No. 189
April 2015

Bits & Bytes

RECHENZENTRUM GARCHING

Garching Computing Center of the Max Planck Society*
Gießenbachstraße 2, D-85748 Garching bei München

Sync & Share

Raphael Ritz, Thomas Zastrow, and Florian Kaiser

Why a new service?

With the increasing prevalence of mobile and handheld devices cloud-based sync&share solutions are getting more and more popular. People appreciate the convenience of services like Dropbox and expect similar offerings in their work environment. To this end the RZG has set up an in-house private storage cloud using the ownCloud software stack.

Who can use RZG's new DataShare service?

All users with a regular user account at the RZG can use this service. In addition, guest user accounts can be created to enable collaboration and sharing with colleagues who do not have an account at the RZG.

How to enable it

Before you can use RZG's DataShare Service you have to indicate that you want to use it. To do so you log in at `https://subscriptions.rzg.mpg.de` and follow the instructions there. Your subscription will be effective immediately. This step is necessary, because the RZG has to pay a fee for everyone who is using this service. At the same site you can also manage your guest user invitations (more on that below). You can now log in at `https://datashare.rzg.mpg.de` using your RZG credentials (username and password).

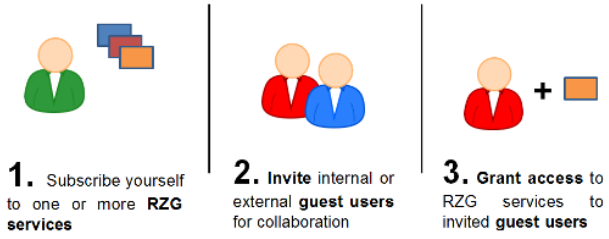

Fig. 1: Functions of the subscription service

By default you are allowed to store up to 20 GB in the RZG storage cloud, but if you need more storage space, just drop a note to datashare@rzg.mpg.de and tell us how much more you need – and why (yes, we are interested to learn what the services are needed for).

Synchronization

Once enabled you can upload data (files or entire directories) to the service and then access this data from all your devices – PC, notebook, tablet, phone – through a web browser or client applications. While no additional software is needed to use the service through a web browser, dedicated applications are available to improve the user experience on desktops and mobile devices. These clients will also make sure that changes that you make on your data using one device will be reflected on the other devices as well, i. e., data are *synchronized* across your devices. There are generic client programs available from `https://doc.owncloud.org` as well as preconfigured clients specifically made for using RZG's DataShare service. You find pointers to those at `https://www.rzg.mpg.de/services/data/share/clients`.

Fig. 2: Data are synchronized across multiple devices.

*Tel.: +49(89) 3299-01, e-mail: benutzerberatung@rzg.mpg.de, URL: http://www.rzg.mpg.de/
Editor: Renate Dohmen

Alternatively, you can integrate a DataShare folder in your local compute environment using WebDAV. Detailed information on how to connect from various devices and operating systems is available from the ownCloud website.

Sharing

Data that is managed by RZG's DataShare service can easily be made available (aka *shared*) to others. Using the web interface there are two options offered to you as soon as you click on the *share* option of a file or directory:

Share by link: Ticking the *share link* option generates a URL that you can communicate to others (e. g. via e-mail). These people can now in turn access (read or download) the respective file or directory through this URL. If you use the share link option, you can choose to password protect the link and to have the link expire automatically after a certain time. This option is great for quickly sharing (not collaborating on) a file or directory with a group of colleagues. It is not possible to grant someone modification rights in this way. For that you have to use the other option stated subsequently.

Share by invitation: If you invite colleagues to share data, you can give them specific rights, i. e., you can determine whether they are allowed to (i) just read or (ii) also edit the file, and (iii) share the file further. This option is ideal for collaboration with your colleagues. To invite a colleague simply start typing his or her name into the textfield offered by clicking on the *share* option, it will try to autocomplete. Only current users of the DataShare service will be available to choose from. That means if your colleague does not show up in the autocomplete list he or she must first be added to the user base. You can do so yourself using RZG's subscription service again. Note that if you invite a colleague who does not have an RZG account you will be responsible for the guest user. You can revoke an invitation any time using RZG's subscription service once more.

