Knowledge Base

How to Set Up a Remote Debug Session Using a Null Modem Cable

PSS ID Number: 151981

Article Last Modified on 3/2/2004

The information in this article applies to:

- Microsoft Windows Server 2003, Standard Edition
- Microsoft Windows Server 2003, Enterprise Edition
- Microsoft Windows Server 2003, Datacenter Edition
- Microsoft Windows 2000 Server
- Microsoft Windows 2000 Advanced Server
- Microsoft Windows 2000 Professional
- Microsoft Windows 2000 Datacenter Server
- Microsoft Windows NT Workstation 3.5
- Microsoft Windows NT Workstation 3.51
- Microsoft Windows NT Workstation 4.0
- Microsoft Windows NT Server 3.5
- Microsoft Windows NT Server 3.51
- Microsoft Windows NT Server 4.0
- Microsoft Windows Small Business Server 2003, Premium Edition
- Microsoft Windows Small Business Server 2003, Standard Edition

This article was previously published under Q151981

SUMMARY

This article describes how to set up the debugger on a Windows-based computer by using a null modem cable that is attached to another Windows-based computer.

MORE INFORMATION

This section includes the following topics:

- Determine the need for the kernel debugger
- About remote debugging
- Use a null modem cable
- Change the attributes on the Boot.ini file
- Debugger options
- Configure the debugger for the computer
- Set up the null modem cable
- Configure the host computer for a Remote.exe session

Determine the Need for the Kernel Debugger

The kernel debugger that comes with Windows provides information about the computer that is typically not available without the aid of the debugger. Windows can turn on the kernel debugger from the Windows Start-Up menu (Boot.ini). When you turn on the kernel debugger, Windows debugger can output debugging information to a remote debugger for a remote user to analyze. This is typically done at the request of a Microsoft support professional for analyzing a fatal error in Windows that cannot be diagnosed from the Memory.dmp file or when a Memory.dmp file is not produced.

About Remote Debugging

For our discussion the target computer is the computer that is being debugged and the host computer is the computer doing the debugging. The process of remote debugging occurs when two computers are connected by the serial communication ports. The target computer and the host computer are running the Windows kernel debuggers, which communicate using a special debug API and protocol. To download the Debugging Tools for Windows, visit the following Microsoft Web site:

http://www.microsoft.com/whdc/ddk/debugging/default.mspx

Use a Null Modem Cable

If you connect the null modem cable to a serial port, the debugger can run at its current top speed of 19200, compared to the speed of 9600 over a modem. This configuration is generally considered more reliable than using a modem for the following reasons:

- This configuration does not involve configuring a modem.
- There is no chance of a modem or line failure.
- You can debug a computer locally.
- This configuration provides faster debugging.

The most common method is to set up a target computer for debugging and the host computer for doing the debugging by using a Remote.exe session. This configuration permits a technical support professional to connect to the Remote.exe session by means of a remote access connection and debug the target computer without physically being present at the target computer.

Change the Attributes on the Boot.ini File

To set up the target computer, edit Boot.ini in the boot partition, and then add the appropriate debugger options. Use a standard ASCII text editor. The Boot.ini file is hidden in the boot partition directory. The Boot.ini file attributes are hidden, read-only and system. Before you can edit the Boot.ini file, you must change these attributes.

To change the attributes of the Boot.ini file, at the command prompt type `attrib -h -r -s c:\boot.ini`, and then press Enter. Use Notepad or another text editor to edit the Boot.ini file.

After you edit the file, type `attrib +h +r +s c:\boot.ini` at the command prompt, and then press ENTER to restore the read-only, hidden, and system file attributes. Alternatively, you can use File Manager instead of a command prompt to change the file attributes. To do so:

1. In File Manager, click **By File Type** on the **View** menu.
2. Click **Show Hidden/System Files**, and then click **OK**.
3. Click **Boot.ini**, and then click **Properties** on the **File** menu.
4. Click to clear the following check boxes, and then click **OK**:
   - Read Only
   - Hidden
   - System

Debugger Options

After you open the Boot.ini file on the target computer for editing, you can add any of the following options to configure the debugger environment on the target computer:

- **/debug**: Use this option to turn on the kernel debugger.
- **/debugport**: Use this option to specify the serial port used by the kernel debugger. If no serial port is specified, the debugger always uses COM2.
- **/crashdebug**: Use this option to send debug information only when a fatal system error (FSE) occurs.
- **/baudrate**: Use this option to set the Baud rate that the kernel debugger uses. The default baud rate is 19200. A baud of 9600 is the typical rate for remote debugging over a modem.

Configure the Debugger for the Computer

There are several options available for configuring the debugger on the target computer. To add any one of the previous options to the Boot.ini file, use the following example.

The following text is a typical Boot.ini. Each entry in the [operating systems] section corresponds to the options that are listed in the boot menu during a typical system startup.

```
[boot loader]
timeout=30
```

```
default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS
[operating systems]
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Windows NT" Version 3.5"
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Windows NT" Version 3.5 [VGA
mode]" /BASEVIDEO
C:\="MS-DOS"
```

To turn on the debugger use COM1 to perform the following procedure:

1. Select the startup option that you typically use and add the **/debug** switch at the end of the line.
2. To specify the communications port, add the **/debugport=com1** switch.

```
[boot loader]
timeout=30
default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS
[operating systems]
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Windows NT" Version 3.5"
/debug /debugport=com1
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Windows NT" Version 3.5 [VGA
mode]" /BASEVIDEO
C:\="MS-DOS"
```

Set Up the Null Modem Cable

A null modem cable is a serial cable that has been configured to allow data to be sent between two serial ports. Typically, you connect a serial port to a serial device. In this case, you must configure the target computer to accept a serial-port-to-serial-port connection. It is important that you use a null modem cable because if you use a typical serial cable the target computer expects a serial device, not another serial port, and you cannot debug the computer.

