Helpful GitHub Repos

This list of helpful GitHub Repos contains both community originating tooling as well as tooling from outside the community that schools have found useful.

Add a Community Cloud Repo Here (email contact on repo for access): https://github.internet2.edu/cloud

Standardized IAM Role Application and Management

- University of Arizona Service Catalog
  - Support toolkit for AWS developed by UA
- University of Illinois' awscli-login
  - awscli-login is a plugin that allows retrieving temporary Amazon credentials by authenticating against a SAML Identity Provider (IdP). Includes support for multiple roles and Duo for MFA.
- Cornell Cross-account DBA Role
  - The shib-dba-role-with-assume-role.yaml CloudFormation template creates a shib-dba IAM role in an AWS account that also gives access to the Cornell CIT DBA team to use that role programmatically, via cross account permissions.
  - The assume-role.sh script is an example script that would be used by the Cornell CIT DBA team to exercise those cross-account privileges.
  - The instance-profile.yaml CloudFormation template was used by the Cornell CIT DBA team to setup the role they use to assume the shib-dba roles in other accounts.
- Cornell samlapi
  - You can use Cornell Shibboleth login for both API and CLI access to AWS. I built docker images that will be maintained by the Cloud Services team that can be used for this and it is as simple as running the following command:

    ```
    docker run -it --rm -v ~/.aws:/root/.aws dtr.cucloud.net/cs/samlapi
    ```

    After this command has been run it will prompt you for your netid and password. This will be used to login you into Cornell Shibboleth. You will get a push from DUO. Once you have confirmed the DUO notification, you will be prompted to select the role you wish to use for login, if you have only one role it will choose that automatically. The credentials will be placed in the default credential file (~/.aws /credentials) and can be used as follows:

    ```
    aws --profile saml s3 ls
    ```

- Capital One Cloud Custodian

  - Cloud Custodian is a rules engine for AWS fleet management. It allows users to define policies to enable a well managed cloud infrastructure, that's both secure and cost optimized. It consolidates many of the adhoc scripts organizations have into a lightweight and flexible tool, with unified metrics and reporting.

    Custodian can be used to manage AWS accounts by ensuring real time compliance to security policies (like encryption and access requirements), tag policies, and cost management via garbage collection of unused resources and off-hours resource management.

    Custodian policies are written in simple YAML configuration files that enable users to specify policies on a resource type (EC2, ASG, Redshift, etc) and are constructed from a vocabulary of filters and actions.

    It integrates with AWS Lambda and AWS Cloudwatch events to provide for real time enforcement of policies with builtin provisioning of the Lambdas, or as a simple cron job on a server to execute against large existing fleets.

    "Engineering the Next Generation of Cloud Governance" by @drewfirment
- aws-rotate-iam-keys
  - Rotate your IAM Keys to be in compliance with security best practices. AWS talks about rotating your keys every 30, 45, or 90 days. But who has the time to make their own script and remember to do that? I did. And security is easier when its less than 3 lines that you need to copy + paste to be secure. Now it's easy to rotate your IAM credentials nightly and wake up with more security than the day before.

Container Lifecycle Management

- CalTech deployfish
  - `deployfish` has commands for managing the whole lifecycle of your application:

    - Create, update, destroy and restart ECS services
    - Manage multiple environments for your service (test, qa, prod, etc.)
    - View the configuration and status of running ECS services
    - Scale the number of containers in your service, optionally scaling its associated autoscaling group at the same time
    - Update a running service safely
    - Run a one-off command related to your service
    - Configure load balancing
    - Configure application autoscaling
    - Configure service discovery
    - Configure placement strategies
    - Manage AWS Parameter Store and utilize in containers
    - Add additional functionality through modules
- Duke's Dockerized RStudio
  - This project is an example of running RStudio from within a Docker container. In addition to the basic RStudio server, the container also has the knitr and Rmarkdown libraries so it is easy to create nicely formatted output. There is also just enough of TeX to allow knitr to generate PDF output.

Cost Management

- costnotify
    - This repo is instructions and code for building an AWS lambda function *costnotify* to send a spending breakdown email to selected recipients for an AWS cloud account. Typically the email subject is "how much did I spend in the past 24 hours" and the email body is further details breaking down cost.