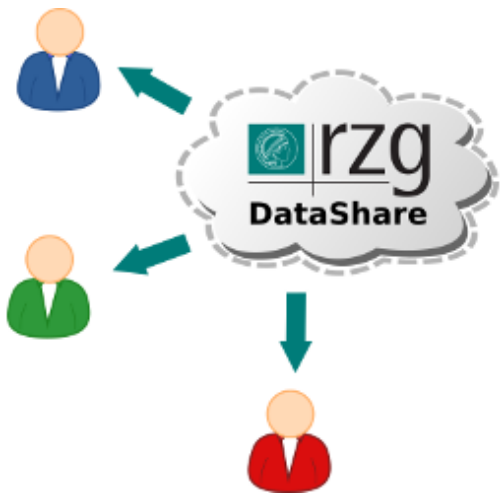

Fig. 3: Data can be shared with colleagues.

There are some applications that you can configure to use this sharing feature for certain purposes. The e-mail client Thunderbird for instance can be instructed using the WebDAV for FileLink plugin to put large e-mail attachments into your DataShare folder rather than adding it to the e-mail. Only the link is embedded into the e-mail then. The recipient can now use the link to access the file.

What this is Not

If you read the ownCloud documentation or if you know other ownCloud installations, you may have noticed that ownCloud also supports shared calendars and address books. The RZG has not enabled these features for two reasons: (i) we consider such functionality in the realm of groupware tools that provide a better and more seamless integration with other applications such as e-mail and (ii) our internal testing has revealed difficulties to use these features under certain circumstances and we cannot provide proper support for such cases. Taking this together we have decided to not offer these features. Similar arguments apply to cross-server sharing. While we do see potential in all those additional features, we will only start to offer them once they are mature and stable enough.

Due to limitations imposed by the technology used under the hood (http-based data transfer) this is not the right tool to use if you want to share vast amounts of data – hundreds of gigabyte and more. There are other means to accomplish this like gridFTP or Globus.

Conclusion

We hope that you will be able to make good use of this new service. Don't hesitate to ask if you run into any issues or if something is not clear or not working as expected. You can best reach us via RZG's help desk or via e-mail to datashare@rzg.mpg.de.

Quick links

- Introduction and documentation: `http://www.rzg.mpg.de/services/data/share`
- Managing subscriptions and invitations: `https://subscriptions.rzg.mpg.de`
- Web interface of the DataShare service: `https://datashare.rzg.mpg.de`
- ownCloud documentation and native clients: `https://doc.owncloud.org`

HPC system Hydra

Fabio Baruffa, Florian Merz (Lenovo), Werner Nagel, Andreas Schott, and Inge Weidl

Application I/O profiling

A new tool is provided by the RZG which can be used to monitor the I/O characteristics of your application on *hydra*. A GPFS utility demon (mmpmon) keeps track of all the I/O that is happening on every node in the system. It logs the aggregated file open and close operations, number of blocks read and written, and the amount of data read and written once every minute. To make this data accessible to the Hydra users, we have implemented a script that collects the data from all nodes of a LoadLeveler job, prints the overall I/O statistics for the job and in addition writes the time-resolved I/O data into a file for further analysis.

To collect the I/O data for your job, add a call to the *collect_io_data.py* script (available via the rzgtools module) at the very end of the LoadLeveler script (i. e. after the application):

```
module load rzgtools
collect_io_data.py -j $LOADL_STEP_ID
```

This writes the overall I/O statistics to the LoadLeveler output file and creates an ASCII file named *io_stats.dat* in the same directory where the job has been run. You can also run *collect_io_data.py* with your LoadLeveler job ID on the login nodes while your job is still running.

To visualize the time resolved I/O statistics in the *io_stats.dat* file, we provide a gnuplot based script in the rzgtools module (gnuplot must be able to open a new window).

```
module load rzgtools
visualize_io_stats
```

For illustration, we report here an output plot from the visualize tool. This has been produced monitoring a synthetic I/O benchmark code. The code writes several times on a file using MPI-IO standard functions. The application runs with 20 tasks on a single node. This is an artificial best case, where the application is doing only I/O.

