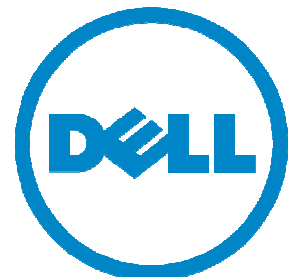


Remote Microsoft® Windows Server® OS Kernel Debugging Using Dell Windows Debugger Utility (DWDU)

Dell | Product Group

Niranjana Vedavyas

March 2, 2010



THIS WHITE PAPER IS FOR INFORMATIONAL PURPOSES ONLY, AND MAY CONTAIN TYPOGRAPHICAL ERRORS AND TECHNICAL INACCURACIES. THE CONTENT IS PROVIDED AS IS, WITHOUT EXPRESS OR IMPLIED WARRANTIES OF ANY KIND.

© 2010 Dell Inc. All rights reserved. Reproduction of this material in any manner whatsoever without the express written permission of Dell Inc. is strictly forbidden. For more information, contact Dell.

Dell, the *DELL* logo, and the *DELL* badge, are trademarks of Dell Inc. *Microsoft*, *Windows*, and *Windows Server* are either trademarks or registered trademarks of Microsoft Corporation in the United States and/or other countries.

Table of Contents

Executive Summary	2
Introduction	3
Overview	3
Server Configuration	3
Client Configuration.....	6
Conclusion	9

Executive Summary

Dell recently launched a new tool, Dell Windows® Debugger Utility (DWDU), to perform server Windows kernel debugging using an Out Of Band (OOB) interface. The tool is designed so that the Dell Remote Access Controller / integrated Dell Remote Access Controller (DRAC/iDRAC) firmware is untouched, and only the host requires new software.

Introduction

This White Paper provides information on using Dell Windows Debugger Utility (DWDU) the way to remotely debug the Windows kernel in case of operating system issues. Currently, the kernel can be debugged using tools such as Windows Debugger(`WinDbg`) or Kernel Debugger(`KD`) that are running on a client machine that is connected to the server using a serial interface; the debugging tool must be available on the machine connected to the server. This paper details how to debug a Windows Server OS on a Dell server over the network utilizing DWDU along with the Microsoft tools (`WinDbg` and `KD`).

Overview

Dell DRAC products have a Serial over LAN (SOL) feature that redirects serial data over the network. It is mainly used to perform serial console redirection of the server over the network to remote systems. This traffic can be captured using tools like `Ipmitool` and by enabling the SOL channel to DRAC. The same channel can also be used to redirect the debug data over the network. Once this data reaches the client, it can be serialized and provided to the Microsoft tools; these tools use this connection as a serial connection even though the data is coming over the network.

DWDU has the following features:

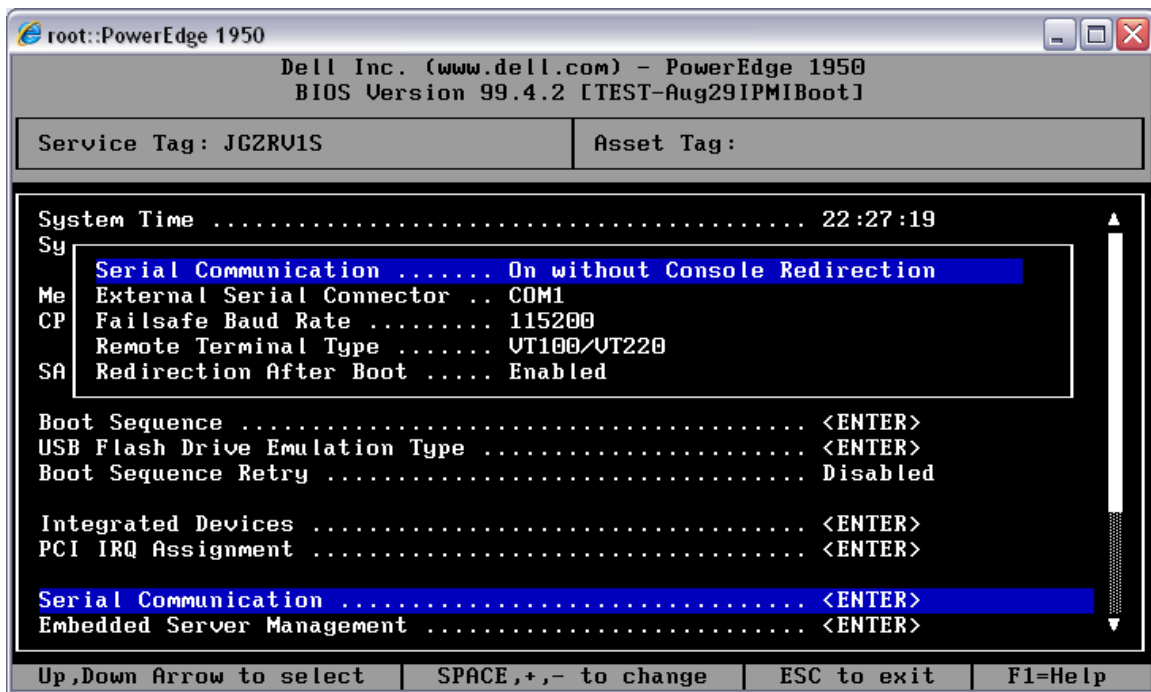
1. Remote debugging of the Windows Server kernel
2. Remotely monitor the Windows boot process
3. Real-time Windows Driver debugging

