How to share your files on Katana and Gadi

John Zaitseff
Research Technology Services
September 2021

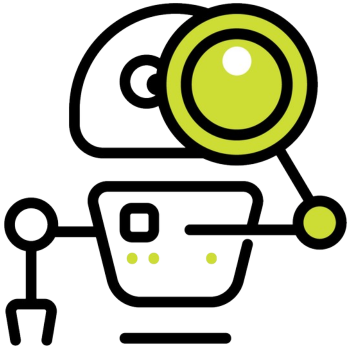
Research Technology Services

1. Research Computing

- High Performance Computing: Gadi, Katana, …
- Cloud computing: Amazon AWS, Microsoft Azure, Google
- Code and algorithm support

2. Research Data

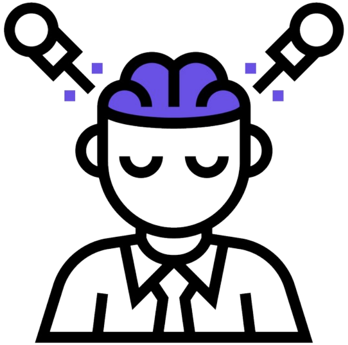
- Data management including highly sensitive or complex data
- Assistance with data moves, storage, planning, tools
- Pilot scheme for publishing Open Data
- UNSW GitHub private, public and limited-sharing repositories



Research Technology Services

3. Research Community

- Over 50 *free* training courses: Linux, Python, Matlab, R, …
- Weekly **Hacky Hour** meetings: via Microsoft Teams, on Thursdays at 3pm. Bring your problems with code, HPC, data and more!
- ResTech seminars, lunch-and-learn series, training videos, …

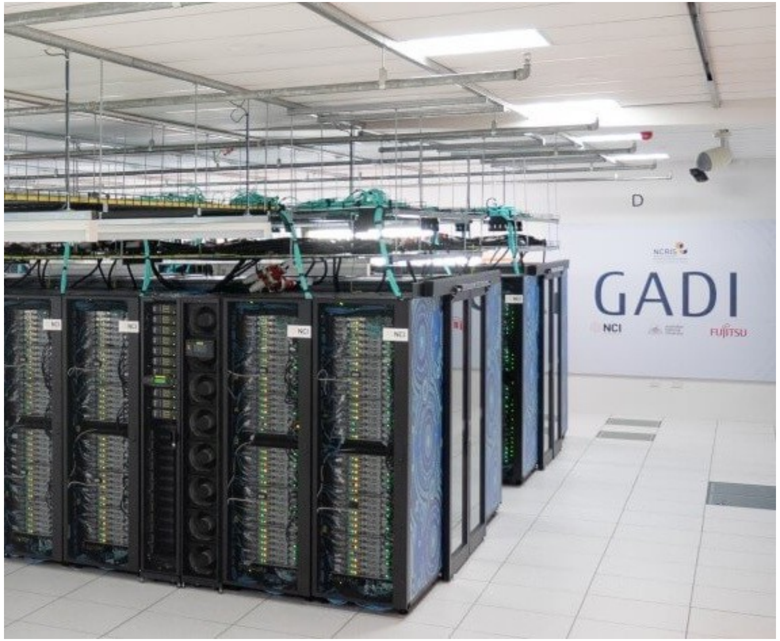


4. ResBaz

- Annual data and compute literacy festival/conference for researchers from all over New South Wales
- Online for 23rd–25th November 2021—see the ResBaz Sydney 2021 website

Sharing your files on Katana and Gadi

- Assumed knowledge
- Users and groups
- Directory listings
- File permissions
- Directory permissions
- Changing group ownership
- Changing file and directory permissions
- Finding file and directory permissions
- Sharing outside HPC systems
- Questions?



