# Resource Automation Project

Aaron Steinberg

September 8, 2023

## Contents

# 1 The problem of resource abandonment

## Managers and Supervisors

When a manager, supervisor, or some sort of team lead creates a shared resource, such as a Shared Drive, they become the owner of the resource. Years can go by, and teams can grow, and many different people can use this shared resource. When the owner of a shared resource leaves the university, their status of ownership does not magically pass down to their immediate subordinate or supervisor, yet. Right now, lots of support tickets come in that involve admins manually making these changes.

## The Cost of Delayed Action

Over time, the distance between current owner and original owner can grow considerably. For example, an owner might leave the university in 2023, their immediate supervisor might leave in 2024, and new hires/replacements in years following might completely change the hierachy in the position. Now, it might become unclear who to transfer the ownership of the still regularly shared resources from the initial current manager, years later, when the first problems arise. This takes more IT time. At some point in time after enough neglect of resource ownership and management, instead of following a paper trail to discover the natural hierachy, a hierachy need be redfined. This is the true cost of delayed action.

## More Risks

Besides the amount of time Google Admins may spend on this problem, the fact is there is a good chance certain rules about privacy and tiered data stop being met when former employees still have access to their large shared drives. This is just one example, but there are more risks to keeping the status quo. There is a more-than-zero probability we already have policy infractions happening directly because of resource mismanagement. This is a very real liablity.

# 2 Identifying Abandoned Resources

## Defining an Abandoned Resource

An abandoned resource in our context is specicifically an abandoned shared resource. It is a shared resource whose owner is no longer affiliated with the university. An abandoned resource can be and is regularly used after it has been abandoned. In a Google sense, this is an account where the active field is set to false.

### How to Identify an Abandoned Resource in a Query

Identifying an abandoned resource with an automated query is possible and the way to go. In a micro scale, you can see if you are a member of an abandoned resource by doing the following:

- For all of your shared resources in your drive that you do not own(a subset of the Shared with Me tab):
- Check if the owner is still an active member of the university
- If they are not still active, this resource is abandoned

This of course, can only check shared resources you are a member of. On a macro scale, where we would need to check for abandoned resources for the entire workspace, things change.

### The Queries and Required Steps

On a macro scale, here is what is necessary:

- A Google Cloud project with scope for all Google Drive files
- An OAuth2.0 client ID
- A service account with read only permissions of all drive meta data in the workspace
- Potential Requirement: scope to all user metadata(email, active status, etc)

With these requirements met, we can query for the information that we need. *These approaches are listed on the next page.*

**Approach 1:**

- For every shared resource in the workspace:
- Check if the owner is a non-active account.
- If the owner is non-active, then the resource is abandoned.
- Return the abandoned resources in an actionable list

**Approach 2:**

- For every non-active user in the workspace:
- For each one of their shared resources:
- Mark down the shared resource as abandoned.
- Return the abandoned resources in an actionable list

**Comparing Approaches** Both of these approaches have working logic, but they do not perform the same. At the top level of the query: approach 1 goes through every single shared resource in the entire workspace, while approach 2 goes through every non-active user in the workspace. At the second layer, approach 1 performs negligible operations, but oeration 2 will go through each owned file of the non-active user. The operations and runtime of approach 2 will almost always be more efficient than approach 1, though.

# 3 Identifying Abandoned Resources as They Become Abandoned

The best approach is not simply running queries on their own. The best approach is to run a sufficient number of queries to solve the problem temporaily, and then introduce a new policy/script/query that takes place during offboarding.

During this redfined offboarding process, all shared resources will be transfered to supervisor or immediate subordinate if there is no supervisor. This can also be automated using a Google Console Project.

**When an Employee Leaves the University:**

- Identify the next in line or supervisor that assets will be transferred to.
- For each shared resource:
- Transfer to supervisor or mark abandoned if transfer is not possible

This approach shifts the window of the solution to policy rather than a fix it after the fact solution. Policy is the better choice because as discovered earlier in the document, after a certain amount of time, resources get harder and harder to identify and transfer.

# 4 Exploring Already Implemented Solutions - GAM

## Technology Used

1. **Advanced GAM**
   https://github.com/taers232c/GAMADV-XTD3
2. **Python**

## Their Approach

Using the power of GAM, this tool compiles a list of Team Drives owned by suspended users. The "suspended" verbage refers to their Google status, but in our case, it also refers to their status at UCSC. If an account is suspended in the Google ecosystem, it is no longer active in any part of the system. If an account is active in the Google ecosystem, it is not necessarily active everywhere else, due to the asynchronous nature of large systems like a Google workspace.

This script uses a variety of GAM commands in order to organize the data into machine readable files. The script then parses through these files. While there is some concern to efficiency in the approach this tool takes, this would be an easy option to modify.

The modification or supllementary program, would take the data from the outputted CSV, and perform some type of action.
This action could be deletion, reassignment of ownerhip, or whatever is deemed the best action to take.

This approach is not policy, and it is still a fix after the fact approach. However, this is a necessary step in attaining good policy. The problem must be fixed up to the present before any kind of resource policy can be established and kept up with. Moreover, this approach can trivially be modified to run during an offboarding process, and it then takes the form of policy rather than a fix.

# 5 Necessary Sub-functionality

## Finding supervisors and direct subordinates

Finding supervisors or subordinates of a suspended user is not incredibly straight forward. For one, not every team will have accurate, up-to-date information about their inner-hierachy.

And, even if they all did have accurate information, it might not be stored or accessible in the same way.

In order to actually automate this process of transferring abandoned resources to the next in line, we need to further standardize how this data is structured.

## Offboarding Overhauls

There are multiple reasons why this project is necessary, but a big one is a lack of a complete offboarding process in regards to Google assets. When an employee offboards, there needs to be a step where shared assets are transfered on the spot. This will eventually completely erase the need of a computationally expensive tool like the described in this document.

Of course, before that happens, it is probably a good idea to fix all abandoned resources in the system. This is what has been described in previous sections.

# Overall Conclusion and Steps to Success

1. Fix everything up to the *present* using one of the methods described.
2. Overhaul the method that was used to fix the issue into a policy that prevents the issue.
3. Update and allow the policy to take different forms, as the UCSC Google ecosystem changes, in a way where abandoned resources no longer have a place.
4. Enforce the policy.

# 6 Links

- https://github.com/taers232c/GAM-Scripts3/blob/master/GetTeamDriveSuspendedUsersACLs.py
- https://github.com/taers232c/GAMADV-XTD3