Server Configuration

1. BIOS Setup

The BIOS should be configured so that only the Windows Debug data is redirected over the SOL connection. The BIOS settings are shown in Figure 1 below.

Figure 1: BIOS Settings



2. Enable Windows Server in Debug mode

Configuring Win2K3 Servers

To enable the debug mode, open the `Boot.ini` file on the target computer and add any of the following options to configure the environment:

- `/debug` - turns on the kernel debugger.
- `/debugport` - specifies the serial port to be used by the kernel debugger; use COM2 for DRAC5/iDRAC tower and rack systems and COM1 for iDRAC modular systems (10G and 11G).
- `/crashdebug` - sends debug information only when a fatal system error (FSE) occurs.
- `/baudrate` - sets the kernel debugger baud rate; set the baud rate to 115200.

Example of a `boot.ini` file:

```
[boot loader]

timeout=5

default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS

[operating systems]

multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Windows Debug (com2) "
/fastdetect
```

```
/debug /debugport=com2 /baudrate=115200
```

Configuring Post Win2K3 Servers

Consider the following when setting up the OS:

- Enable Windows OS for debug
 - Bcdedit /debug ON
- For DRAC5,iDRAC6 rack and tower servers:
 - Set the debug settings to point to COM2

```
Bcdedit /dbgsettings serial baudrate:115200 debugport:2
```

