Sending backup log output to a text file in PowerShell

When running Macrium Reflect from a PowerShell script it may be desirable to pipe the backup log to a text file for further processing. If you try to action this using -RedirectStandardOutput in the script this will result in a 0 byte file.

Why isn't the output populating the file?

Macrium Reflect is a Windows GUI application and, unlike the Console subsystem, GUI applications do not automatically map stdout and stderr streams to a parent console Window. Because of this, Macrium Reflect calls AttachConsole and maps stdout and stderr to the parent console Window buffer - CONOUT$. Unfortunately, this can lead to the output being 'lost' if there is no console Window,

The Solution

Macrium Reflect **v7.1.2899** has added a new parameter in the command line, **-noconsole** , that enables redirection of stdout and stderr. The following solution involves modifying a generated script using Macrium Reflect. You can use the same principles for your own script if different.

1. At the top of the source file change the file name to run from **'Reflect.exe'** to **'ReflectBin.exe'.**

ReflectBin.exe

```
$strReflectPath = "C:\program files\macrium\reflect\ReflectBin.exe";
```

> **Note:** If the script was generated with Macrium Reflect version **7.1.2899** or later then this step isn't necessary

.

2. In function '**Backup()**', add **-noconsole** to the Reflect parameter list '**$strArgs**' and **-RedirectStandardOutput** with your log file name to the '**Start-Process**' line as shown below:

-noconsole -RedirectStandardOutput

```
#*********************************************************************
**********
#* Func: Backup
#*
#* Desc: Calls Reflect.exe passing an XML BDF as a parameter.
#*
#*********************************************************************
**********
function Backup()
{
    Write-Host ' * Running the backup... ' -NoNewLine;
    $strType = GetBackupTypeParameter;
    $strArgs = "-e -w -noconsole $strType `"$strXmlFilePath`" -g";
    $iResult = (Start-Process -FilePath $strReflectPath -ArgumentList
$strArgs -PassThru -Wait -RedirectStandardOutput
c:\backup_log.txt).ExitCode;
    Write-Host 'Done.';
    switch ($iResult)
    {
        2 { OnXmlValidationError; break; }
        1 { OnBackupError; break; }
        0 { OnBackupSuccess; break; }
    }
    return $iResult;
}
```

The script will now pipe log output to the log file name provided with the **-RedirectStandardOutput** switch**.**