When you use a null modem to make a direct serial connection between two computers, your cable must be wired as shown in the following tables.

```
9-Pin NULL Modem Cabling

Remote host
serial port connector Host system
serial port connector

Signal
3  2  Transmit Data
2  3  Receive Data
7  8  Request to Send
8  7  Clear to Send
6, 1  4  Data Set Ready and Carrier Detect
5  5  Signal Ground
4  6, 1  Data Terminal Ready
25-Pin NULL Modem Cabling
```

Configure the Host Computer for a Remote.exe Session

In this scenario, the host computer is being set up to do a debug of the target computer as if there were a technical support professional sitting at the host computer. Remote.exe runs to permit this debug session to be accessed remotely through a remote access connection.

Remote.exe is a command-line utility that you can use to run command-line programs on remote computers. Remote.exe uses two parts, the server component and the client component. To use Remote.exe, you must first start the server end on the computer that you are debugging from. When you do so, other users can connect to your debugging session by using the client portion of Remote.exe. Remote.exe comes with the Resource Kit. For more details and the command syntax of this utility, see the Resource Kit online help.

The Windows installation CD and the Windows Resource Kit CD is available to achieve this setup.

1. Set up a remote access account on the network for the technical support professional. This information, including the name of the host computer, must be documented and ready to give to the support professional.
2. The host computer must be configured as described in the following Microsoft Knowledge Base article. This article describes in detail how to set up the symbols on the host computer to debug the target computer. For additional information, click the article number below to view the article in the Microsoft Knowledge

Base:

148659 How to Set Up Windows NT Debug Symbols

3. Microsoft recommends the best way to run I386KD is to copy all the files that are located in the Support\Debug\I386\ folder (assuming that you are debugging an I386 computer) to a folder on the hard disk, and change to that folder.
4. Copy Remote.exe from the Resource Kit to the same folder.
5. The I386KD relies on environment variables for information it needs to run successfully. I386KD uses many environment variables. In this scenario, the minimal I386KD must have is the _NT_SYMBOL_PATH and the _NT_DEBUG_PORT variable. Use the **set** command to set the environment variables. The _NT_SYMBOL_PATH variable points to the path of the symbols file that the debugger uses for the debug session. The _NT_DEBUG_PORT variable points to the active serial port being used on the host computer for debugging. If this serial port is not serial port 2, the _NT_DEBUG_PORT environment variable must be used.

   Example:
   The host computer null modem cable connects to serial port 1, and then copies the symbols to the C:\Debug\Symbols folder. At a command prompt, type the following command:

   ```
   set_nt_debug_port=com1
   set_nt_symbol_path=c\debug\symbols
   set_nt_debug_log_file_append=c:\temp\debug.log
   ```
6. Make sure that the debugging tools were step up correctly. To do this, follow these steps:
   a. Click **Start**, click **Run**, in the **Open** box, type `Cmd`, and then click **OK**.
   b. Change directory to the path location where you copied the files in step 3 of this procedure.
   c. Type `i386kd -v`, and then press **Enter**.
   d. Press **Ctrl**-**C** to break into the Target. You will receive a **kd** prompt and the Target system will stop responding (hang) because you have broken into it with the debugger.
   e. Type `!process` at the **kd** prompt, and then press **Enter**. The system returns a list of processes. This lets you know that the debugger is working correctly.
   f. Type `g`, and then press **Enter** to tell the Target to Go. This will allow the Target to resume normal operation.
   g. Type `exit` and press `Enter` to close the command window.
7. Start the debug session, and then remote the session out. At a command prompt type the following:

   ```
   remote /s "i386kd -v" debug
   ```

Additional Information Resources

- Windows Resource Kit
- The Driver Development Kit Online help
- To view the Kernel-Debug How-To series of articles search on the keyword "debugref" in the Microsoft Knowledge Base.

Additional query words: 3.50 3.51 prodnt debugref

Keywords: KB151981
Technology: kbSBServ2003Pre kbSBServ2003Search kbSBServ2003St kbSBServSearch kbwin2000AdvServ kbwin2000AdvServSearch kbwin2000DataServ kbwin2000DataServSearch kbwin2000Pro kbwin2000ProSearch kbwin2000Search kbwin2000Serv kbwin2000ServSearch kbWinAdvServSearch kbWinDataServSearch kbWinNT350search kbWinNT351search kbWinNT400search kbWinNTS350 kbWinNTS350search kbWinNTS351 kbWinNTS351search kbWinNTS400 kbWinNTS400search kbWinNTsearch kbWinNTSsearch kbWinNTW350 kbWinNTW350search kbWinNTW351 kbWinNTW351search kbWinNTW400 kbWinNTW400search kbWinNTWsearch kbWinServ2003Data kbWinServ2003DataSearch kbWinServ2003Ent kbWinServ2003EntSearch kbWinServ2003Search kbWinServ2003St

---

Send feedback to Microsoft