stripe

Stress-Free Airflow development: From Dev to Prod @ Stripe

Nick Bilozerov and Thomas Tauber-Marshall

September 2024 @ Airflow Summit 2024

Stripe team

Meet us

Software Engineer at Data Orchestration team
2.5 years at Stripe
Airflow user from 2018

Software Engineer at Data Orchestration team
3.5 years at Stripe
Airflow infra engineer from 2021

stripe

Agenda

What we are going to talk about today?

- Stripe flavours of Airflow
- Deep dive into Stripe Airflow user testing environment - User Scope Mode
- Our learnings and the future
- *[Optional] User Discussion after the talk*

What is Stripe

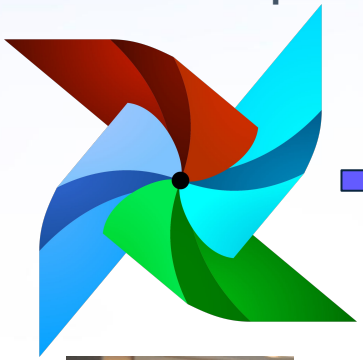
- Payment infrastructure of the internet
- Stripe employs around 7,000 people across 23 countries
- Our users processed a collective $1 trillion on Stripe in 2023, equivalent to 1% of global GDP

Airflow scale

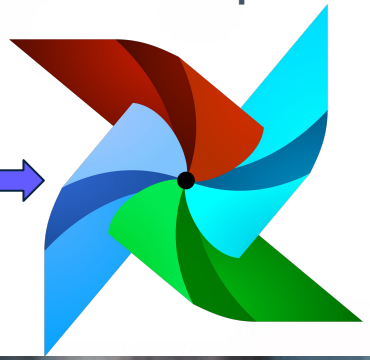
- 250+ dags
- 150k+ tasks
- 500+ teams
- Processing multiple petabytes of data daily

From a Fork to the Better World

Airflow1.10 Stripe fork



Airflow2.8 upstream!



Specific Airflow Task definition

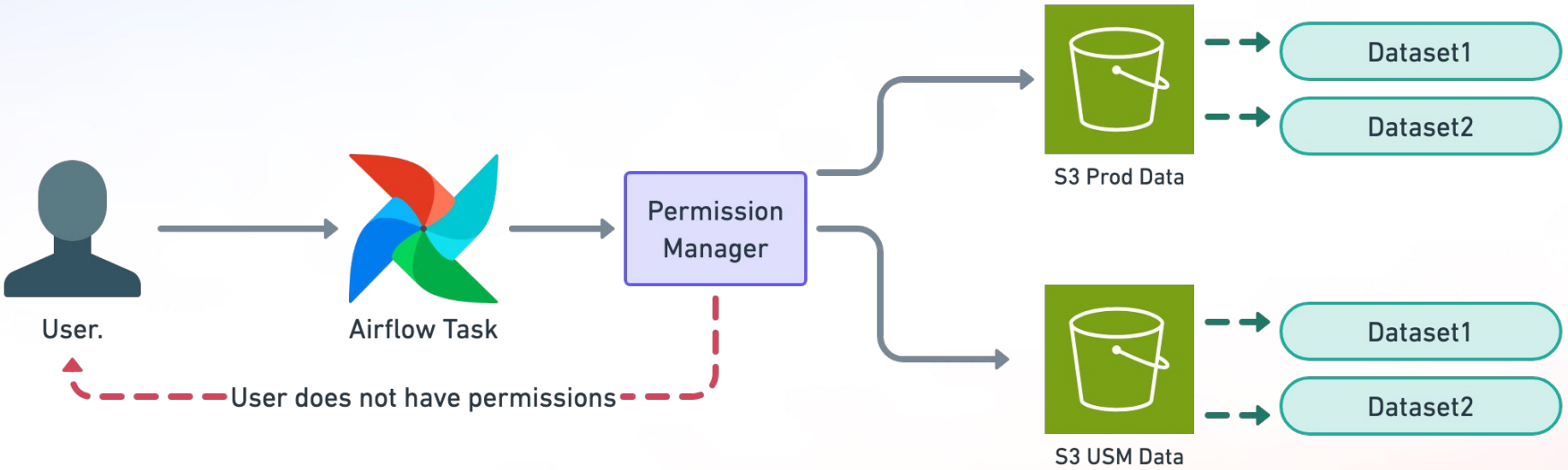
```
class Task(TaskBaseClass):
  def config():
      return {
          "hadoop_cluster": "prod_hadoop_cluster",
          "hadoop_queue": "prod_hadoop_queue",
          "custom_config": "prod_custom_config",
      }

  # s3://data-lake-prod/output_dataset/2024/08/24/
  def output(self):
      return Dataset(name="output_dataset",
                     version="{{ ts_nodash }}")

  def job(self):
      return JobRunner(
           "my_ml_training_job",
 # s3://data-lake-prod/output_dataset/2024/08/24/
           output=self.output(),
 # s3://data-lake-prod/input_dataset/2024/08/23/
           input=UpstreamTask().output()
      )
```