Part of the Gadi cluster in Canberra, ACT
Image credit: National Computational Infrastructure

Assumed knowledge

- You have an account on a High Performance Computing system
  - Katana at UNSW
  - Gadi at NCI
- Your colleagues have an account *on the same HPC system*
- You and your colleagues know how to log in to that system via SSH (Secure Shell)
  - See the Katana documentation or NCI Help pages for details
- You and your colleagues know basic Linux commands
  - See the Introduction to Linux and High Performance Computing course notes and associated recorded video
- You are not afraid to try doing things yourself!

Users and groups

- Every user on a Unix-based system has a **username** which is a string (upper and lower-case letters, numbers, underscores, hyphens)
- That username has an associated **user identifier** (UID) which is an integer number
- Every user also belongs to one or more **groups**, each of which are strings
- Each group has an associated **group identifier** (GID), an integer number
- A user's **primary group** is the main group used when creating files and directories and when running processes. All other groups are **secondary groups**
- You can use `whoami`, `groups` and `id` to display your user and group information
- You can use `$USER`, `$UID`, `$GROUPS` and/or `${GROUPS[@]}`, in your job scripts
- For advanced users: effective UID and GIDs vs real UID and GIDs

Users and groups

- The `id` command gives full details of your user and groups

```
$ id
uid=19693022(z9693022) gid=5000(unsw) groups=5000(unsw),
30029(spree),30074(cstelec),40066(MECH) context=unconfined_u:
unconfined_r:unconfined_t:s0-s0:c0.c1023
```

- The `uid=19693022(z9693022)` shows my UID is 19693022 and username is **z9693022**. On Gadi, it is `uid=8832(jjz561)` (UID 8832, username **jjz561**)
- The `gid=5000(unsw)` shows my primary group is **unsw** (GID 5000)
- The `groups=` shows all my groups, primary first, followed by secondary groups
- For advanced users: `context=` shows the SELinux context my account uses
- Use `sg` *`GROUP COMMAND`* to set *GROUP* as the primary group for the duration of running a single command *COMMAND*

Users and groups

- To get a list of users in a group, use `getent group GROUP_OR_GID`

```
$ getent group spree
spree:x:30029:z9693022,z3313725,z5010232,z3505796,z5168588, …
```

- Four fields separated by "`:`": group name, group password (ignored), GID, and list of usernames in that group (separated by commas)
- Does *not* work for primary groups!
- To show information about a user, use `getent passwd USERNAME_OR_UID`

