Preface

To determine the influence of the parameters of bareos tape devices, we did some lab tests using btape and backups with real data.

The tests were conducted on the following test setup generously provided by IBM:

IBM System x3690 X5

CPU : Intel(R) Xeon(R) CPU E7- 2803 @ 1.73GHz stepping 02

RAM: 148 GB

Tape Drive: IBM ULT3580-TD6 D2D2 PQ: 0 ANSI: 6

Connection:

QLogic Fibrechannel controller: QLogic QLE2562 - QLogic 8Gb FC Dual-port HBA for System x.

Different Block Sizes and their speed implications

Bareos can be configured to write different blocksizes on the tape devices.

The parameter to set the blocksize are:

```
Minimum block size =
```

```
Maximum block size =
```

The **default value** for the blocksize is **64512 Bytes.**

The following blocksizes were tested using the **speed** command in **btape**:

- 64512 (default)
- 131072 (128k)
- 262144 (256k)
- 524288 (512k)
- 1048576 (1M)
- 2097152 (2M)
- 4194304 (4M)

During the speed test, btape writes **raw blocks** and **bareos blocks**, both with **zero data** and with **random data**.

For each type of data and block type, btape test writes 3 Tests:

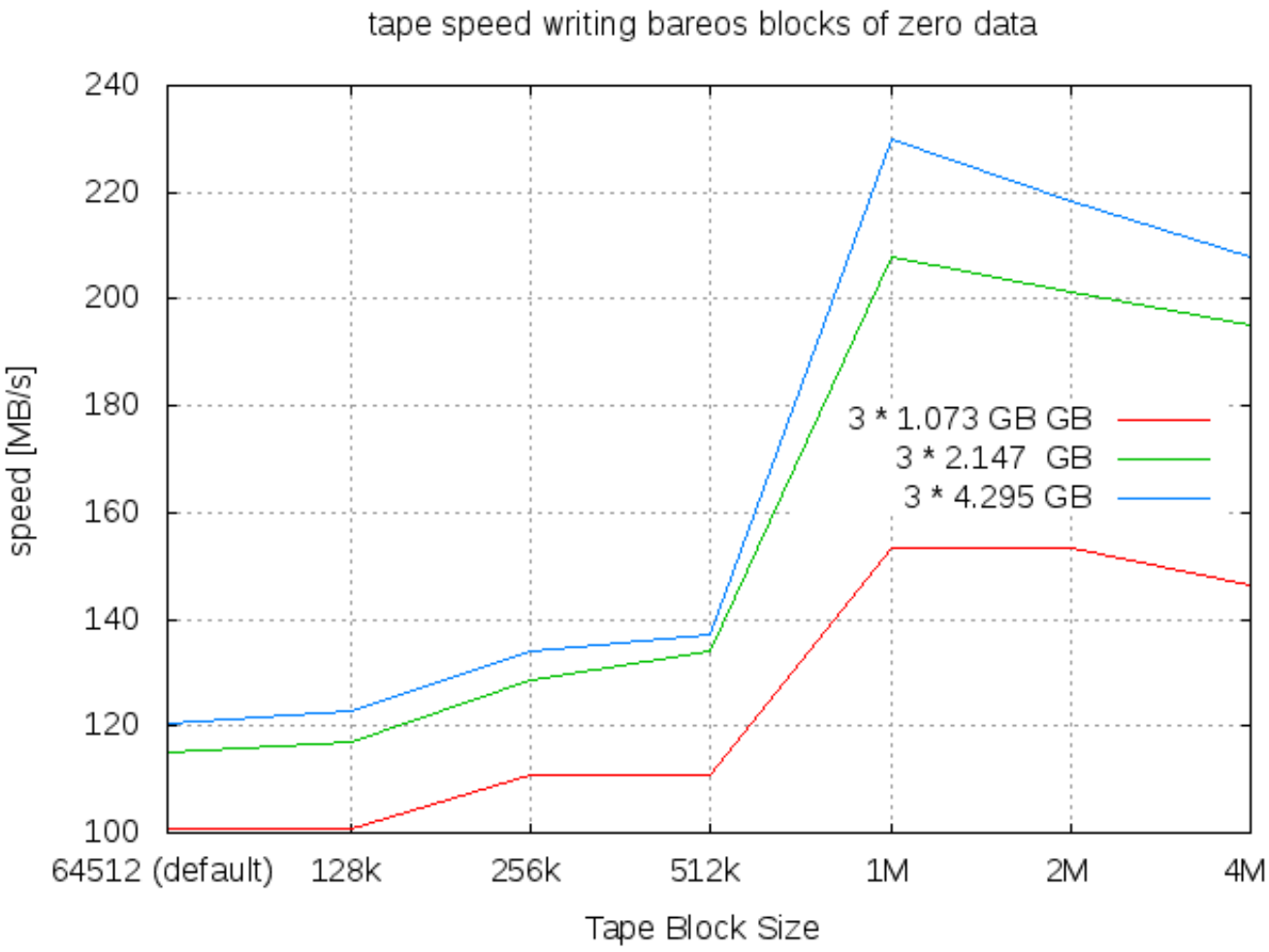
- 3 * 1.074 GBytes
- 3 * 2.147 GBytes
- 3 * 4.294 GBytes

