Static Driver Verifier for WDM Drivers: WHDC Lab

December 20, 2007 — WHDC Preview Draft Version 0.9

Abstract: This lab introduces the basic features of the Static Driver Verifier (SDV) and demonstrates how to use SDV to examine a Windows Driver Model (WDM) driver.

Feedback: Please tell us whether this preview lab is useful to you. Give us your comments at <http://go.microsoft.com/fwlink/?LinkId=101534>.

Contents

[Lab Overview 3](#_Toc180548246)

[Time to Complete This Lab 3](#_Toc180548247)

[Lab Objectives 3](#_Toc180548248)

[Prerequisites 4](#_Toc180548249)

[Feedback for This Lab 4](#_Toc180548250)

[Lab Setup 4](#_Toc180548251)

[Tips for This Lab 4](#_Toc180548252)

[Exercise 1: Preparing to Use SDV to Verify Fail\_Driver2 6](#_Toc180548253)

[Task 1: Select Role-Based Scanning 6](#_Toc180548254)

[Task 2: Confirm Role-Type Definitions 6](#_Toc180548255)

[Task 3: Process the Library 7](#_Toc180548256)

[Summary of Exercise 1 8](#_Toc180548257)

[Exercise 2: Using SDV to Verify WDM Rules 8](#_Toc180548258)

[Task 1: Verify WDM Rule Usage 8](#_Toc180548259)

[Summary of Exercise 2 9](#_Toc180548260)

[Exercise 3: Examining the SDV Results 10](#_Toc180548261)

[Task 1: Examine the Static Driver Verifier Report Page 11](#_Toc180548262)

[Summary of Exercise 3 13](#_Toc180548263)

[Resources 14](#_Toc180548264)

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.

© 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.

# Lab Overview

This lab introduces Static Driver Verifier (SDV), a compile-time testing tool that systematically tests all code paths through the driver to detect violations of Windows® Driver Model (WDM) rules. In completing the three exercises, you will:

* Prepare files for running SDV.
* Scan a sample driver that has intentional coding errors.
* Use the SDV Defect Viewer to review the results of the verification.

Each exercise builds upon the work that was completed in the previous exercise.

**About the WDM Fail\_Driver2 sample driver.** This sample driver:

* Contains coding errors that SDV detects.
* Is installed with the Windows Driver Kit (WDK) at:

%*wdk*%\tools\sdv\samples\fail\_drivers\wdm\fail\_driver2

where *%wdk%* represents the root installation directory for the WDK.

* Consists of two parts:

A support library in the fail\_driver2\library folder.

The actual driver in the fail\_driver2\driver folder.

The library and the driver each consist of a single header file and a single source file.

## Time to Complete This Lab

This lab is designed to be completed in 45 minutes.

## Lab Objectives

After completing this lab, you will be able to:

* Prepare the build environment to use SDV to detect WDM rule violations in the Fail\_Driver2 sample driver.
* Verify four WDM rules that find intentional defects in the sample driver code.
* Use the results to trace defects in the driver code.
* Identify and correct common errors in WDM drivers.

## Prerequisites

To get the most benefit from this lab, you should:

* Know how to use a WDK build environment window.
* Be familiar with SDV as described in the following resources:

*Developing Drivers with the Windows Driver Foundation* from Microsoft Press.

White papers about SDV on the WHDC Web site.

WDK documentation for SDV.

You can find links to the resources at the end of this lab manual.

## Feedback for This Lab

To provide feedback about this lab:  
 <http://go.microsoft.com/fwlink/?LinkId=101534>

# Lab Setup

SDV runs on Windows XP and later versions of Windows. The tool supports free and checked build environments for both x86 and x64 architectures.

This lab requires the following software and settings:

* WDK version 6001 or later, which includes:

Static Driver Verifier in:   
*%wdk%\*tools\sdv

Fail\_Driver2 sample in:  
*%wdk%\*tools\sdv\samples\fail\_drivers\wdm\fail\_driver2

At the time of this publication, the Windows Server® 2008 WDK build 6001 was in beta, but is available from MS Connect. The final release of the Windows Server 2008 WDK is provided through MS Connect and MSDN®. For information about obtaining current builds of the WDK, see "How to Get the Windows Driver Kit and the Windows Logo Kit."