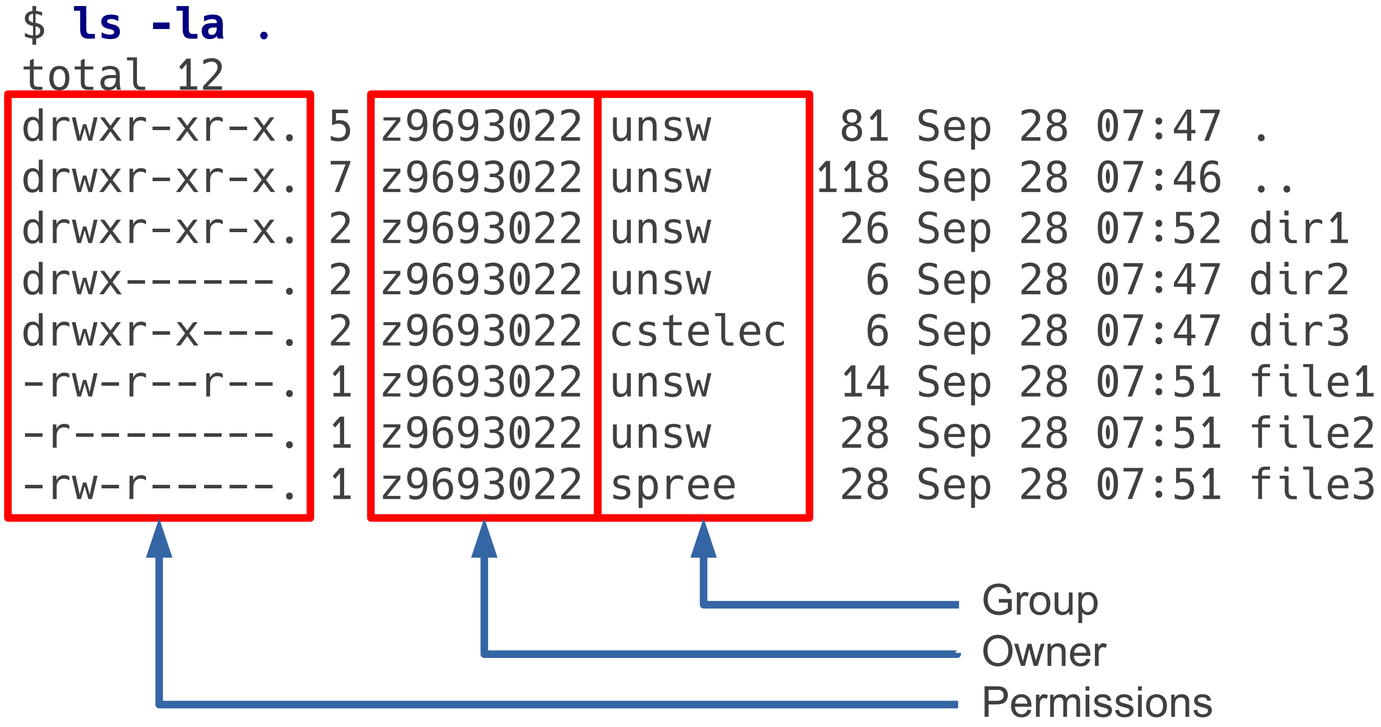
```
$ getent passwd z9693022
z9693022:*:19693022:5000:John Zaitseff:/home/z9693022:/bin/bash
```

- Seven fields separated by "`:`": username, password (not stored here!), UID, primary GID, information about the user ("GECOS field"), home directory, and login shell

Directory listings

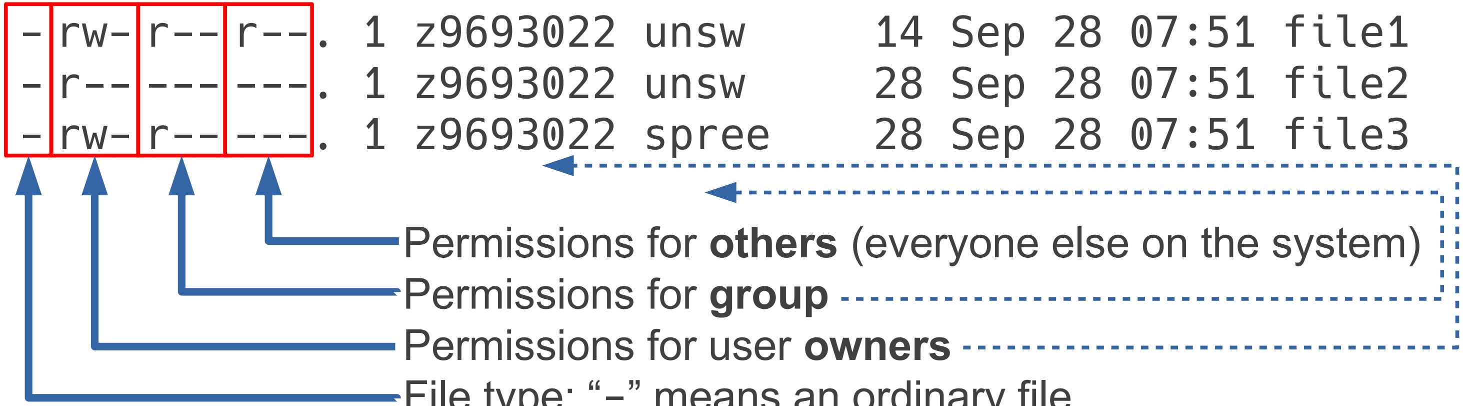
- Every file and directory has one **owner** and one **group** to which the file belongs
- Each file and directory has **permissions** which determine who can access that file or directory and what they can do with it
- You can list the owners and permissions of files and directories by using `ls` with the `-al` options (minus sign, lowercase A, lowercase L)
- Seven fields are displayed:
  - File type and permissions (eg, "`-rw-r--r--`", "`-rw-------`", "`drwxr-xr-x`")
  - Number of links (usually "1" for ordinary files)
  - File owner
  - Group
  - File size in bytes
  - Time of last modification to the file ("change time" or "ctime")
  - Filename