and calculates the averate of the speed.

As we are only interested what speed we can achieve in real environments, we only look at tests writing *bareos blocks*.

Speed results for btape writing Bareos Block with *Zero* Data

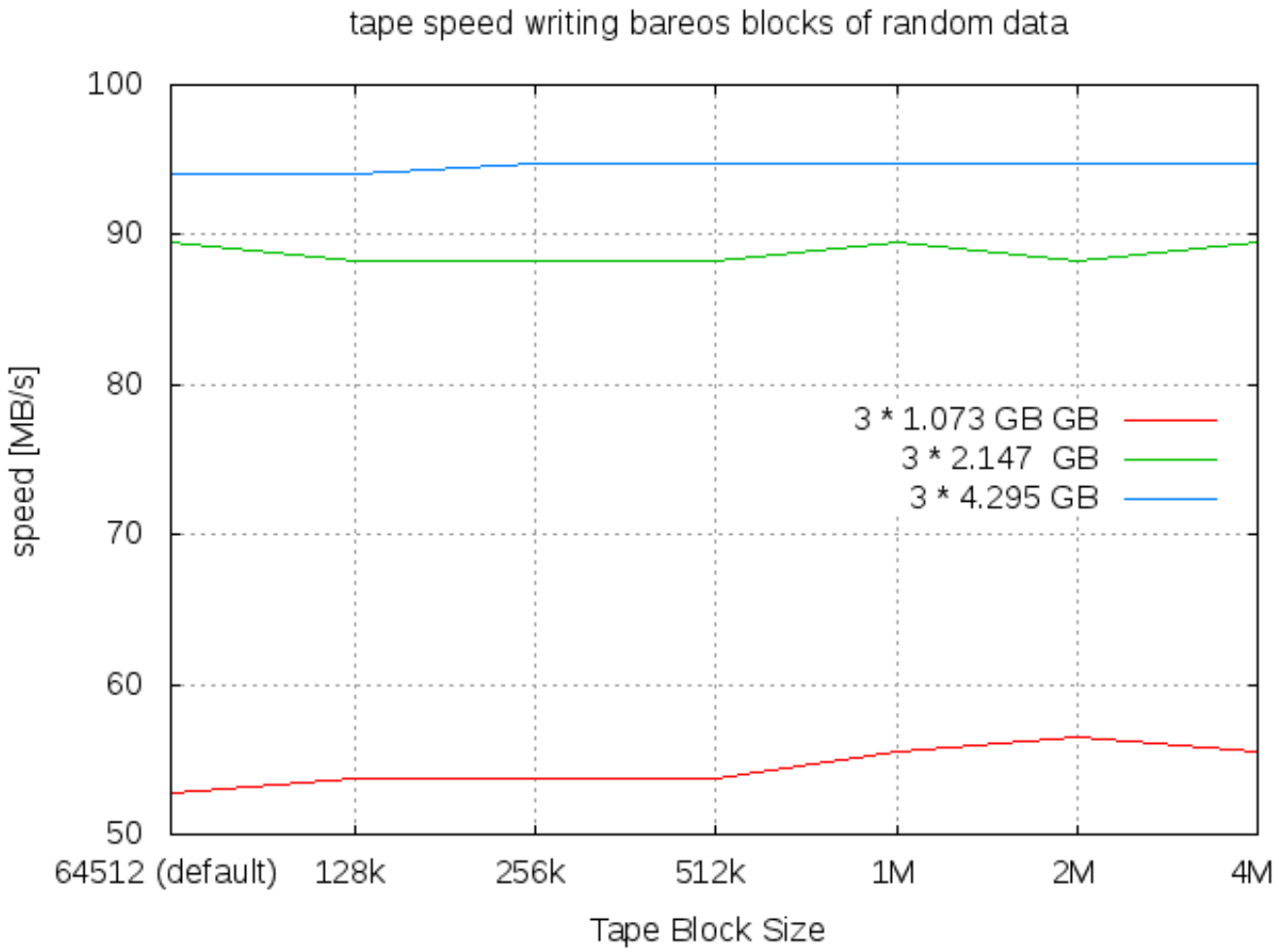
The drives' hardware compression should work best with this kind of data so we expect the maximum write speed with zero data.