* Permissions to create, edit, and delete files in the driver's sources directory and in the \tools\sdv subdirectories of the WDK.

## Tips for This Lab

* If you copy code from the online version of this manual, paste it into an instance of Notepad and then copy it from Notepad into your driver.
* If you want to begin the lab in the middle or return to a recoverable state, you can start each exercise from where you last concluded.
* Be sure to install the full WDK on your machine. See "Windows Driver Installation Guide" in Install.htm on the WDK installation media.
* If you are working in a build environment window, you can view a summary of available SDV commands by typing the following command:

staticdv /?

You can view the Static Driver Verifier Help file by typing the following command:

staticdv /help

* If your host computer is running Windows Vista®, the command window that you use for SDV must run with elevated privileges. To run an application or command window with elevated privileges:

On the taskbar, click **Start**, right-click the application, and then click **Run As Administrator**.

If you already have administrative privileges, Windows Vista displays a **User Account Control** dialog box asking for permission to proceed. To run the application, click **Continue**.

If you do not have administrative privileges, Windows Vista asks for administrative credentials.

To open a build environment window

1. Click **Start**, and then click **All Programs**.

2. Click **Windows Driver Kits**, click the latest WDK version (6001 or later), and then click **Build Environments**.

3. Click the appropriate CPU architecture, and then open a checked or free build environment window for the appropriate Windows version.

The build environment window for a specified version of Windows works for that version and all later versions.

For this exercise, use **Windows Vista and Windows Server 2008 x86 Free Build** by navigating to the following:

**Start** > **All Programs > Windows Driver Kits** > **WDK 6001** >   
**Build Environments** > **Windows Vista and Window Server 2008** > **Windows Vista and Windows Server 2008 x86 Free Build Environment**

All of the exercises for this lab are in the following directory:

*%wdk%\*tools\sdv\samples\fail\_drivers\wdm\fail\_driver2

# Exercise 1: Preparing to Use SDV to Verify Fail\_Driver2

To prepare to use SDV on the Fail\_driver2 sample WDM driver:

* Select role-based scanning.
* Confirm that role types are defined for the driver’s entry points.
* Process the library.

## Task 1: Select Role-Based Scanning

Role types provide SDV with information about the intended use of a function, so that SDV can better determine whether a particular bug exists. To enable SDV to analyze a WDM driver by using function role-type declarations, you must change the default method that SDV uses to scan your driver code for entry points.

To define the method that SDV uses to scan the sample driver code for entry points

1. Open a build environment window with elevated privileges.

2. Navigate to the following subdirectory:

*%wdk%\*tools\sdv\data\WDM

3. Use Notepad to open the options file named Sdv-default.xml.

4. Edit the Role\_Based\_Scanning attribute by changing "false" to "true" in the options file.

Change:

<Role\_Based\_Scanning xmlns="http://www.microsoft.com/SDV">false</Role\_Based\_Scanning>

to:

<Role\_Based\_Scanning xmlns="http://www.microsoft.com/SDV">**true**</Role\_Based\_Scanning>

3. Save the Sdv-default.xml file.

Note: In future releases, SDV will use role-based scanning by default so you will not be required to make this change.

For a detailed explanation of role-based scanning, see "Selecting Role-Based Scanning" in the StaticDV help file.

## Task 2: Confirm Role-Type Definitions

To take advantage of SDV capabilities, the role type for each driver-supplied routine in a WDM driver must be declared in one of the driver’s header files.

For an explanation of role-type declarations, see the StaticDV help file and Chapter 24, "Static Driver Verifier," in *Developing Drivers with the Windows Driver Foundation*.

**Note:** Role types have been predefined for Fail\_driver2, so you do not need to create them for this lab, but you can view them to see how they are defined in a driver’s header file, as shown in the following steps.