Directory listings



```
$ ls -la .
total 12
drwxr-xr-x. 5 z9693022 unsw     81 Sep 28 07:47 .
drwxr-xr-x. 7 z9693022 unsw    118 Sep 28 07:46 ..
drwxr-xr-x. 2 z9693022 unsw     26 Sep 28 07:52 dir1
drwx------. 2 z9693022 unsw      6 Sep 28 07:47 dir2
drwxr-x---. 2 z9693022 cstelec   6 Sep 28 07:47 dir3
-rw-r--r--. 1 z9693022 unsw     14 Sep 28 07:51 file1
-r--------. 1 z9693022 unsw     28 Sep 28 07:51 file2
-rw-r-----. 1 z9693022 spree    28 Sep 28 07:51 file3
```

File permissions



```
-rw-r--r--. 1 z9693022 unsw     14 Sep 28 07:51 file1
-r--------. 1 z9693022 unsw     28 Sep 28 07:51 file2
-rw-r-----. 1 z9693022 spree    28 Sep 28 07:51 file3
```

Permissions for **others** (everyone else on the system)

Permissions for **group**

Permissions for user **owners**

File type: “-” means an ordinary file

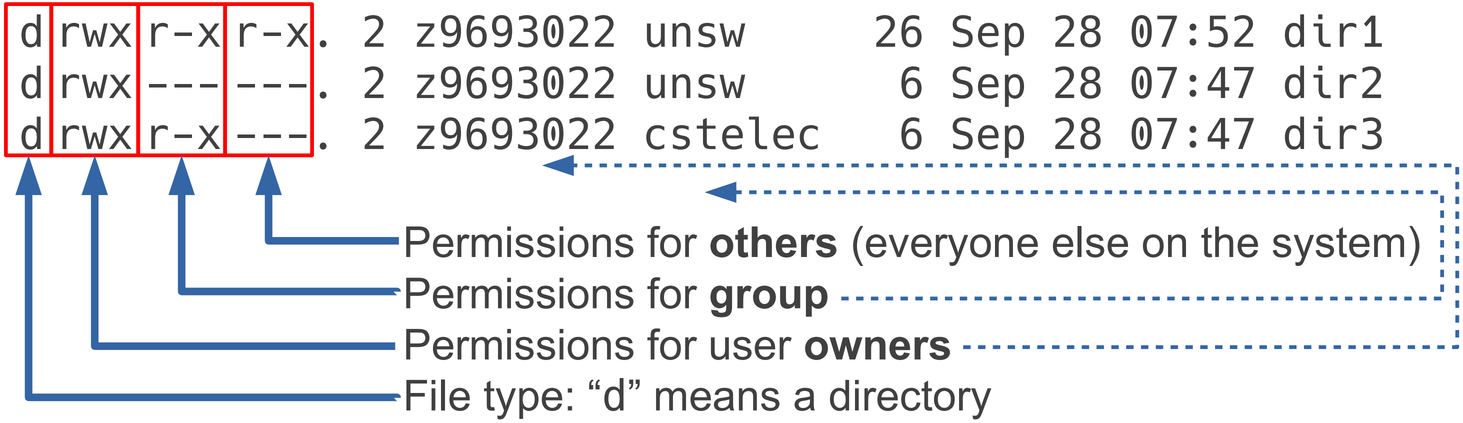
- File permissions are **read**, **write** and **execute** (“r”, “w”, “x”)
  - Read: the owner, group or anyone else can *read* the contents of this file
  - Write: the owner, group or anyone else can *write* data into this file
  - Execute: the owner, group or anyone else can *execute* (run) this file as a program
- A hyphen “-” indicates that permission (read, write, execute) is *not* given
- Owner permissions override group permissions; group’s override others’ permissions!

