NASA Openscapes 2i2c JupyterHub Usage and Costs

Monthly report for July 2024

Introduction

A key objective of NASA Openscapes is to minimize "the time to science" for researchers. Cloud infrastructure can facilitate shortening this time. We use a 2i2c-managed JupyterHub ("Hub"), which lets us work in the cloud next to NASA Earthdata in AWS US-West-2. The purpose of the JupyterHub is to provide initial, exploratory experiences accessing NASA Earthdata in the cloud. It is not meant to be a long-term solution to support on-going science work or software development. For those users that decide working in the Cloud is advantageous and want to move there, we support a migration from the Hub to their own environment through Coiled.io, and are working on other "fledging" pathways.

The main costs of running the JupyterHub come from two sources:

1. Compute, using AWS EC2
2. Storage using AWS EFS, via storage in users' home directories

Compute costs scale up and down as the Hub is used, however storage costs are fixed - we pay for "data at rest", with ongoing daily costs/GB even while the Hub is not running.

Storing large amounts of data in the cloud can incur significant ongoing costs if not done optimally. We are developing technical strategies and policies in the Earthdata Cloud Cookbook to reduce storage costs that will keep the Openscapes 2i2c Hub a shared resource for us all to use, while also providing reusable strategies for other admins.

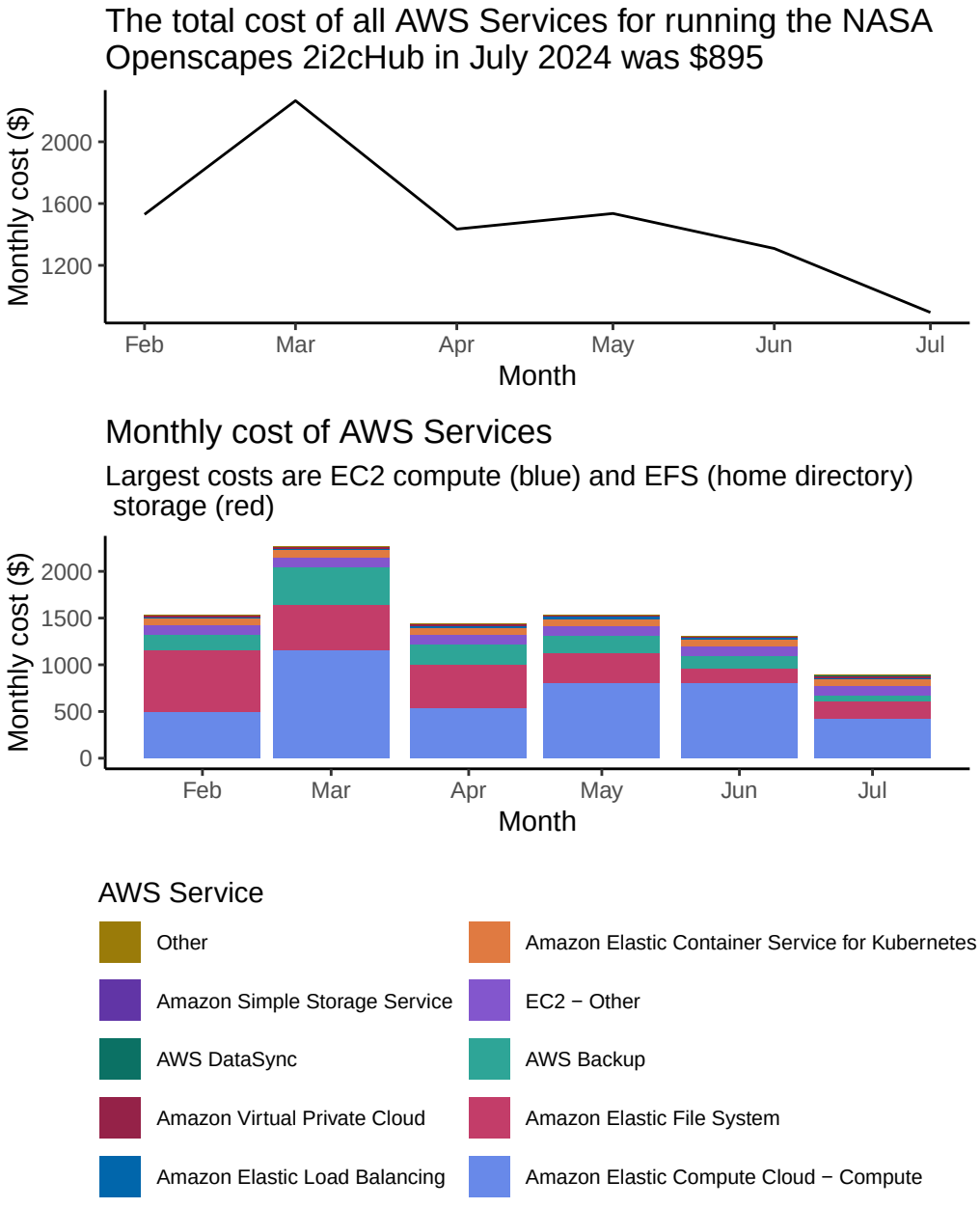
This report is intended to give a monthly summary of usage of the Hub and its resources, by tracking metrics on costs and usage of key components of storage (EFS) and compute (EC2).