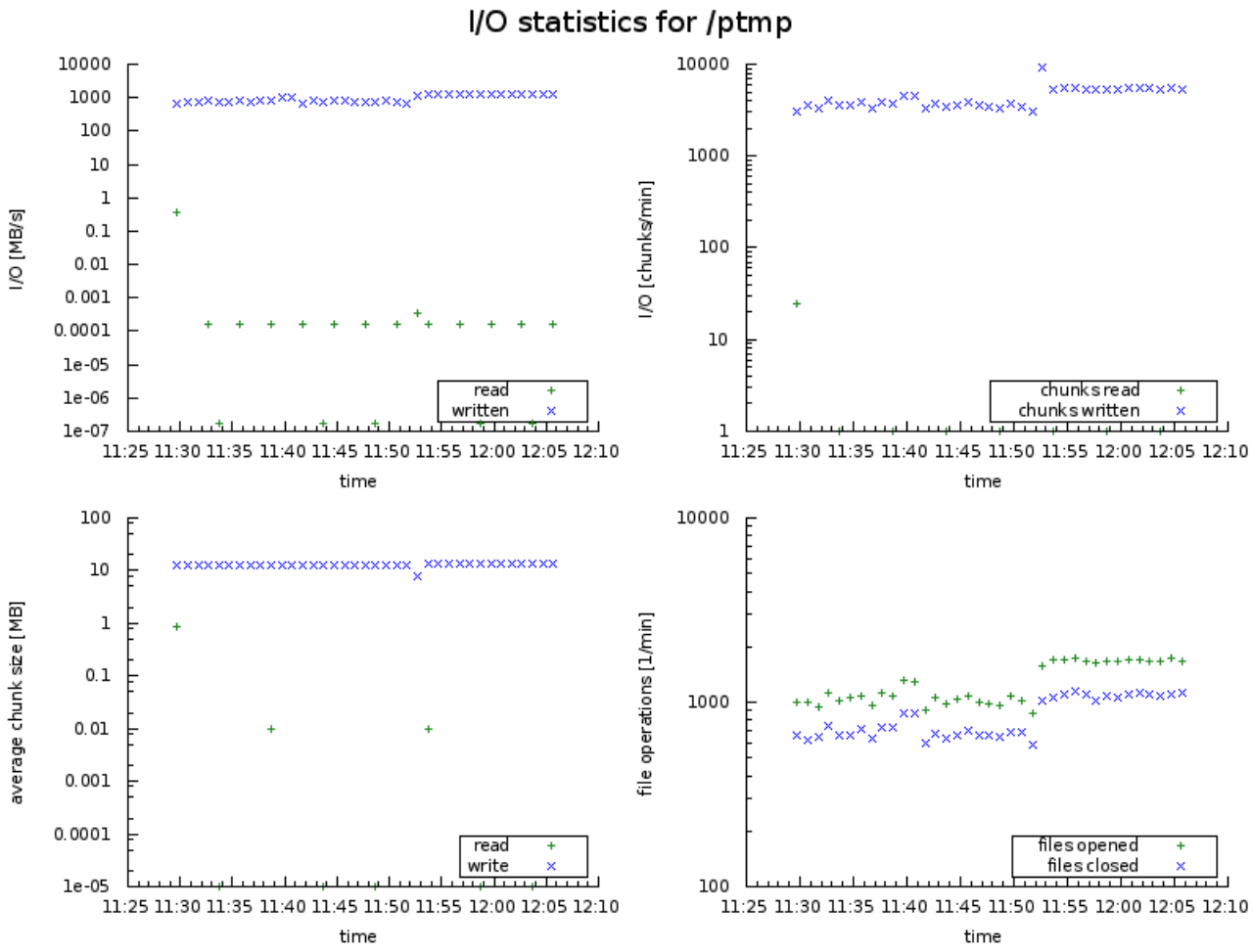

Fig. 4: I/O statistics for the /ptmp filesystem.