File permissions

```
-rw-r--r--. 1 z9693022 unsw     14 Sep 28 07:51 file1
-r--------. 1 z9693022 unsw     28 Sep 28 07:51 file2
-rw-r-----. 1 z9693022 spree    28 Sep 28 07:51 file3
```

- **`file1`** permissions (“publically readable file”):
  - The owner `z9693022` can **r**ead and **w**rite this file, but not e**x**ecute it (“`rw-`”)
  - Anyone in the group `unsw` can **r**ead this file, but not **w**rite to it or e**x**ecute it (“`r--`”)
  - Everyone else on the system can **r**ead this file, but not **w**rite to it or e**x**ecute it (“`r--`”)
- **`file2`** permissions (“private read-only file”):
  - The owner `z9693022` can **r**ead this file, but not **w**rite to it or e**x**ecute it (“`r--`”)
  - Anyone in the group `unsw` cannot **r**ead, **w**rite or e**x**ecute this file (“`---`”)
  - Everyone else on the system cannot **r**ead, **w**rite or e**x**ecute this file (“`---`”)
- **`file3`** permissions (“group readable file”):
  - The owner `z9693022` can **r**ead and **w**rite this file, but not e**x**ecute it (“`rw-`”)
  - Anyone in the group `spree` can **r**ead this file, but not **w**rite to it or e**x**ecute it (“`r--`”)
  - Everyone else on the system cannot **r**ead, **w**rite or e**x**ecute this file (“`---`”)

Directory permissions



```
drwxr-xr-x. 2 z9693022 unsw     26 Sep 28 07:52 dir1
drwx------. 2 z9693022 unsw      6 Sep 28 07:47 dir2
drwxr-x---. 2 z9693022 cstelec   6 Sep 28 07:47 dir3
```

Permissions for **others** (everyone else on the system)

Permissions for **group**

Permissions for user **owners**

File type: “d” means a directory

- Directory permissions are **read**, **write** and **execute** (“r”, “w”, “x”)
  - Read: can *list* files in this directory and *read* such files (if the files permit it)
  - Write: can *write* to files (if the files permit it) and *delete* any files in this directory
  - Execute: can *execute* (traverse or search) the contents of this directory
- A hyphen “-” indicates that permission (read, write, execute) is *not* given
- Owner permissions override group permissions; group’s override others’ permissions!

Directory permissions

```
drwxr-xr-x. 2 z9693022 unsw     26 Sep 28 07:52 dir1
drwx------. 2 z9693022 unsw      6 Sep 28 07:47 dir2
drwxr-x---. 2 z9693022 cstelec   6 Sep 28 07:47 dir3
```

- **`dir1`** permissions (“publically readable directory”):
  - The owner `z9693022` can **r**ead, **w**rite and e**x**ecute (traverse) this directory (“`rwx`”)
  - Anyone in the group `unsw` can **r**ead and e**x**ecute this directory, but not **w**rite to it (“`r-x`”)
  - Everyone else on the system can **r**ead and e**x**ecute this directory, but not **w**rite to it (“`r-x`”)
- **`dir2`** permissions (“private directory”):
  - The owner `z9693022` can **r**ead, **w**rite and e**x**ecute (traverse) this directory (“`rwx`”)
  - Anyone in the group `unsw` cannot **r**ead, **w**rite or e**x**ecute this directory (“`---`”)
  - Everyone else on the system cannot **r**ead, **w**rite or e**x**ecute this directory (“`---`”)
- **`dir3`** permissions (“group readable directory”):
  - The owner `z9693022` can **r**ead, **w**rite and e**x**ecute (traverse) this directory (“`rwx`”)
  - Anyone in the group `cstelec` can **r**ead and e**x**ecute this directory, but not **w**rite to it (“`r-x`”)
  - Everyone else on the system cannot **r**ead, **w**rite or e**x**ecute this directory (“`---`”)

