01.03/2022 / cr

Kennen Sie den Befehl 'tape_perf_test' ?

Diesen Befehl gibt es bereits seit mehr als 20 Jahren - allerdings ist er seit der Einführung des Backup-to-Disk (B2D) in der Praxis eigentlich nicht mehr so wichtig.

Der Befehl führt einen ausführlichen Test mit dem ausgewählten Bandlaufwer durch, und zwar mit veränderten Blockgrößen und Datenmustern. **Dies dauert fast eine Stunde lang**. Anschließend stellt er die Ergebnisse in einer Art Tabelle gegenüber. Sie waren vor allem als Entscheidungshilfe zur Auswahl/Einstellung der Blockgrößen gedacht - auf das Datenmuster haben Sie ja ohnehin keinerlei Einfluß.

Da Tapes heute fast nur noch als Clone-Medien verwendet werden, spielt die Geschwindigkeit heutzutage eine eher untergeordnete Rolle. Trotzdem möchte ich Ihnen den Befehl einmal vorstellen.

Achtung - die Tests sind zeitintensiv. Bei meinem älteren LTO3 Bandlaufwerk dauerten Sie ca. 45 min.

Außerdem gibt es einmal mehr ein NetWorker-typisches, 'kosmetisches' Problem - hier sind es überflüssige Fehlermeldungen. Aus diesem Grund sollten Sie die Ausgabe immer in eine Textdatei umleiten, in der Sie später - mit einem leistungsstarken Editor - die überflüssigen Zeilen herausfiltern können.

So wenden Sie den Befehl an:

```
C:\>tape_perf_test -?
usage:  tape_perf_test -f <device name> { -t total test size} {-x max blocksize}
{-n minblocksize}
      defaults:  total test size = 500
                 max blocksize   = 1024 kB
                 min blocksize   = 16 kB

C:\>
C:\>tape_perf_test -f \\.\Tape0
Block size performance test for device: QUANTUM ULTRIUM 3 2182 (\\.\Tape0)
        from 1024 kB to 16 kB with a data size of 500 MB
 >>> results being logged to file QUANTUM-ULTRIUM_3-2182-19-snode19-2-windows-
02-16-2022-1218.log <<<
               run on windows system 19-snode19-2 on 02-16-2022 @ 12:18
               drive identifiers:
                       serial: QP0715AME00020
                       atnn: ATNN:QUANTUM ULTRIUM 3       QP0715AME00020
                       wwnn: WWNN:500E09E0001A28FB
                       wwpn: WWPN:500E09E4001A28FB
                       port: PORT:00000001
max scsi transfer for system is 1024 kBytes
cdi_open failed: cdi_info.status = CDI_DEVICE_FILE_NAME_UNUSABLE (8)
testing 1024kB blocks:
        *** test using random data:
cdi_open failed: cdi_info.status = CDI_DEVICE_FILE_NAME_UNUSABLE (8)

C:\>
C:\>tape_perf_test -f \\.\Tape0 > D:\tape_perf_test.txt

C:\>
```

Und das ist das Ergebnis, nachem Sie die Fehlermeldungen ...

```
win32 read error: ask for ####, read 0 kbytes
```