To confirm role-type definitions for Fail\_driver2’s entry points

1. In the build environment window, navigate to the Fail\_Driver2 directory at:

*%wdk%*\tools\sdv\samples\fail\_drivers\wdm\fail\_driver2\driver

2. Use Notepad to open the Fail\_driver2.h file.

3. Scroll through the file to view the role declarations, which are shown in bold in the following example:

include <wdm.h>

#include "fail\_library2.h"

**DRIVER\_INITIALIZE DriverEntry;**

**DRIVER\_ADD\_DEVICE DriverAddDevice;**

**\_\_drv\_dispatchType(IRP\_MJ\_CREATE)**

**DRIVER\_DISPATCH DispatchCreate;**

**\_\_drv\_dispatchType(IRP\_MJ\_READ)**

**DRIVER\_DISPATCH DispatchRead;**

**\_\_drv\_dispatchType(IRP\_MJ\_WRITE)**

**DRIVER\_DISPATCH DispatchWrite;**

**\_\_drv\_dispatchType(IRP\_MJ\_POWER)**

**DRIVER\_DISPATCH DispatchPower;**

**\_\_drv\_dispatchType(IRP\_MJ\_SYSTEM\_CONTROL)**

**DRIVER\_DISPATCH DispatchSystemControl;**

**DRIVER\_UNLOAD DriverUnload;**

## Task 3: Process the Library

You should verify the driver libraries along with the driver code. Library verification is essential for determining whether a driver complies with SDV rules. For example, if verification does not include a driver’s library code, a driver might appear to omit a required call that is part of the library or the library might contain a call that the driver duplicates, thus causing an error such as releasing a lock twice.

Before you can include a library in the verification of a driver, SDV must process the library. Processing a library prepares the internal representation of the library for use in verifying the driver.

You should process any libraries that you create for use in your drivers. You do not need to process the Windows libraries that provide device driver interfaces (DDIs)—such as Wdm.lib, Wdf.lib, Bufferoverflow.lib, Hal.lib, Ntoskrnl.lib, Wmi.lib, and so on—because the SDV operating system model supplies verification stubs for the functions in the Windows libraries.

To process the library

1. In the build environment window, navigate to the library for WDM Fail\_Driver2:

*%wdk%*\tools\sdv\samples\fail\_drivers\wdm\fail\_driver2\library

2. To process the library, type the following SDV command:

staticdv /lib

The command output will be similar to the following example:

C:\WinDDK\6001\tools\sdv\samples\fail\_drivers\wdm\fail\_driver2\library>  
 staticdv /lib

---------------------------------------------------------------------

Microsoft (R) Windows (R) Static Driver Verifier Version 1.5.315.0

Copyright (C) Microsoft Corporation. All rights reserved.

---------------------------------------------------------------------

Build 'library' ...Done

Publishing :fail\_library2.lib.li to c:\winddk\6001\tools\sdv\samples\fail\_driver

s\wdm\fail\_driver2\library\objfre\_wlh\_x86\i386\fail\_library2.lib.li

The cache contains 1 library.

Compiled 'fail\_library2.lib' ...Done

For more information, see "Library Processing in Static Driver Verifier" in the WDK.

## Summary of Exercise 1

Exercise 1 prepared for SDV verification of the driver through the following steps:

* Confirmed that the Fail\_Driver2 header file contains role-type definitions, which SDV uses to identify the entry points in the driver.
* Prepared the driver’s library for SDV verification.

# Exercise 2: Using SDV to Verify WDM Rules

## Task 1: Verify WDM Rule Usage

After preparing to use SDV, the next step is to identify the rules that SDV will verify in the driver’s code. Fail\_Driver2 has been injected with defects that violate the following four SDV rules for WDM:   
 **SpinLock**  
 **NullExFreePool**  
 **CancelSpinLock**   
 **LowerDriverReturn**

These rules and the related defects are explained in Exercise 3.

**Important:** Be sure to type the **staticdv** commands exactly as shown in these steps. Do not add extra spaces.

To verify WDM rule usage

1. In the build environment window, navigate to the following folder:

*%wdk%*\tools\sdv\samples\fail\_drivers\wdm\fail\_driver2\driver

2. In the folder, the Config.sdv file lists the names of four SDV rules for WDM. To analyze the driver for these rules, type the following command:

staticdv /config:config.sdv

3. Examine the results that appear in the build environment window and verify that "**4 Defects**" appears as shown in the following example:

C:\WinDDK\6001\tools\sdv\samples\fail\_drivers\wdm\fail\_driver2\driver>   
 staticdv /config=config.sdv

---------------------------------------------------------------------

Microsoft (R) Windows (R) Static Driver Verifier Version 1.5.315.0

Copyright (C) Microsoft Corporation. All rights reserved.