Changing group ownership

- To change the group owner of a file, use `chgrp` *`GROUP_OR_GID FILENAME`* …

```
$ chgrp spree ./file1 ../dir2
```

- To change all files and subdirectories within a directory, use the `–R` (recursive) option

```
$ chgrp -R spree dir1
```

- For system security, you can only change to groups that you are a member of!
- Only system administrators can place you in a group that you are not yet a member of
  - For Katana: send an email to *itservicecentre@unsw.edu.au* mentioning Katana and the group you would like to join. This will require seeking permission from the group owner (otherwise you would get access to their potentially sensitive files).
  - On Gadi, use the Mancini system to request the group owner permission to join that group. Once granted, the system administrators automatically add you to the group.

Changing file and directory permissions

- To change the permissions on a file or directory, use `chmod` *`PERM FILENAME`* …

```
$ chmod u=rw,go= ./file1 ./file2 dir1/file3
```

- To change all files and subdirectories within a directory, use the `-R` (recursive) option

```
$ chmod -R u+rwX,g+rX-w,o-rwx dir1
```

- Permissions *PERM* have the format **[*WHO*][[+-=][*WHAT*][,…]]**
  - *WHO* is one or more of “**u**” (user/owner), “**g**” (group), “**o**” (others) or “**a**” (all: user *and* group *and* others: “**a**” is the same as “**ugo**”)
  - “**+**” adds permissions, “**-**” removes them and “**=**” sets the permissions to be exactly what is specified in *WHAT* for *WHO*
  - *WHAT* is one or more of “**r**” (**r**ead), “**w**” (**w**rite), “**x**” (e**x**ecute), “**X**” (execute only if the execute permission is already set for the user)

Changing file and directory permissions

- Examples

```
$ chmod u=rw,go= file1
```

— Give user read/write permission only to `file1`, group/other no permissions

```
$ chmod a+rX file1
```

— Add read permissions to `file1` for all (user, group, others), add the execute permission if the user (owner) already has that permission

```
$ chmod -R g+rX dir1
```

— Recursively add read and possibly execute permissions for those in the group for all files and subdirectories in `dir1`, don’t touch permissions for user (owner) or others

```
$ chmod go-rwx file1
```

— Remove read, write and execute permissions for group and other on `file1`, don’t touch permissions for user (owner)

Finding file and directory permissions

- To discover the permissions on a file or directory, use the `ls -al` command
- To discover whether you have files or directories with specific permissions, use `find` *`STARTDIR`* `-perm` **[-/]***`PERM`*

```
$ find dir1 -perm /go+w
```

- *PERM* is the same as for the `chmod` command: **[***WHO***][[+-=][***WHAT***][,...]]**
- You can put an optional "**-**" or "**/**" in front of *PERM*
  - No "**-**" or "**/**" means the permission is *exactly* as specified by *PERM*
  - "**-**" means *all* permission bits in *PERM* are set, but other permissions may be present as well
  - "**/**" means *any* of the permission bits in *PERM* are set (not necessarily all)

How do you share files and directories?

