs an error to setupapi.dev.log.
* If the driver was signed by a publisher of unknown trust, Windows prompts the user with the dialog box shown in Figure 1:



Figure 1. Windows Security dialog box for an driver with unknown trust

The user must explicitly choose whether to install this driver. He or she is also given the option of adding the publisher to the list of trusted publishers. If the user selects this option, all future drivers from this publisher are treated as trusted. If the user does not select this option, the publisher remains in the unknown trust category and administrative users continue to receive this prompt if they attempt to install additional drivers from this publisher.

* If the driver lacks a valid signature or has been altered, Windows prompts administrators with the dialog box shown in Figure 2. The user must then explicitly choose whether to install the driver.



Figure 2. Windows Security dialog box for a driver without a valid signature

**Note**: If users want to play next-generation premium content on Windows Vista, such as HD DVD and other formats that are licensed under the Advanced Access Content System (AACS) specification, all kernel-mode components on their system must be signed. That means that, if an administrative user chooses to install an unsigned or altered driver, the system is not allowed to play premium content. For more information, see the white paper titled “Code Signing for Protected Path Components in Windows Vista.”

# Ranking Multiple Drivers

The previous section described the selection process that is used when Windows detects only one driver for a given device. If Windows detects multiple drivers for the same device, it must first determine which driver is the best driver to install. To accomplish this, Windows assigns each driver a rank that is based on the factors that are described below. It then chooses the best ranked driver and launches the process that was described in the previous section to install the driver.

The ranking scheme is based on the following criteria:

1. Driver-signing score.
This score is based on whether the driver is signed and, if so, the identity of the publisher. This scoring integrates naturally with the criteria that was described in the previous section, with trustworthy drivers scored higher than untrustworthy drivers. For details on how Windows determines the driver-signing score, see the next section.

2. Feature score.
This score is based on the **FeatureScore** directive in the driver’s INF file. For more information on **FeatureScore**, see the WDK documentation.

3. Hardware ID score.
This score is based on how closely the device's hardware ID matches the hardware ID in the driver's INF file.

4. Driver date score.
This score is based on the date that was specified by the DriverVer directive in the driver's INF file.

5. Driver version score.
This score is based on the version number as specified by the **DriverVer** directive in the driver's INF file.

Windows first compares the signing scores of the competing drivers. If one driver has the highest signing score, Windows installs that driver. If two or more drivers share the highest signing score, Windows next compares feature scores. If these are equal, Windows compares hardware ID, and so on.

Windows uses the following scheme to calculate a driver's signing score:

* The highest score goes to drivers that are signed by a Microsoft Windows signing authority. This includes in-box drivers, Windows Sustained Engineering signatures, WHQL signatures for Windows Vista, and select WHQL signatures for earlier versions of Windows as determined by the **LowerLogoVersion** value (discussed later).
* The next highest score goes to drivers that are signed by third parties and drivers with WHQL signatures for earlier versions of Windows as specified by their **LowerLogoVersion** value (discussed later). Valid third-party signature types include:

Drivers signed using a certificate that was issued by an enterprise CA.

Drivers signed using a certificate that was issued by a Class 3 CA.

Drivers signed using a certificate that was created by the MakeCert tool.

**Note:** IT departments can configure Windows Vista to give drivers that are signed by third parties equal rank with Microsoft-signed drivers. This allows IT departments to deploy updates over Microsoft signed drivers.

* The third highest score goes to drivers that lack a valid signature, but whose INF file's DDInstall section has a .nt platform extension.
* The fourth highest score goes to drivers that lack a valid signature and whose INF file's DDInstall section does not have a .nt platform extension. A driver of this type might be intended for Windows 95, Windows 98, or Windows Millennium.
* The fifth and worst score goes to drivers that are not signed or whose signing state is unknown.

The **LowerLogoVersion** device setup class property specifies the version of Windows for which the driver was signed. It is used to determine which WHQL-signed drivers receive the best signing score and which receive the second best.

* Drivers that have a WHQL signature for a Windows version that is the same or later than that specified by the **LowerLogoVersion** property receive the best score.
* Drivers that have a WHQL signature for earlier versions than that specified by the **LowerLogoVersion** property receive the second best score.

The default **LowerLogoVersion** value for a system-defined device setup class is "5.1". This means that drivers that have a WHQL signature for Windows Server® 2003 and Windows XP have the same signing score as a driver that is signed by Microsoft for Windows Vista.

# References

White Papers [WHDC Web site]:

Kernel-Mode Code Signing Walkthrough

<http://www.microsoft.com/whdc/winlogo/drvsign/kmcs_walkthrough.mspx>

Code-Signing Best Practices

<http://www.microsoft.com/whdc/winlogo/drvsign/best_practices.mspx>

Code Signing for Protected Path Components in Windows Vista

<http://www.microsoft.com/whdc/winlogo/drvsign/Pmp-sign.mspx>

Digital Signatures for Kernel Modules on Systems Running Windows Vista

<http://www.microsoft.com/whdc/system/platform/64bit/kmsigning.mspx>

Using Authenticode to Digitally Sign Driver Packages for Windows Server 2003

<http://www.microsoft.com/whdc/driver/install/authenticode.mspx>

MSDN:

Creating, Viewing, and Managing Certificates

<http://go.microsoft.com/fwlink/?LinkId=95777>

Introduction to Code Signing

<http://go.microsoft.com/fwlink/?LinkId=95781>

MakeCert

<http://go.microsoft.com/fwlink/?LinkId=95782>

TechNet:

Building an Enterprise Root Certification Authority in Small and Medium Businesses

<http://go.microsoft.com/fwlink/?LinkId=59549>

Certificate Revocation and Status Checking

<http://www.microsoft.com/technet/prodtechnol/winxppro/support/tshtcrl.mspx>

Step-By-Step Guide to Device Driver Signing and Staging

<http://technet2.microsoft.com/windowsserver2008/en/library/4bbbeaa0-f7d6-4816-8a3a-43242d71d5361033.mspx?mfr=true>

Windows Driver Kit
(particularly the sections titled “How Setup Selects Drivers” and “Signing Drivers for Development and Test”)

<http://www.microsoft.com/whdc/driver/WDK/default.mspx>