Month over month changes

A comparison of monthly costs in the Hub can help us to compare usage over time and identify longer-term patterns. We can query the AWS Cost Explorer API to explore these costs.

Total Costs

The following plot shows the total monthly costs of all AWS services related to the Hub, as well as a breakdown of costs by service each month.

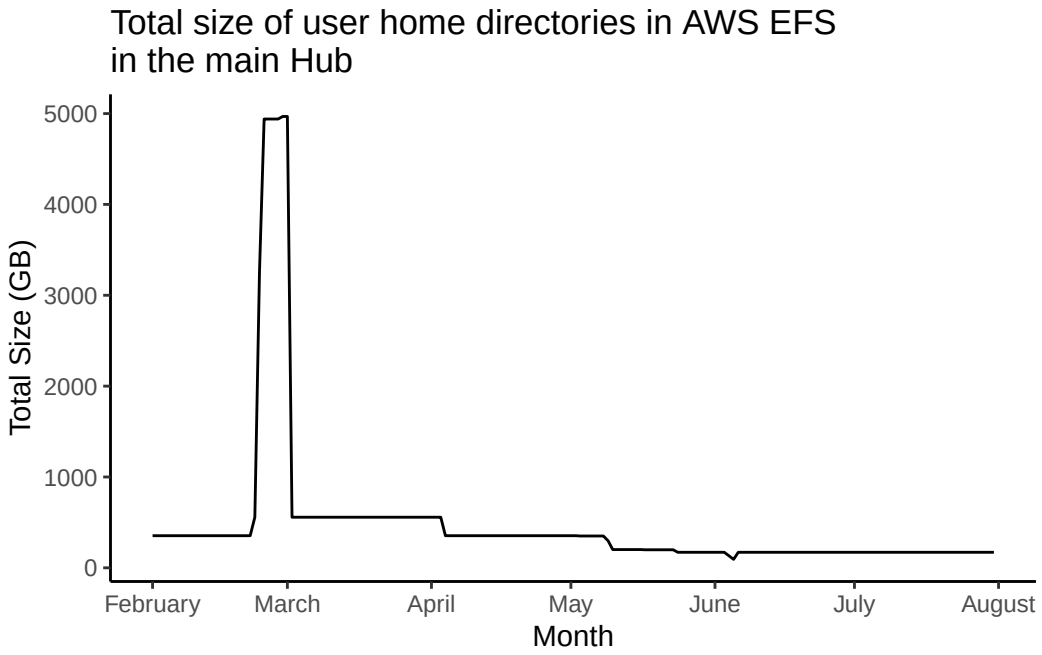


*The top nine services are shown individually, with any remaining grouped into 'Other'

Storage

Managing storage is an effective way to manage long-term costs in the Hub, as data-at-rest is an ongoing cost, much of which can be avoided by monitoring and reducing storage of data that is not required.

User home directories are in an AWS "Elastic File System" (EFS) mount, which is a relatively expensive option for long-term storage of large files. The following figure plots the daily total size of data storage in the user home directories in the Hub over the past six months. The size of the home drives is directly correlated with the costs for "Amazon Elastic File System" in the previous chart.



Detailed breakdown for the month of July

To understand more about usage and costs during the current month, we can look at daily usage metrics and costs.