- Change the group owner of files and directories to the group you want to give access to

```
$ chgrp -R spree dir1
```

- Change the permissions of files and directories as required

```
$ chmod -R u+rwX,g+rX-w,o-rwx dir1
```

- Check that permissions are as you expect

```
$ ls -al dir1
$ find dir1 -perm /go+w
```

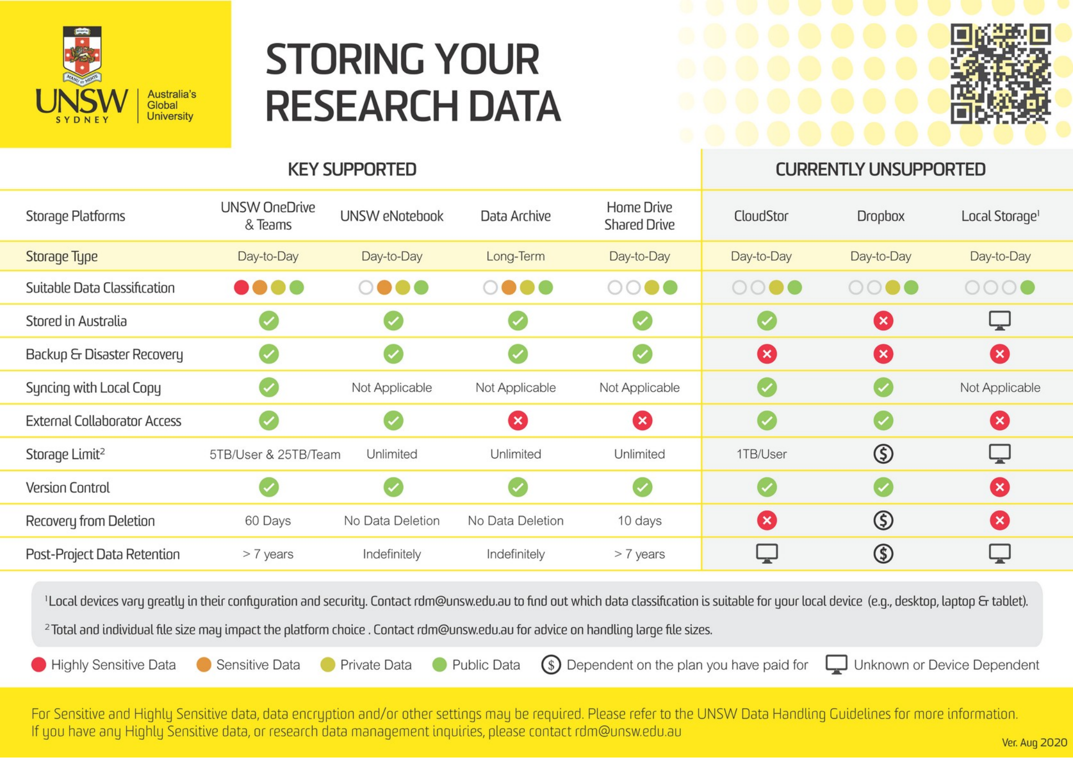
— this will list any files that are writable by the group and/or by others

What if I want to learn more?

- Many good books about the Linux command line
  - William Shotts, *The Linux Command Line*, 2nd edition, No Starch Press, 2019. ISBN 9781593279523.
  - Daniel J. Barrett, *Efficient Linux at the Command Line*, O'Reilly Media, Inc., 2022. ISBN 9781098113339.
- Read the manual pages for full technical information

```
$ man chmod
$ man chgrp
$ man find
$ man sg
$ man id
$ man getent
```