---------------------------------------------------------------------

Build 'driver' ...Done

Link 'driver' for [fail\_library2.lib] ...Done

Scan 'driver' ...Done

Warning 'driver' No entry points have been approved in SDV-map.h for this driver.

Compile 'driver' for [sdv\_flat\_harness] ...Done

Link 'driver' for [fail\_library2.lib] ...Done

Compile 'driver' for [sdv\_flat\_simple\_harness] ...Done

Link 'driver' for [fail\_library2.lib] ...Done

Compile 'driver' for [sdv\_flat\_harness] ...Done

Link 'driver' for [fail\_library2.lib] ...Done

Compile 'driver' for [sdv\_flat\_harness] ...Done

Link 'driver' for [fail\_library2.lib] ...Done

Check 'driver' for 'spinlock' ...Running

Check 'driver' for 'nullexfreepool' ...Running

Check 'driver' for 'spinlock' ...Done

Check 'driver' for 'nullexfreepool' ...Done

Check 'driver' for 'cancelspinlock' ...Running

Check 'driver' for 'lowerdriverreturn' ...Running

Check 'driver' for 'cancelspinlock' ...Done

Check 'driver' for 'lowerdriverreturn' ...Done

Static Driver Verifier performed 4 check(s) with:

**4 Defect(s)**

Start Time :4/21/2007 4:39:55 PM and End Time :4/21/2007 4:40:23 PM

Notice that the following warning appears in the staticdv output:

Warning 'driver' No entry points have been approved in SDV-map.h for this driver.

However, in Exercise 1 you confirmed that role types are declared for the driver-supplied routines in the Fail\_Driver2.h file. Therefore, you can safely ignore this warning.

## Summary of Exercise 2

The steps in Exercise 2 specify the rules that SDV verifies and provide preliminary results of the verification. To examine the detailed results, complete the next exercise.

# Exercise 3: Examining the SDV Results

If the preliminary results indicate defects in the driver, you can generate and examine a report in the **Static Driver Verifier Report Page**, as shown in Figure 1.



Figure 1. Graphical User Interface for Static Driver Verifier

The interface contains the following elements:

**➀** The **Trace Tree** pane on the top left displays the sequence of the critical elements of the source code that are executed in the path to the rule violation.

**➁** The **Source Code** pane in the center highlights the corresponding line of code as you step through the source code in the **Trace Tree** pane. Trace fragments appear in red.

Each tab on the **Source Code** pane displays a step in the trace through the particular source code file.

**➂** The **State** pane at the bottom left allows you to examine the values of variables in the driver, the SDV operating system model, and the rule. The values appear as Boolean expressions.

**➃** The status bar at the bottom of the window describes the defect.

**➄** The **Results** pane is shown in detail in Figure 2.

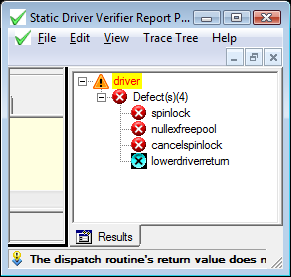


Figure 2. Results Pane in an SDV Report

The **Results** pane shows the results of the verification, including a list of the defects that SDV discovered.

## Task 1: Examine the Static Driver Verifier Report Page

For details about how to use Defect Viewer to filter and manage SDV results, see "Defect Viewer" in the WDK.

To examine the Static Driver Verifier Report Page for Fail\_Driver2

1. In the build environment window, navigate to the following directory:

*%wdk%*\tools\sdv\samples\fail\_drivers\wdm\fail\_driver2\driver

2. To open the Static Driver Verifier Report Page, type the following command:

staticdv /view

3. The **Results** pane on the right side of the **Static Driver Verifier Report Page** lists the rules that SDV verified.

This is area **➄** in Figure 1.

4. To display the trace of the code path to a rule violation, double-click the rule name in the **Results** pane.

This is area **➄** in Figure 1.

If a defect trace is already open, a **Warning** dialog box opens to confirm that you want to view a different defect trace, as shown in Figure 3. To view the trace, click **Yes**.

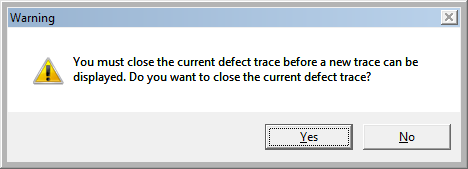


Figure 3. Warning Dialog Box in an SDV Report

5. To review the WDM rule that the code violated, examine the .slic file for the rule by clicking the tab in the **Source Code** pane in Defect Viewer.