As can be seen, our tape device creates the maximum writespeed at a block size of 1M when writing bareos blocks with zero data.

Speed results for btape writing Bareos Block with *Random* Data

The drives' hardware compression should not be able to do much about random data, so we expect the lowest speed with this test data.



When writing bareos block with random data, the block size has almost no impact to the total writespeed.

Different File Sizes and their speed implications

Additionally to the block size, a bareos device can also be configured to use a certain **maximum filesize**.

```
Maximum Filesize =
```

After this amount of data, an EOF file mark is written to the tape by bareos. These filemarks can be directly accessed by the tape drive, so that having filemarks at certain data amounts helps to improve the restore speed.

To see the effect of this parameter, we did a test backup with 45GBytes of backup data including the complete Redhat Linux System and some virtual machine images. We expect that the achievable backup speed lies between the tests with random and with zero data.

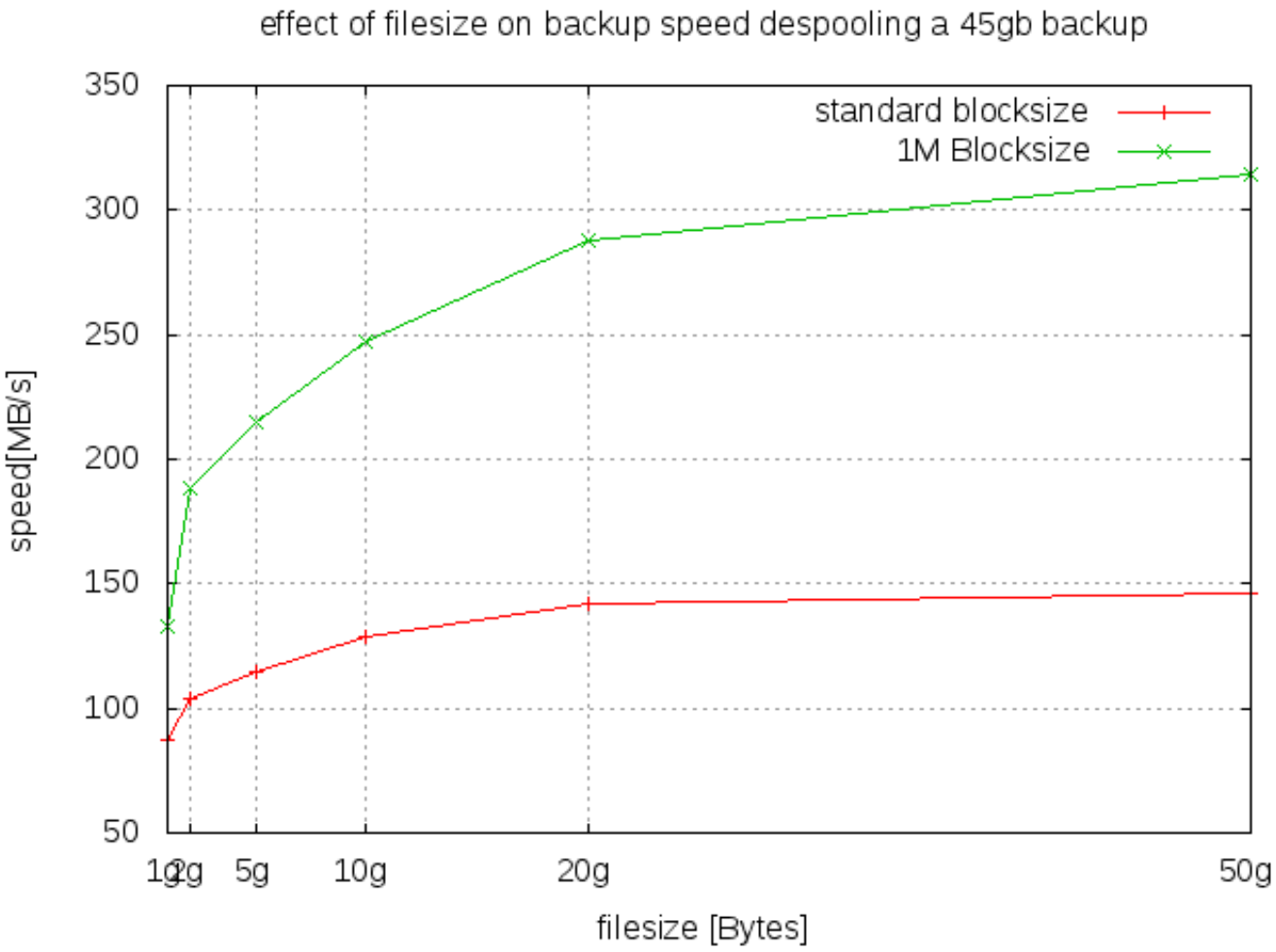
This 45GB Backup was written to tape using spooling, so that the limiting factor for speed really only is the tape itself and not the fd collecting data.

The default value for the *maximum file size* directive is **one gigabyte = 1000000000 bytes.**

The tests were conducted using maximum file sizes of:

- 1g (default)
- 2g
- 5g
- 10g
- 20g
- 50g

The following graph shows the write speed depending on the configured filesize for the standard block size and for 1M blocksize.



As the figure shows, bigger filesizes help very much to improve the backup speed. The default value of maximum file size is much too small for modern tape drives.

conclusion

To achieve the best throughput in modern tape drives, it is absolutely advisable to set the device directives for **block size** and **file size**.

Prior Bareos Version 14.2, it was unfortunately not possible to change the block size in an existing installation, as then the existing tapes will not be readable by bareos with the new blocksize.

So if you wanted to change the block size in an exisiting installation, you had to switch back your device configuration to the old value when you want to restore data.

From Version 14.2 on, it **is possible** to configure the blocksizes in the Pool Ressource. This opens the option to configure different blocksizes for different pools, so that it is poosible to have a smooth migration to a different blocksize without loosing the possibility to easily restore old data. Details can be found in the Bareos documentation.

The **maximum file size** can be changed in an existing system without problems, also in Bareos prior Version 14.2.