Sharing outside HPC systems



UNSW SYDNEY Australia's Global University

STORING YOUR RESEARCH DATA

	KEY SUPPORTED				CURRENTLY UNSUPPORTED		
Storage Platforms	UNSW OneDrive & Teams	UNSW eNotebook	Data Archive	Home Drive Shared Drive	CloudStor	Dropbox	Local Storage[1]
Storage Type	Day-to-Day	Day-to-Day	Long-Term	Day-to-Day	Day-to-Day	Day-to-Day	Day-to-Day
Suitable Data Classification	🔴🟠🟡🟢	○🟠🟡🟢	○🟠🟡🟢	○○🟡🟢	○○🟡🟢	○○🟡🟢	○○○🟢
Stored in Australia	✓	✓	✓	✓	✓	✗	🖥
Backup & Disaster Recovery	✓	✓	✓	✓	✗	✗	✗
Syncing with Local Copy	✓	Not Applicable	Not Applicable	Not Applicable	✓	✓	Not Applicable
External Collaborator Access	✓	✓	✗	✗	✓	✓	✗
Storage Limit[2]	5TB/User & 25TB/Team	Unlimited	Unlimited	Unlimited	1TB/User	$	🖥
Version Control	✓	✓	✓	✓	✓	✓	✗
Recovery from Deletion	60 Days	No Data Deletion	No Data Deletion	10 days	✗	$	✗
Post-Project Data Retention	> 7 years	Indefinitely	Indefinitely	> 7 years	🖥	$	🖥

[1] Local devices vary greatly in their configuration and security. Contact rdm@unsw.edu.au to find out which data classification is suitable for your local device (e.g., desktop, laptop & tablet).

[2] Total and individual file size may impact the platform choice . Contact rdm@unsw.edu.au for advice on handling large file sizes.

🔴 Highly Sensitive Data 🟠 Sensitive Data 🟡 Private Data 🟢 Public Data $ Dependent on the plan you have paid for 🖥 Unknown or Device Dependent

For Sensitive and Highly Sensitive data, data encryption and/or other settings may be required. Please refer to the UNSW Data Handling Guidelines for more information.
If you have any Highly Sensitive data, or research data management inquiries, please contact rdm@unsw.edu.au

Ver. Aug 2020

With whom do I discuss my HPC needs?

1. Your colleagues
2. Your supervisor
3. Hacky Hour: every Thursday 3pm on Microsoft Teams (Research Technology Training, Hacky Hour channel)
4. The Research Technology Services team
   - John Zaitseff J.Zaitseff@unsw.edu.au
   - The whole team at UNSW restech@unsw.edu.au

https://restech.unsw.edu.au/

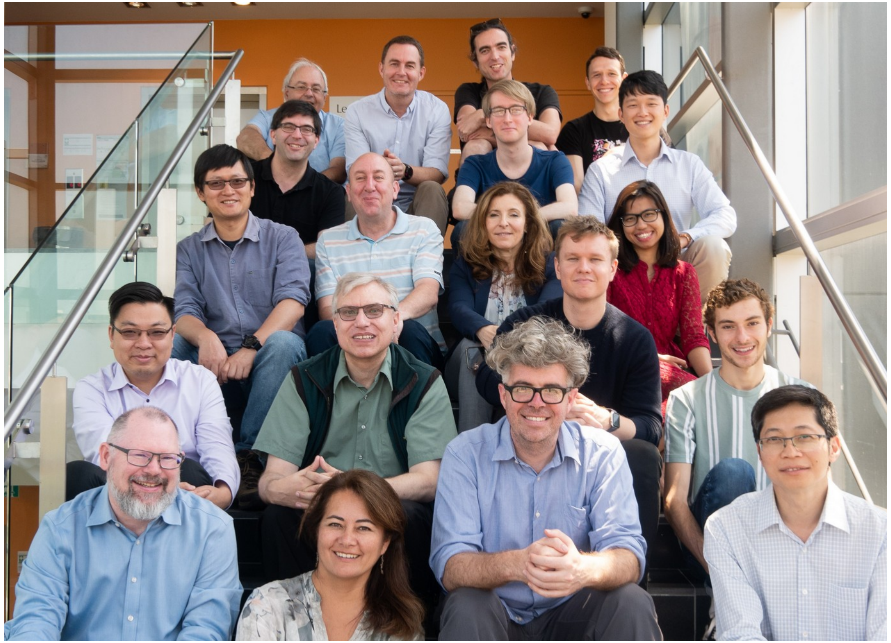


Image credit: UNSW Sydney