УДК 004

SCALABILITY OF GLUSTERFS IN A SMALL CLUSTER FOR PROVIDING FAST ACCESS TO LARGE AMOUNT OF DATA

A.V. Reviakin
Scientific advisor: A.S. Fadeev
National Research Tomsk Polytechnic University
E-mail: presler@tpu.ru

With growing of data volumes, providing of fast access to information becomes a challenge. There are two ways to solve it: scale up (getting more power machines without increasing of its number) and scale out (getting larger number of machines). Scaling up is easier way, but it is often expensive and there is always a physical limit at current point in time when a more powerful hardware cannot be purchased for any price. Scaling out is usually cheaper, however it requires more skills to set up such systems and reach anticipated scale level. This article will be dedicated to one of scale out tools – distributed file system GlusterFS.

Keywords: GlusterFS, file system.

Introduction

GlusterFS is an open-source software, which is developed currently by Red Hat Inc. and community. GlusterFS can aggregate separate storage over Ethernet or InfiniBand into one file storage, providing transparent access for end users. GlusterFS has a client and server.

Servers (which are called bricks) store data using different type of volumes. There are three general types:

- distributed volumes used to spread files randomly across available bricks in the volume. For example, the first file can be stored in the first brick, the second file in the second one. There is no redundancy, because the main purpose of such volume type is easy scaling of volume size;
- replicated volumes are used for providing redundancy and reliability. Each brick has the same copy of data. So, if one brick is failed, the data will be still protected and accessible;
- striped volumes divide individual files into small chunks between available bricks, due to that the load will be distributed among servers and files can be fetched faster than when distributed volumes are used. Redundancy is not provided because of file split.

There is a possibility to create hybrid volume, such as striped-replicated, distributed-replicated, distributed-striped-replicated to combine advantages of separate volume types [1].

Testing

For experiments there were used Linux OpenSUSE 13.1 and GlusterFS version 3.6.2 (the last version at this time). For hardware there were used 5 servers with 16 GB of RAM in each, HDD disks and the same configuration in the client. InfiniBand QDR was used to connect computers, which provide effective speed up to 32.89 Gbit/s.

The task was to provide fast access to large amount of already saved data. So, stripe type of volume was chosen.

There are two available types of transport: tcp and RDMA [2]. Tcp was chosen, because current version of GlusterFS has some issues when RDMA is used in combination with stripe volume.

Two files were created for tests: the first one is 13 GB (13,958,643,712 bytes) and the second one is 65 GB (69,793,218,560 bytes). Such size of file (13 GB) was chosen to fill RAM of client fully (3 GB for OS and test tools).

For measurement there were made 10 iterations of reading for 13 GB file and 5 iterations for 65 GB file.

For calculating of average time, reading results for 13 GB file of the first iteration were excluding to prevent influence of cache. For 65 GB file, all five iterations were used because it is too big for available size of RAM.

Table 1

Test results for 13 GB file. Reading speed for first iteration and average of reading speed for 2–10 iterations

	1 brick, MB/s	2 bricks, MB/s	3 bricks, MB/s	4 bricks, MB/s	5 bricks, MB/s
1 iteration	127.3	255.66	371.64	498.58	545.13
2–10 iterations	128.87	1091.15	1000.9	1091.1	1008.48

Table 2

Test results for 65 GB file. Average reading speed

	Time of reading 65 GB file, MB/s
1 brick	125.88
2 bricks	250.4 (1.99x faster than 1 brick)
3 bricks	361.48 (2.87x faster than 1 brick)
4 bricks	459.73 (3.65x faster than 1 brick)
5 bricks	545.13 (4.33x faster than 1 brick)

According to results in table 1, the average speed of access to 13 GB file is from 1000 to 1091 MB/s (excluding results test with one brick). However, judging by the much smaller reading speed of the first iteration for 13 GB file in table 1, we can make a conclusion that Linux uses cache to provide fast access to just recently read file.

Results of reading speed 65 GB file can help to exclude influence of caching because it much bigger than RAM size. In the table 2, we can see increase in reading speed with every next added brick, but each next gains less speed than previous one.

Conclusion and future work

To conclude the results, GlusterFS can scale-out cluster efficiently, providing fast access to big files. In tests of reading, it showed 4.33 times advantage in reading speed with 5 servers in comparison to 1.

In future version, when RDMA transport with stripe volumes will have stable work and will get performance optimizations [3], it will be very perspective to use it. RDMA supports zero-copy networking, which allows the network adapter to transfer data directly to or from application memory, eliminating the need of copying data between application memory and the data buffers of the operating system. Such type of data transferring requires no CPU's load, caches, context switches. When a program use RDMA delivering data directly to the network, it reduces latency, give boost in speed and unload a CPU.

References

1. Marcin Stangel. GlusterFS – introduction, 15.10.2014. http://www.rackspace.com/knowledge_center/article/glusterfs-introduction.
2. Anand Babu. Infiniband, 10GigE and GlusterFS, 18.11.2009. http://unlocksmith.org/2009/11/infiniband-10gige-and-glusterfs.html.
3. Performance enhancement for RDMA, 2015. https://bugzilla.redhat.com/show_bug.cgi?id=1187456.