Google drive is a great tool, specifically we’ve been using “G Suite” (the equivalent of google drive but for businesses), for a long time. Lately I noticed it’s missing an important feature – monitoring file shares and permission of google drive items across organization is non-trival (at least in the G suite basic subscription).

I wanted to get a better sense of how my files and folders are shared across users within and outside the organization. I decided to give the googledrive package a try and extract the shares and permissions of important folders I had on my account. Here’s how I did that.

**Using googledrive to mass extract shares and permission of google drive items**

First, I wanted to focus on specific folders which contain a lot of subfolders. My goal was to generate a tibble with the names of all sub-items (folder or files one level under the specific folders), along with all users which have access to these folders.

Here is a description of what you need to do in order to accomplish what I described, followed by the code I used.

1. Get the id (or URL) for the folder you want to retrieve data from. You can get the id by just visiting the folder, and the id is in the URL, i.e., <https://drive.google.com/drive/u/1/folders/>.
2. Then, retrieve all items within the folder.
3. Use purrr functions (i.e., map and map\_df to iterate over the results and bring them into a tidy form).
4. (Optional) Use pivot\_wider to create a tibble in a wide format where each row is an item and each column is the type of access each user has on the item.

The first function I’m using is a function I defined called get\_permissions\_df():

library(tidyverse)

library(googledrive) # the package used to iterface with the google api

# Function to retrieve email addresses of permissions (read/write) --------

get\_permissions\_df <- function(permission\_list){

map\_df(permission\_list, ~{

if (!is.null(.$emailAddress)){

tibble(user\_email = .$emailAddress, user\_role = .$role)

} else {

tibble(user\_email = NA, user\_role = NA)

}

})

}

The function returns a tibble with two columns: user\_email and user\_role, according to the users with access to the folder (access can be owner/writer/reader).

Now, to actually pulling the data and processing it:

# Read the contents of the folder -----

# note that the first time you run this, you will be asked to login into your gmail using a web browser.

folder\_contents <- drive\_ls(as\_id("")) # replace here with the URL or ID of the folder.

# Get a tidy form of all shares and permissions of subfolders

tidy\_permissions <- folder\_contents %>%

mutate(new\_creds =

map(drive\_resource,

~{get\_permissions\_df(.$permissions)})

) %>%

select(name, new\_creds) %>%

unnest(new\_creds) %>%

filter(![is.na](http://is.na)(user\_email))

# Optional - turn into a wider form where each column is a user,

# each row is a subfolder, and values are permissions of users.

wide\_permissions <- tidy\_permissions %>%

distinct(name, user\_email, .keep\_all = T) %>%

pivot\_wider(id\_cols = name,

names\_from = user\_email, values\_from = user\_role, values\_fill = list(user\_role = "none"))

There you have it: tidy\_permissions will hold the names of all subfolders with permissions. A folder will appear as many times as it has permissions (with the user email and the type of permission). The wide\_permissions will hold a wide version in which each row is a folder and each column is a user.

Note that this works for specific folders. You can also use drive\_ls() without any arguments (or use it with recursive=TRUE), to pull everything on the drive (or everything within all subfolders, recursively). When I did that it took me around 5-10 minutes to pull all the data and about 5 minutes to prepare it, since I have \(>100k\) items.

**Conclusions**

The post provides a method to create a concise tibble with the contents of you google drive items, and their user permissions.  
You can either run it on all items in your google drive or on selected folders (and sub-folders within). The method is especially useful in the context of data safety and security, when you want to make sure you are not sharing sensitive items in an undesired manner.