CMSC 412 - Fall 2010
William Arbaugh
11/10/2010

LAB 5 - KERNEL TO USER LAND COMMUNICATIONS

Introduction

The purpose of this lab is to introduce you to passing information between a kernel mode driver and user land. There are a number of ways to do this, but this lab will focus on using IOCTL's.

This project consists of two parts. The first part involves reading and understanding the code in a sample driver from the WDK aka DDK. The second part involves researching and implementing kernel mode process list utility for Windows XP that DOES NOT use any Windows API (to find and walk the process list).

Background Material

It is highly recommended that you read and understand the following articles:

Introduction to Driver Development
http://www.codeproject.com/KB/system/driverdev.aspx

Introduction to IOCTL's
http://www.codeproject.com/KB/system/driverdev2.aspx

Introduction to Driver Contexts
http://www.codeproject.com/KB/system/driverdev3.aspx

Sample Driver

The WDK includes a number of sample driver projects. One, "How to Handle Different IOCTLS", demonstrates kernel to user communications using the various forms of IOCTL communications provided by Windows. You'll find this sample under the src\general\ioctl\kmdf path of your WDK/DDK (latest version).

You have two tasks for this effort:

1. Read and understand both the driver and user land code for this project. This involves building the projects, running, and testing them.

2. Understand that differences between METHOD_IN_DIRECT, METHOD_OUT_DIRECT, METHOD_BUFFERED, and METHOD_NEITHER.

3. Pick one of the methods to use for this project and justify your choice in the comments section of your source.

4. Determine a method to find and walk the list of active processes in Windows XP that does not use an internal Windows API function call, i.e. do not use ZwQuerySystemInformation. NOTE: This is going to require a significant amount of research on your part.

5. Implement and test both the driver and user mode application. The user mode program, when run, will print for each of the active processes on the system the following triple: PID IMAGE_NAME PROCESS_STATE.

Submission

Submit the source for your user mode application, and driver along with ALL of your build/ project files in a single tar file to the submit server. More information on the submission requirements will be provided by the TA's during discussions sections.

Grading

Grading is as follows:

Style - 10%

Approach - 20% (this includes the IOCTL method you used along with your justification there of along with how you find the active process list)

Correctness - 70% (does your code compile without errors, and does it list all processes correctly without crashing)

Due Date

This project is due 11/17/2010 at midnight.