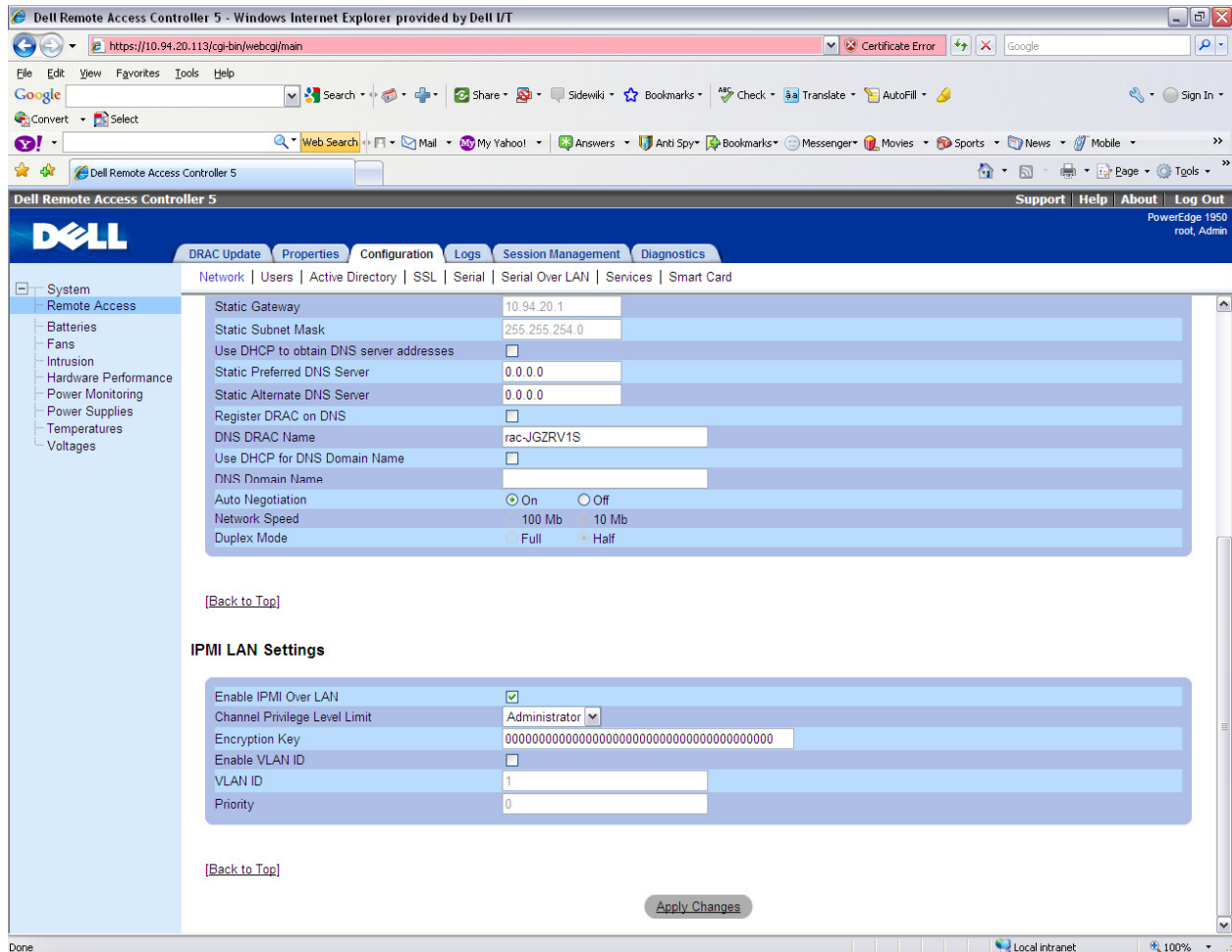
- For 10th generation and newer Dell blade servers:
 - Set the debug settings to point to COM1

```
Bcdedit /dbgsettings serial baudrate: 115200 debugport: 1
```