This is area **➁** in Figure 1.

6. In the **Trace Tree** pane, step through the execution path by clicking the + icon to expand the execution path, and then clicking a specific line to examine the source code in the **Source Code** pane.

This is area **➀** in Figure 1.

7. Look for the following when you view the violation of the **SpinLock** rule for Fail\_Driver2:

Inside the DispatchRead dispatch routine of the Fail\_Driver2 sample, the driver calls the LibraryRoutine3 library function.

Inside this library function, the **KeAcquireSpinLock** system function acquires the spin lock that was passed to LibraryRoutine3.

The DispatchRead dispatch routine returns without releasing the lock acquired in the library function.

SDV finds a defect because the driver should not hold a spin lock when the dispatch routine returns.

8. Look for the following when you view the violation of the **NullExFreePool** rule for Fail\_Driver2:

Inside the DispatchCreate dispatch routine of the Fail\_Driver2 driver, the driver sets a pointer to NULL.

Inside the same dispatch routine, the driver calls the LibraryRoutine2 library function.

Inside this library function, the **ExFreePool** system function frees the pointer that was passed to the library function.

SDV finds a defect because the driver passes a NULL pointer to **ExFreePool**, which can cause a bug check.

9. Look for the following when you view the violation of the **CancelSpinLock** rule for Fail\_Driver2:

Inside the DispatchPower dispatch routine of the Fail\_Driver2 driver, the driver calls the LibraryRoutine5 library function.

Inside this library function, the **IoAcquireCancelSpinLock** system function acquires the spin lock that was passed to LibraryRoutine5.

The DispatchPower dispatch routine returns without releasing the spin lock that was acquired in the library function.

SDV finds a defect because the driver should not hold the cancel spin lock when the dispatch routine returns.

10. Look for the following when you view the violation of the **LowerDriverReturn** rule for Fail\_Driver2:

Inside the DispatchSystemControl dispatch routine of the Fail\_Driver2 driver, the driver calls the LibraryRoutine6 library function.

Inside this library function, the **IoCallDriver** system function is called. The LibraryRoutine6 function returns the status value from **IoCallDriver**.

The DispatchSystemControl dispatch routine returns STATUS\_SUCCESS, ignoring the status value that LibraryRoutine6 retrieved from **IoCallDriver**.

SDV finds a defect because the return value of the dispatch routine does not match the value returned from the lower driver in the driver stack.

## Summary of Exercise 3

The steps in Exercise 3 provided:

* An introduction to the Static Driver Defect Report Page.
* An examination of code execution.
* Tips for identifying errors that SDV finds in source code.

# Resources

* To find out more about SDV and WDM rules, send e‑mail to: **sdvfdbk@microsoft.com**

### Tools and Files:

SDV in the WDK

*%wdk%*\tools\sdv   
(Windows Vista or later version)

Wdm.h header file

*%wdk%*\inc\ddk

### WDK Sample Drivers Annotated for SDV:

OsrUsbFx2

*%wdk%*\src\kmdf\osrusbfx2

KMDF fail drivers

*%wdk%*\tools\sdv\samples\fail\_drivers\kmdf

WDM fail drivers

*%wdk%*\tools\sdv\samples\fail\_drivers\wdm

### WDK Documentation:

Static Driver Verifier

<http://go.microsoft.com/fwlink/?LinkId=80084>

Library Processing in Static Driver Verifier

<http://go.microsoft.com/fwlink/?LinkId=80077>

Defect Viewer

<http://go.microsoft.com/fwlink/?LinkId=80066>

### Other Resources:

Static Driver Verifier - Finding Driver Bugs at Compile-Time

<http://go.microsoft.com/fwlink/?LinkId=80082>

How to Get the Windows Driver Kit and the Windows Logo Kit

<http://www.microsoft.com/whdc/DevTools/WDK/WDKpkg.mspx>

Microsoft Research: SLAM

[http://research.microsoft.com/slam](http://research.microsoft.com/slam%20)

Chapter 24: "Static Driver Verifier," in *Developing Drivers with the Windows Driver Foundation*

<http://www.microsoft.com/whdc/driver/wdf/wdfbook.mspx>