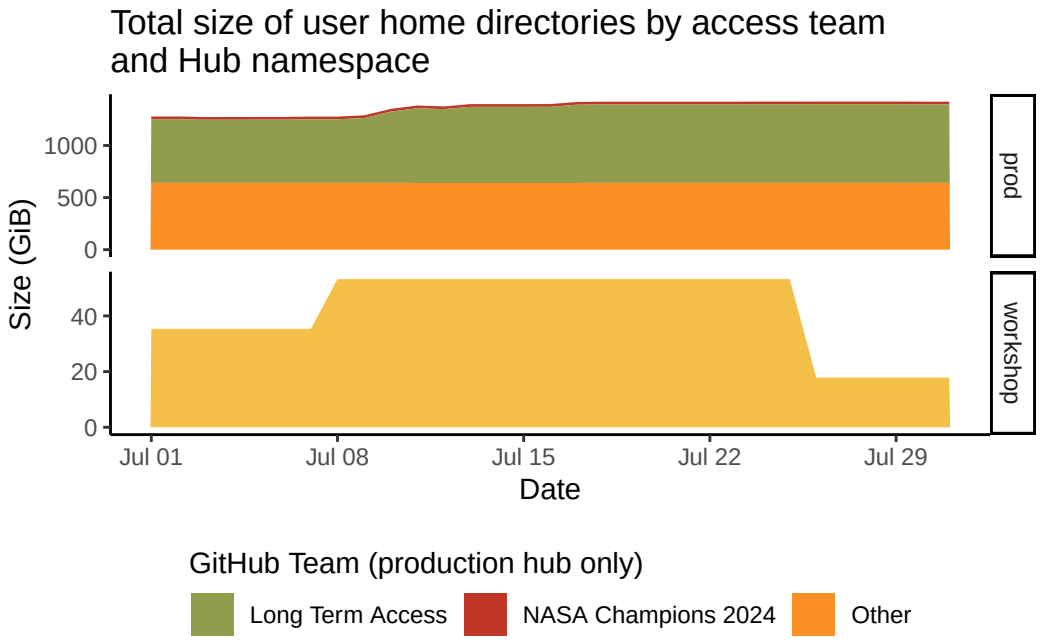
Home directory sizes

The Hub can currently be accessed via two different "namespaces": "production" (or "prod"), and "workshop". The production namespace is where participants are given medium to long-

term access, as NASA mentors, Champions participants, etc. Access is managed via GitHub by assigning user's GitHub usernames to specific teams.

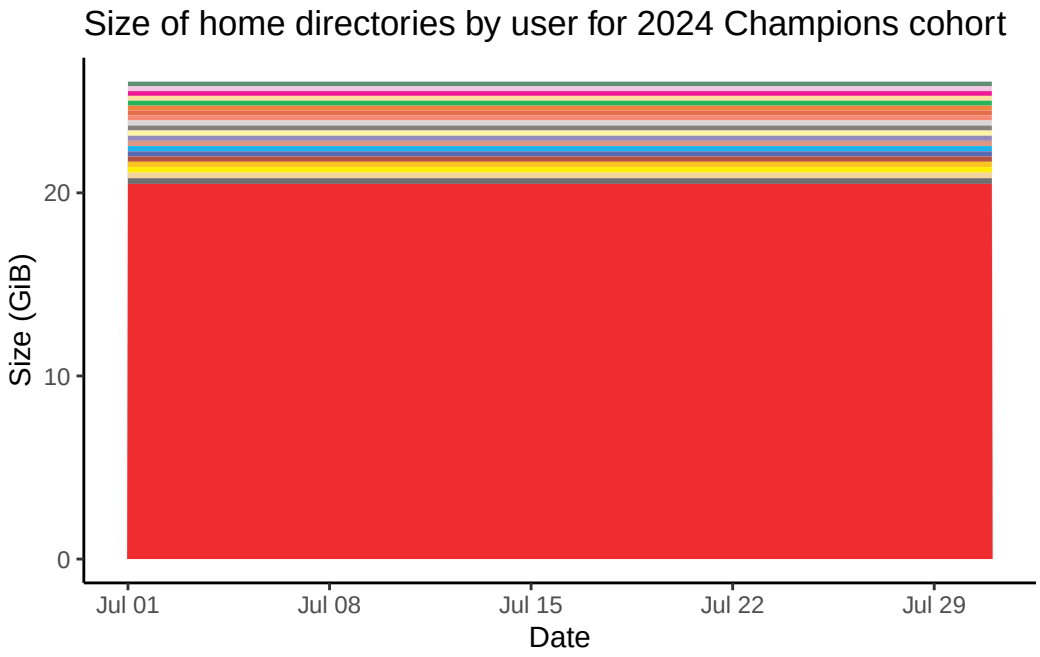
The "workshop" namespace is used specifically for large workshops and access is granted on the day of the workshop by use of a shared password rather than using GitHub teams. Access is short-term and usually revoked a week after the workshop, at which point users' home directories are removed.