... herausgefiltert haben:

```
Block size performance test for device: QUANTUM ULTRIUM 3 2182 (\\.\Tape0)
  from 1024 kB to 16 kB with a data size of 500 MB
 >>> results being logged to file   //
  QUANTUM-ULTRIUM_3-2182-19-nwserver-windows-02-10-2022-0101.log <<<
    run on windows system 19-nwserver on 02-10-2022 @ 01:01
    drive identifiers:
      serial: QP0715AME00020
      atnn: ATNN:QUANTUM ULTRIUM 3       QP0715AME00020
      wwnn: WWNN:500E09E0001A28FB
      wwpn: WWPN:500E09E4001A28FB
      port: PORT:00000001
max scsi transfer for system is 1024 kBytes
testing 1024kB blocks:
  *** test using random data:
    write returns 9 sec --> 56888 kB/s
    read returns 7 sec --> 73142 kB/s
  *** test using bigasm-like data:
    write returns 7 sec --> 73142 kB/s
    read returns 8 sec --> 64000 kB/s
  *** test using copies of tape_perf_test.exe in buffer:
    write returns 5 sec --> 102400 kB/s
    read returns 9 sec --> 56888 kB/s
  *** test using 2:1 compressible data:
    write returns 9 sec --> 56888 kB/s
    read returns 7 sec --> 73142 kB/s
  *** test using 3:1 compressible data:
    write returns 6 sec --> 85333 kB/s
    read returns 10 sec --> 51200 kB/s
  *** test using 4:1 compressible data:
    write returns 5 sec --> 102400 kB/s
    read returns 11 sec --> 46545 kB/s
  *** test using buffer full of zeros:
    write returns 4 sec --> 128000 kB/s
    read returns 8 sec --> 64000 kB/s
testing 768kB blocks:
  *** test using random data:
    write returns 9 sec --> 56888 kB/s
    read returns 8 sec --> 64000 kB/s
  *** test using bigasm-like data:
    write returns 7 sec --> 73142 kB/s
    read returns 8 sec --> 64000 kB/s
  *** test using copies of tape_perf_test.exe in buffer:
    write returns 5 sec --> 102400 kB/s
    read returns 9 sec --> 56888 kB/s
  *** test using 2:1 compressible data:
    write returns 6 sec --> 85333 kB/s
    read returns 7 sec --> 73142 kB/s
  *** test using 3:1 compressible data:
    write returns 6 sec --> 85333 kB/s
    read returns 10 sec --> 51200 kB/s
  .....
```

```
  .....
  *** test using 4:1 compressible data:
    write returns 5 sec --> 102400 kB/s
    read returns 11 sec --> 46545 kB/s
  *** test using buffer full of zeros:
    write returns 4 sec --> 128000 kB/s
    read returns 8 sec --> 64000 kB/s
testing 512kB blocks:
  *** test using random data:
    write returns 9 sec --> 56888 kB/s
    read returns 9 sec --> 56888 kB/s
  *** test using bigasm-like data:
    write returns 7 sec --> 73142 kB/s
    read returns 9 sec --> 56888 kB/s
  *** test using copies of tape_perf_test.exe in buffer:
    write returns 5 sec --> 102400 kB/s
    read returns 11 sec --> 46545 kB/s
  *** test using 2:1 compressible data:
    write returns 6 sec --> 85333 kB/s
    read returns 8 sec --> 64000 kB/s
  *** test using 3:1 compressible data:
    write returns 6 sec --> 85333 kB/s
    read returns 11 sec --> 46545 kB/s
  *** test using 4:1 compressible data:
    write returns 6 sec --> 85333 kB/s
    read returns 11 sec --> 46545 kB/s
  *** test using buffer full of zeros:
    write returns 5 sec --> 102400 kB/s
    read returns 9 sec --> 56888 kB/s
testing 384kB blocks:
  *** test using random data:
    write returns 9 sec --> 56888 kB/s
    read returns 10 sec --> 51200 kB/s
  *** test using bigasm-like data:
    write returns 7 sec --> 73142 kB/s
    read returns 10 sec --> 51200 kB/s
  *** test using copies of tape_perf_test.exe in buffer:
    write returns 6 sec --> 85333 kB/s
    read returns 12 sec --> 42666 kB/s
  *** test using 2:1 compressible data:
    write returns 6 sec --> 85333 kB/s
    read returns 9 sec --> 56888 kB/s
  *** test using 3:1 compressible data:
    write returns 6 sec --> 85333 kB/s
    read returns 13 sec --> 39384 kB/s
  *** test using 4:1 compressible data:
    write returns 6 sec --> 85333 kB/s
    read returns 13 sec --> 39384 kB/s
  *** test using buffer full of zeros:
    write returns 5 sec --> 102400 kB/s
    read returns 10 sec --> 51200 kB/s
  .....
```

```
   .....
   testing 256kB blocks:
   *** test using random data:
     write returns 9 sec --> 56888 kB/s
     read returns 12 sec --> 42666 kB/s
   *** test using bigasm-like data:
     write returns 7 sec --> 73142 kB/s
     read returns 12 sec --> 42666 kB/s
   *** test using copies of tape_perf_test.exe in buffer:
     write returns 5 sec --> 102400 kB/s
     read returns 11 sec --> 46545 kB/s
   *** test using 2:1 compressible data:
     write returns 5 sec --> 102400 kB/s
     read returns 13 sec --> 39384 kB/s
   *** test using 3:1 compressible data:
     write returns 4 sec --> 128000 kB/s
     read returns 12 sec --> 42666 kB/s
   *** test using 4:1 compressible data:
     write returns 4 sec --> 128000 kB/s
     read returns 12 sec --> 42666 kB/s
   *** test using buffer full of zeros:
     write returns 3 sec --> 170666 kB/s
     read returns 12 sec --> 42666 kB/s
testing 192kB blocks:
   *** test using random data:
     write returns 9 sec --> 56888 kB/s
     read returns 15 sec --> 34133 kB/s
   *** test using bigasm-like data:
     write returns 7 sec --> 73142 kB/s
     read returns 14 sec --> 36571 kB/s
   *** test using copies of tape_perf_test.exe in buffer:
     write returns 5 sec --> 102400 kB/s
     read returns 13 sec --> 39384 kB/s
   *** test using 2:1 compressible data:
     write returns 6 sec --> 85333 kB/s
     read returns 14 sec --> 36571 kB/s
   *** test using 3:1 compressible data:
     write returns 5 sec --> 102400 kB/s
     read returns 14 sec --> 36571 kB/s
   *** test using 4:1 compressible data:
     write returns 4 sec --> 128000 kB/s
     read returns 14 sec --> 36571 kB/s
   *** test using buffer full of zeros:
     write returns 4 sec --> 128000 kB/s
     read returns 14 sec --> 36571 kB/s
   .....
```

```
  .....
testing 128kB blocks:
   *** test using random data:
     write returns 9 sec --> 56888 kB/s
     read returns 19 sec --> 26947 kB/s