The first page that pops up shows the I/O statistics of your job w. r. t. any subdirectory of /u, the second page (press enter on the shell window to navigate) shows the corresponding statistics of your job w. r. t. the /ptmp file system. Each page comprises four time-resolved subplots. The first subplot on each page is the I/O throughput measured in MB/s. The I/O performance for our test case looks quite good, since the maximum value is around 1000 MB/s, which is about 20 % of the theoretical peak. Furthermore one can also distinguish the amount of data written and read. Another important parameter is the number of data chunks written per minute (second subplot). This allows to understand how the I/O is done in the application and how the data is fragmented, which is sometimes not obvious when parallel I/O or libraries like HDF5 are used. The average chunk size is also derived from the data (third subplot). And finally, the number of files opened and closed by the application is monitored (keep in mind that for parallel file I/O each MPI task has to open/close the file, so that this number can get large for large jobs).

The user has to be aware that the monitored I/O traffic for /u contains the application's I/O plus other traffic not directly related to the job, like the access to dynamically linked libraries and system related I/O. The monitored data for /ptmp does not contain such additional traffic. To get a clean profiling of the I/O traffic of your application it is therefore recommended to direct all application I/O to /ptmp. Since the maximum I/O bandwidth to /ptmp is higher than the bandwidth to /u, large and performance critical I/O should always be written to /ptmp anyway.

If your application spends a significant fraction of the runtime in I/O operations, it can be beneficial to optimize the I/O pattern. The most common cause for poor I/O performance is a very fragmented I/O pattern, i. e. the I/O is written in a large number of small chunks instead of fewer big chunks. Fragmented I/O can result from unoptimized I/O subroutines in the user code, but can also be a result of the mapping between the data layout in a shared file and the local data layout in the memory for parallel I/O like MPI-IO or HDF5. To get close to the peak I/O performance, the chunk size should be in the range of the GPFS block size of 4 MB. Another pattern that can cost performance is excessive file open/close operations.

Note that since the I/O is monitored on a node basis, this tool does not work for queues that share nodes like 'tiny' or the 'develop' queues.

Sharing files in GPFS

The GPFS file system offers the possibility of sharing files and directories with other users, even if these users belong to different groups. Not everybody seems to be aware of this, so we offer here a quick 'how to' to our users (cf. also the respective RZG webpage).

Although you can use the native commands `mmgetacl` and `mmputacl`, which allow somewhat finer control to manipulate access control lists (ACLs), the `setfacl` and `getfacl` commands may help you avoid some pitfalls. To make a whole subtree readable for the user 'john', for example, you can issue the following two commands:

```
setfacl -R -m user:john:rx /u/my/own/directory
setfacl -m user:john:rx /u/my/own /u/my
```

The first command applies recursively (`-R`) to every file and directory below /u/my/own/directory. The second command augments the access rights for the upper level directories, which by default are only accessible to the owner. The 'x' (execute) bit is necessary in order to traverse directories, for plain files the 'r' (read) bit is sufficient. You can also define access rights for individual files. When you list such files and directories with `ls -l`, they will have a '+' sign appended to their mode bits.

The action can be reversed with the command

```
setfacl -R -x user:john /u/my/own/directory
```

which you can verify with the comman `getfacl -R /u/my/own/directory`. A more radical measure is

```
setfacl -b /u/my/own/directory
```

which removes all additional ACL entries and leaves access control to the standard UNIX mode bits.

With the `setfacl` command you can, of course, give access also to entire groups, but you do not have control about who belongs to which groups, which are normally defined by the system administrator.

Exhaustive information about GPFS in general can be found in a pdf document on the IBM public library website.

How to synchronize file trees with hydra

Machines with IP addresses belonging to the MPG network are permitted access to *hydra* and can transfer files directly into and out of *hydra* with rsync based on ssh. For example to transfer data from *hydra* into a local directory, the following command can be used:

```
mpg-host> rsync -a user@hydra.rzg.mpg.de:hydradir localdir
```

which is usually equivalent to the command:

```
mpg-host> rsync -a -e 'ssh user@hydra.rzg.mpg.de' :hydradir localdir
```

Best is to use absolute path names for hydradir and localdir. The colon identifies the remote path, which can be accessed using the command specified behind the option '-e'.