The following figure shows the total size of home directories by namespace. Note the different y axis scales in each panel. The "prod" namespace panel is broken out by the GitHub team by which they are granted access to the Hub (Long-Term Access and NASA Champions 2024).



Champions cohort

It is also helpful to look more deeply into the Champions cohort to see how they are using the Hub, and how much storage is being used. The following figure breaks down the home directory size of Champions by user - usernames are not displayed, but we can see if any users are using a disproportionate amount of space. When we see disproportionate amount of space used, we reach out to users and work with them to reduce their storage, and update the Cookbook tutorials as needed.

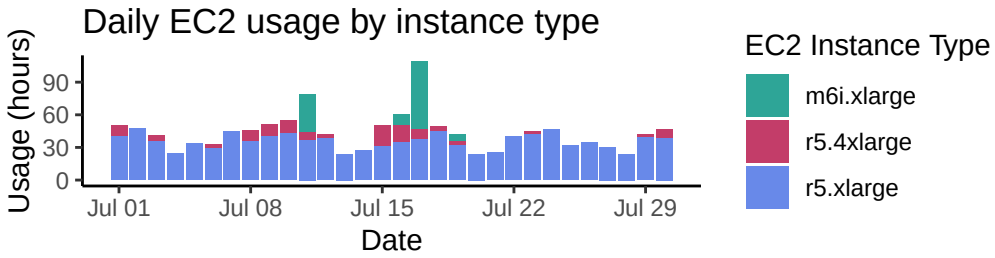


Compute costs and usage

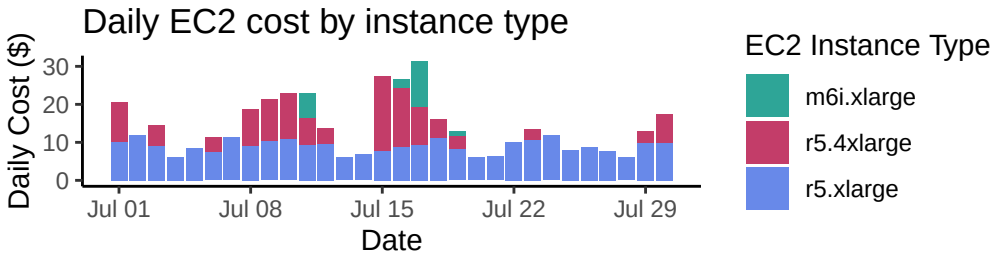
When a user logs into the Hub, they can choose the amount of RAM and number of CPUs they would like to use, enabling them to scale computing power appropriate to the tasks they are running. More powerful compute resources have higher hourly costs, so it is important to not choose a powerful instance when it isn't required.

Examining both the usage and the costs of the EC2 instance types that users choose can help us understand users's needs as well as compute costs. This helps us develop policies and recommendations for Hub compute usage.

The following plots show the usage and costs broken down by instance type. The compute profiles that users can choose from run on `r5.xlarge` (4 CPUs, 32 GiB memory) or `r5.4xlarge` (16 CPUs, 128 GiB memory) instances. Note that during some large workshops, administrators will choose very large instance types (for example `r5.16xlarge`; 64 CPUs, 512 GiB memory) so they can provision a small number of nodes with many users per node. This is more efficient than launching many nodes at once. Other instance types, such as `m6i.xlarge` indicate usage of the AWS infrastructure outside of the Hub, mostly using coiled.



*Hub resource allocation options up to 3.7 CPUs run on 'r5.xlarge' instances, and those with up to 15.6 CPUs run on 'r5.4xlarge' instances.



Finally, it is useful to look at the relationship between compute hours and total cost by instance type, to understand both the highest cost and highest usage, as well as the cost-efficiency of the instance types.