3. DRAC/iDRAC Configuration

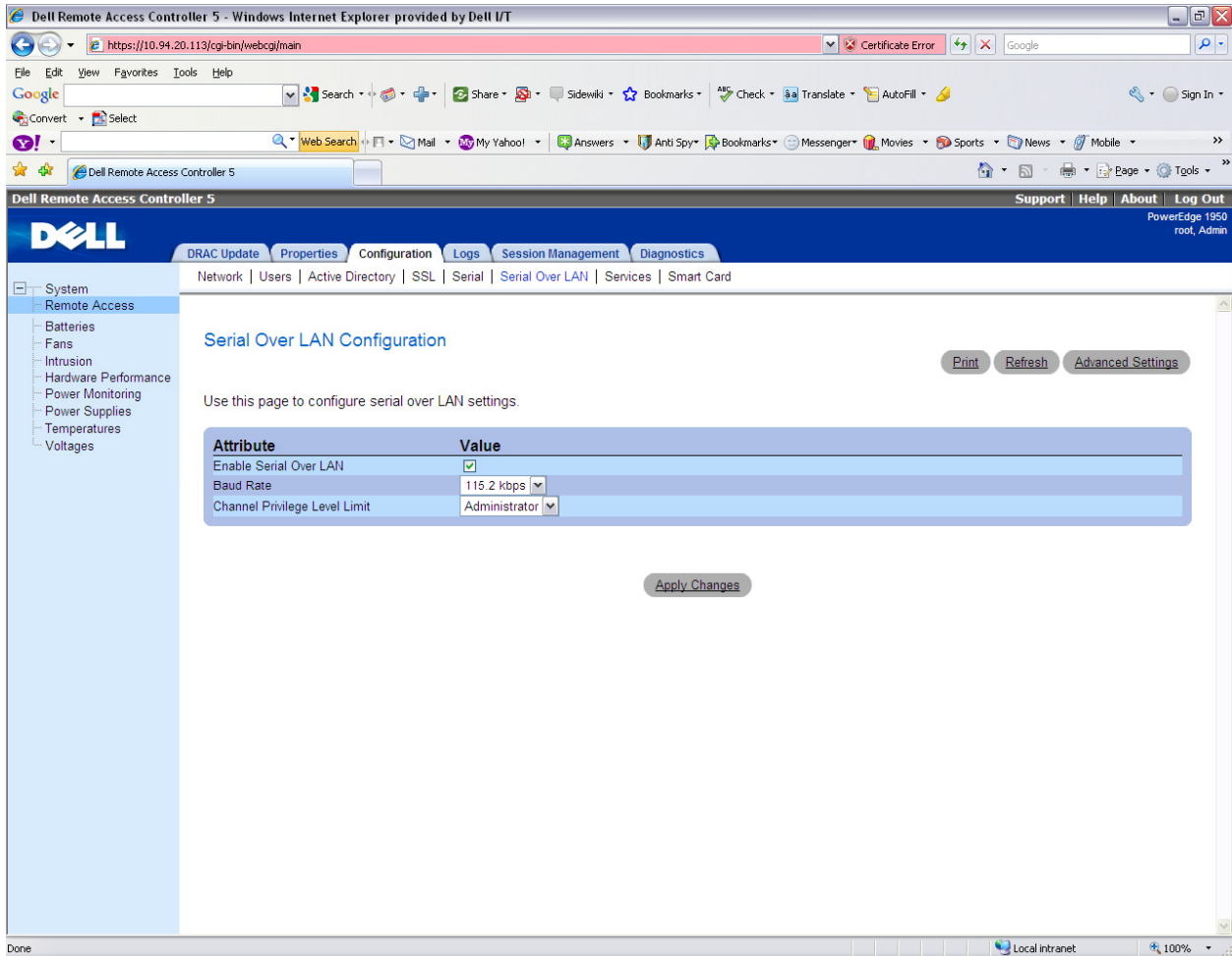
- Enable IPMI Over LAN with following settings as shown in Figure 2 below