   *** test using bigasm-like data:
     write returns 7 sec --> 73142 kB/s
testing 256kB blocks:
   *** test using random data:
     write returns 9 sec --> 56888 kB/s
     read returns 12 sec --> 42666 kB/s
   *** test using bigasm-like data:
     write returns 7 sec --> 73142 kB/s
     read returns 12 sec --> 42666 kB/s
   *** test using copies of tape_perf_test.exe in buffer:
     write returns 5 sec --> 102400 kB/s
     read returns 11 sec --> 46545 kB/s
   *** test using 2:1 compressible data:
     write returns 5 sec --> 102400 kB/s
     read returns 13 sec --> 39384 kB/s
   *** test using 3:1 compressible data:
     write returns 4 sec --> 128000 kB/s
     read returns 12 sec --> 42666 kB/s
   *** test using 4:1 compressible data:
     write returns 4 sec --> 128000 kB/s
     read returns 12 sec --> 42666 kB/s
   *** test using buffer full of zeros:
     write returns 3 sec --> 170666 kB/s
     read returns 12 sec --> 42666 kB/s
testing 192kB blocks:
   *** test using random data:
     write returns 9 sec --> 56888 kB/s
     read returns 15 sec --> 34133 kB/s
   *** test using bigasm-like data:
     write returns 7 sec --> 73142 kB/s
     read returns 14 sec --> 36571 kB/s
   *** test using copies of tape_perf_test.exe in buffer:
     write returns 5 sec --> 102400 kB/s
     read returns 13 sec --> 39384 kB/s
   *** test using 2:1 compressible data:
     write returns 6 sec --> 85333 kB/s
     read returns 14 sec --> 36571 kB/s
   *** test using 3:1 compressible data:
     write returns 5 sec --> 102400 kB/s
     read returns 14 sec --> 36571 kB/s
   *** test using 4:1 compressible data:
     write returns 4 sec --> 128000 kB/s
     read returns 14 sec --> 36571 kB/s
   *** test using buffer full of zeros:
     write returns 4 sec --> 128000 kB/s
     read returns 14 sec --> 36571 kB/s
     read returns 19 sec --> 26947 kB/s
   *** test using copies of tape_perf_test.exe in buffer:
     write returns 5 sec --> 102400 kB/s
     read returns 18 sec --> 28444 kB/s
  .....
```

```
  .....
  *** test using 2:1 compressible data:
    write returns 6 sec --> 85333 kB/s
    read returns 19 sec --> 26947 kB/s
  *** test using 3:1 compressible data:
    write returns 7 sec --> 73142 kB/s
    read returns 19 sec --> 26947 kB/s
  *** test using 4:1 compressible data:
    write returns 5 sec --> 102400 kB/s
    read returns 22 sec --> 23272 kB/s
  *** test using buffer full of zeros:
    write returns 4 sec --> 128000 kB/s
    read returns 19 sec --> 26947 kB/s
testing 96kB blocks:
  *** test using random data:
    write returns 9 sec --> 56888 kB/s
    read returns 23 sec --> 22260 kB/s
  *** test using bigasm-like data:
    write returns 7 sec --> 73142 kB/s
    read returns 23 sec --> 22260 kB/s
  *** test using copies of tape_perf_test.exe in buffer:
    write returns 6 sec --> 85333 kB/s
    read returns 23 sec --> 22260 kB/s
  *** test using 2:1 compressible data:
    write returns 6 sec --> 85333 kB/s
    read returns 23 sec --> 22260 kB/s
  *** test using 3:1 compressible data:
    write returns 5 sec --> 102400 kB/s
    read returns 26 sec --> 19692 kB/s
  *** test using 4:1 compressible data:
    write returns 6 sec --> 85333 kB/s
    read returns 25 sec --> 20480 kB/s
  *** test using buffer full of zeros:
    write returns 5 sec --> 102400 kB/s
    read returns 24 sec --> 21333 kB/s
testing 64kB blocks:
  *** test using random data:
    write returns 9 sec --> 56888 kB/s
    read returns 32 sec --> 16000 kB/s
  *** test using bigasm-like data:
    write returns 6 sec --> 85333 kB/s
    read returns 32 sec --> 16000 kB/s
  *** test using copies of tape_perf_test.exe in buffer:
    write returns 8 sec --> 64000 kB/s
    read returns 32 sec --> 16000 kB/s
  *** test using 2:1 compressible data:
    write returns 7 sec --> 73142 kB/s
    read returns 34 sec --> 15058 kB/s
  *** test using 3:1 compressible data:
    write returns 6 sec --> 85333 kB/s
    read returns 34 sec --> 15058 kB/s
  *** test using 4:1 compressible data:
    write returns 6 sec --> 85333 kB/s
    read returns 34 sec --> 15058 kB/s
  *** test using buffer full of zeros:
    write returns 5 sec --> 102400 kB/s
    read returns 32 sec --> 16000 kB/s
  .....
```

```
  .....
testing 32kB blocks:
  *** test using random data:
    write returns 10 sec --> 51200 kB/s
    read returns 57 sec --> 8982 kB/s
  *** test using bigasm-like data:
    write returns 12 sec --> 42666 kB/s
    read returns 58 sec --> 8827 kB/s
  *** test using copies of tape_perf_test.exe in buffer:
    write returns 13 sec --> 39384 kB/s
    read returns 58 sec --> 8827 kB/s
  *** test using 2:1 compressible data:
    write returns 12 sec --> 42666 kB/s
    read returns 59 sec --> 8677 kB/s
  *** test using 3:1 compressible data:
    write returns 10 sec --> 51200 kB/s
    read returns 59 sec --> 8677 kB/s
  *** test using 4:1 compressible data:
    write returns 10 sec --> 51200 kB/s
    read returns 59 sec --> 8677 kB/s
  *** test using buffer full of zeros:
    write returns 9 sec --> 56888 kB/s
    read returns 58 sec --> 8827 kB/s
testing 16kB blocks:
  *** test using random data:
    write returns 18 sec --> 28444 kB/s
    read returns 112 sec --> 4571 kB/s
  *** test using bigasm-like data:
    write returns 18 sec --> 28444 kB/s
    read returns 113 sec --> 4530 kB/s
  *** test using copies of tape_perf_test.exe in buffer:
    write returns 18 sec --> 28444 kB/s
    read returns 121 sec --> 4231 kB/s
  *** test using 2:1 compressible data:
    write returns 19 sec --> 26947 kB/s
    read returns 110 sec --> 4654 kB/s
  *** test using 3:1 compressible data:
    write returns 18 sec --> 28444 kB/s
    read returns 113 sec --> 4530 kB/s
  *** test using 4:1 compressible data:
    write returns 17 sec --> 30117 kB/s
    read returns 111 sec --> 4612 kB/s
  *** test using buffer full of zeros:
    write returns 16 sec --> 32000 kB/s
    read returns 114 sec --> 4491 kB/s
  .....
```