If your local computer is not in the MPG network, you have to use the gateway computer of the RZG to access the data. Here it is required that you have password-free access from the gateway to *hydra*, which should be the default. Due to current restrictions in the Kerberos setup you cannot use the name *hydra*, but have to decide for either *hydra01i* or *hydra02i*. Then the following command should synchronize your data from *hydra* onto your local computer:

```
any-host> rsync -a -e 'ssh user@gate.rzg.mpg.de ssh hydra02i' :hydradir localdir
```

Please note, that none of the profiles on the gateway or *hydra* may output anything if not connected to a tty.

You may also use the environment variable RSYNC_RSH to set the command used by rsync, thus shortening the rsync command:

```
any-host> export RSYNC_RSH='ssh user@gate.rzg.mpg.de ssh hydra02i'    # bash syntax
any-host> rsync -a :hydradir localdir
```

In order to check the correct operation, the following command should ask for your password only on the gateway machine and output 'hydra02':

```
any-host>ssh user@gate.rzg.mpg.de ssh hydra02i uname -n
```

Finally some hints on the rsync syntax:

```
rsync -a :/a /b
rsync -a :/a/ /b
```

The first one will result in /b/a on the local computer, while in the second case all entries below /a will be copied below /b.

General News

Christof Hanke

Retirement of Kerberos 4

The Kerberos 4 protocol is no longer supported by major implementations for some years now. In order to keep up-to-date with software updates, we have to disable this protocol on our servers.

Since we support very heterogeneous environments in different institutes, we advised the IT admins of the relevant groups how to deal with this. Furthermore, we already disabled the Kerberos 4 protocol for some institutes, so they can see whether everything works fine. In case of the IPP at Garching adequate Windows packets have been prepared and were rolled out onto some 500 clients with different Windows versions by the RZG.

You can find instructions how to install a Kerberos 5 version of the AFS client on your personal Windows machine at the RZG homepage. On Unix (Linux, Solaris, AIX), just update the AFS client packages to the latest version. The most prominent change is that the command `klog` will not work anymore. Please use

```
kinit   # to get a Kerberos5 ticket
aklog   # to create an AFS token out of it
```

for getting an AFS token using Kerberos 5. See also the FAQ section of our AFS troubleshooting web page.

If you are unsure how to configure your machine correctly for Kerberos 5, please look at the respective RZG web

page. In case of any questions, please contact at first instance your local IT admin and only after this ask at the RZG helpdesk.

Helpdesk

A new version of the helpdesk with extended functionality has been installed. One improvement is that news like current issues or configuration changes (as e. g. the retirement of Kerberos 4) are posted on the front page of the helpdesk. So, if you notice that something is not working as expected with the RZG services, you can now check https://helpdesk.rzg.mpg.de about news from the RZG. In other words, looking at the helpdesk might be useful before creating a ticket.

Events

Hermann Lederer and Markus Rampp

Continuing the series of annual GPU programming workshops, the RZG in collaboration with NVidia holds a one-day workshop on GPU programming on May 7th, 2015 in Garching. Scientists from the Max Planck Society and partner institutions are invited to register at http://www.rzg.mpg.de/cgi-bin/register/registerGPU.pl.

The focus of this year's workshop is on high-level programming paradigms (OpenACC) and on in-situ visualization (i. e. doing visualization on the fly during execution of the program). The program comprises lectures by NVidia on their GPU technology roadmap, on OpenACC programming, and on the basics of 'in-situ' visualization. Participants can take the opportunity to discuss their actual code samples with NVidia's and RZG's application specialists and with colleagues. Application developers from the MPG are cordially invited to present their experiences with GPUs in a short talk (approx. 15 min). Please send an informal proposal for a talk to the organizers. Contact: gpu-workshop@rzg.mpg.de.