1 | Understanding Authorization

To be able to model and create intuitive and understandable authorization flows, one must understand the basis of authorization flows and the elements by which they are made successful.

In the most simply basis, authorization is the process by which permissions are assigned to a user. There are a few models by which authorization is done, and we will aim to list a few successful systems and their downfalls.

To begin this discussion, we will aim to describe a few terms:

- `model`: a system of authentication, like the ones discussed below
- `agent`: the software tool in a `model` by which authorization is checked
- `rule`: a statement made available to the `agent` to validate claims made by `users`
- `resource`: a file/page/tool by which the `model` aims to protect
- `action`: what the `agent` grants to do to a `resource`
- `user`: an actor leveraging the `model`'s `agent` to gain perform `actions`

1.1 | UNIX/BSD PAM

The PAM authentications model, manifested in the `/etc/shadows` files on most *nix systems, is one of the most familiar system of authentication to most.

PAM protects individual resources by checking for an octal permission representing whether or not an `action` on a `resource` is accessible to a `user` or a group of users.

There exists 3 `actions`: read, write, and execute. The authorization `rules` are determined on a `resource` level, and `agents` check against `rules` on access time by users. User permissions override group permissions, which override global permissions.

This systems does not have permission dependencies nor exceptions, group grants and are the only batch executor available.

1.2 | AFS

jack

1.3 | Amazon IAM

1.4 | Microsoft Graph Permissions

1.5 | OAuth