Die abschließende Ergebnistabelle sehen Sie auf der nächsten Seite im Querformat ...

```
Results:

xfer    random      bigasm-like   executable        2:1           3:1           4:1           zeros
size    W     R       W     R       W     R       W     R       W     R       W     R       W     R
---- ------------ ------------ ------------ ------------ ------------ ------------ ------------
1024 056888/073142 073142/064000 102400/056888 056888/073142 085333/051200 102400/046545 128000/064000
0768 056888/064000 073142/064000 102400/056888 085333/073142 085333/051200 102400/046545 128000/064000
0512 056888/056888 073142/056888 102400/046545 085333/064000 085333/046545 085333/046545 102400/056888
0384 056888/051200 073142/051200 085333/042666 085333/056888 085333/039384 085333/039384 102400/051200
0256 056888/042666 073142/042666 102400/046545 102400/039384 128000/042666 128000/042666 170666/042666
0192 056888/034133 073142/036571 102400/039384 085333/036571 102400/036571 128000/036571 128000/036571
0128 056888/026947 073142/026947 102400/028444 085333/026947 073142/026947 102400/023272 128000/026947
0096 056888/022260 073142/022260 085333/022260 085333/022260 102400/019692 085333/020480 102400/021333
0064 056888/016000 085333/016000 064000/016000 073142/015058 085333/015058 085333/015058 102400/016000
0032 051200/008982 042666/008827 039384/008827 042666/008677 051200/008677 051200/008677 056888/008827
0016 028444/004571 028444/004530 028444/004231 026947/004654 028444/004530 030117/004612 032000/004491
```

Es bedeuten:

`xfer size`	Blockgröße [KB]
`random`	Zufalls-Datenmuster
`bigasm-like`	Wie bei der NetWorker `bigasm` Directive
`executable`	'Standard'-Programmdatei
`#:1`	Komprimierbare Datenmuster
`zeros`	Lauter '0'en