Figure 2: IPMI over LAN settings



- Enable Serial Over LAN with following settings as shown in Figure 3 below

Figure 3: Serial over LAN settings



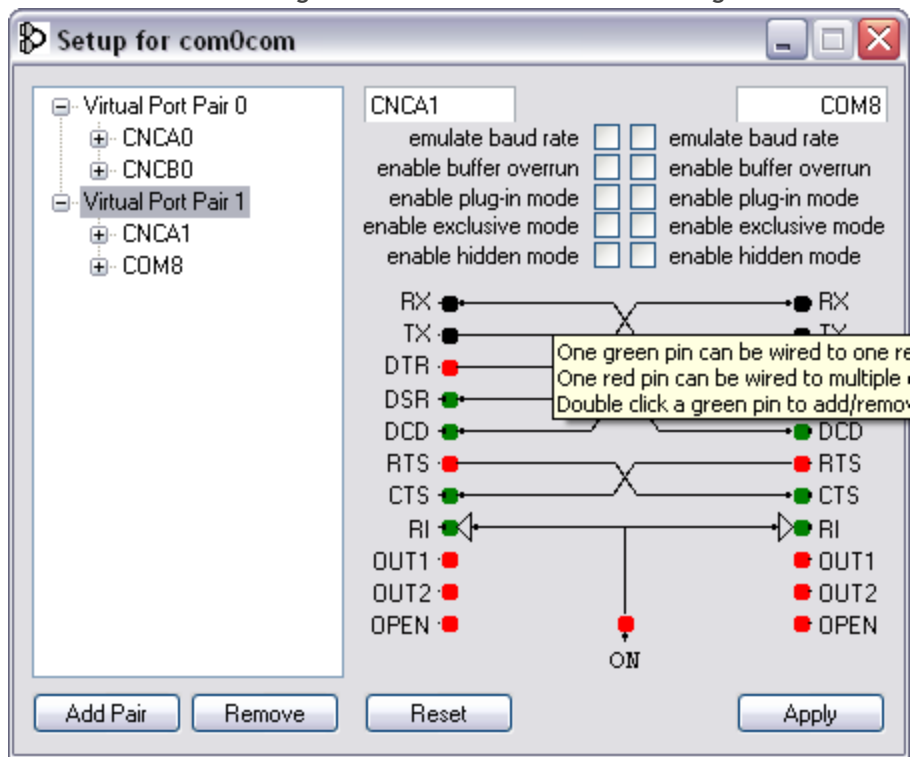
Client Configuration

1. Debugging tools - there are two Kernel debugging tools available from Microsoft
WinDbg and KD
 - WinDbg is a multi-purpose debugger for Microsoft Windows that is distributed on the Web by Microsoft.
 - KD is the command line debugging tool

These tools are prerequisite for remote debugging and can be downloaded from:
<http://www.microsoft.com/whdc/devtools/debugging/default.aspx>

2. NULL Modem Emulator - the NULL modem emulator (com0com) is an open source kernel-mode virtual serial port driver for Windows. You can create an unlimited number of virtual COM port pairs using this tool and use any pair to connect one COM port based application to another. This tool must be installed on the client system, and configured with a COM port pairs for Dwindbg (CNCA1 and COM8). Figure 4 below provides the needed setup information.

Figure 4: NULL Modem Emulator Settings

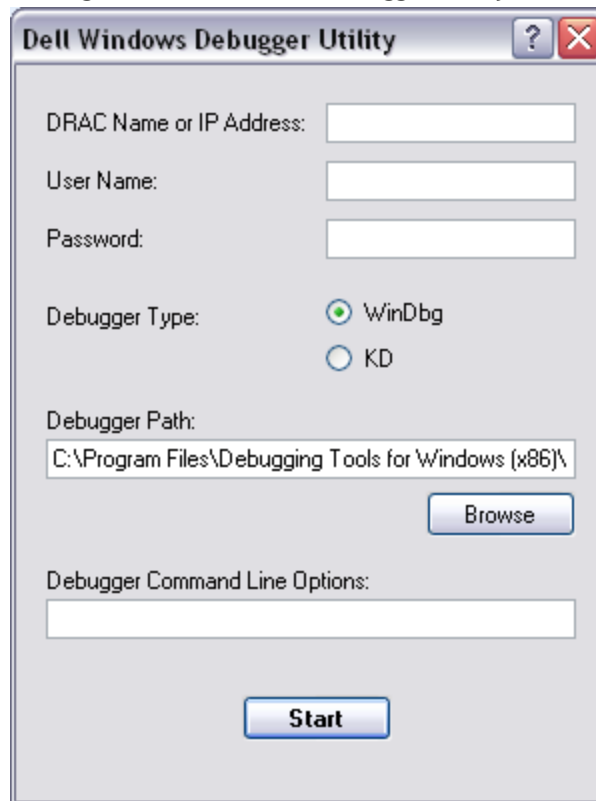


For more information, go to the following Website:

<http://sourceforge.net/projects/com0com/>

3. Dwindbg tool - download Dwindbg.msi from <https://support.dell.com>, and install it on the client machine. Once it is installed, the Windows Debugger Utility icon is on the desktop. Click the icon to display the following screen, as is shown in Figure 5.

Figure 5: Dell Windows Debugger Utility



Enter all of the required parameters - **DRAC Name or IP address**, **User Name**, **Password**, and **Debugger Type** and press **Start**. The selected Microsoft debugger will be launched and the environment is ready for debugging. Figure 6 shows the launch screen for WinDbg, and Figure 7 shows the launch screen for KD.

Figure 6: Launch Screen for WinDbg

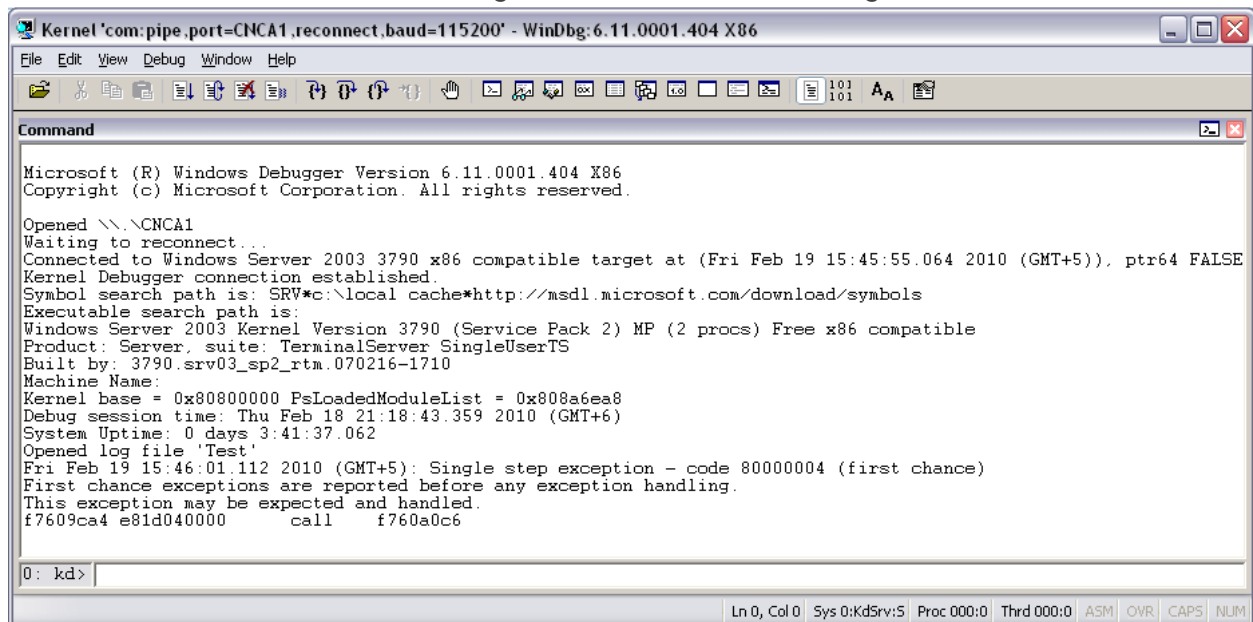


Figure 7: Launch Screen for KD

```

C:\Program Files\Debugging Tools for Windows (x86)\kd.exe
Microsoft (R) Windows Debugger Version 6.11.0001.404 X86
Copyright (c) Microsoft Corporation. All rights reserved.

Opened \\.\CNCA1
Waiting to reconnect...
Connected to Windows Server 2003 3790 x86 compatible target at <Fri Feb 19 15:5
:18.752 2010 (GMT+5)>, ptr64 FALSE
Kernel Debugger connection established.
Symbol search path is: *** Invalid ***
*****
* Symbol loading may be unreliable without a symbol search path.          *
* Use .symfix to have the debugger choose a symbol path.                  *
* After setting your symbol path, use .reload to refresh symbol locations. *
*****
Executable search path is:
*****
* Symbols can not be loaded because symbol path is not initialized.      *
*
* The Symbol Path can be set by:
*   using the _NT_SYMBOL_PATH environment variable.
*   using the -y <symbol_path> argument when starting the debugger.
*   using .sympath and .sympath+
*****
*** ERROR: Symbol file could not be found. Defaulted to export symbols for ntk
pamp.exe -
Windows Server 2003 Kernel Version 3790 (Service Pack 2) MP (2 procs) Free x86
ompatible
Product: Server, suite: TerminalServer SingleUserTS
Built by: 3790.srv03_sp2_rtm.070216-1710
Machine Name:
Kernel base = 0x80800000 PsLoadedModuleList = 0x808a6ea8
Debug session time: Thu Feb 18 21:18:43.359 2010 (GMT+6)
System Uptime: 0 days 3:41:37.062
Single step exception - code 80000004 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
f7609ca4 e81d040000      call    f760a0c6
0: kd>

```

Conclusion

Dell Windows Debugger Utility (DWDU) is a tool that can be used to perform remote Windows debugging in an environment where servers are maintained in a secured lab, and physical access to the servers is difficult. This tool provides a mechanism to remotely connect to the server through the server's DRAC infrastructure, and then can be used to capture debug traces and perform remote debugging activities.