Meet User Scope Mode aka USM

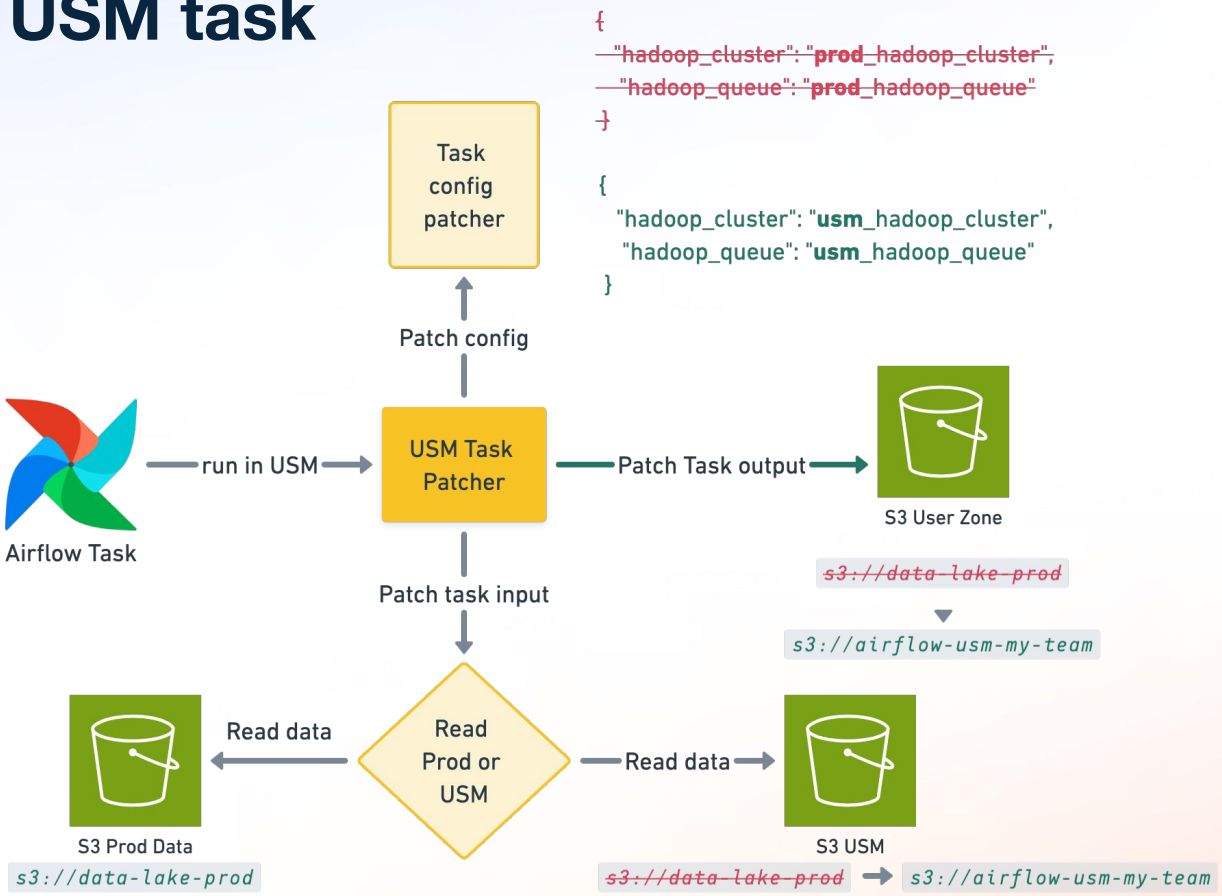
Powerful concept we use at Stripe that helps magically transform a task definition and test your changes with a breeze.

- USM allows to run a job without impacting production data with production like settings locally
- Run an Airflow job and compare the results to the most recent production job.

How to access data?



Patch USM task



User Scope Mode

Power of USM

Task: payments.Transactions

Config:

```
 - hadoop_cluster=usm_hadoop_cluster
 - hadoop_queue=usm_hadoop_queue
 - custom_config=usm_custom_config
```

Inputs:

- s3://data-lake-prod/input_dataset/2024/08/23/

Outputs:

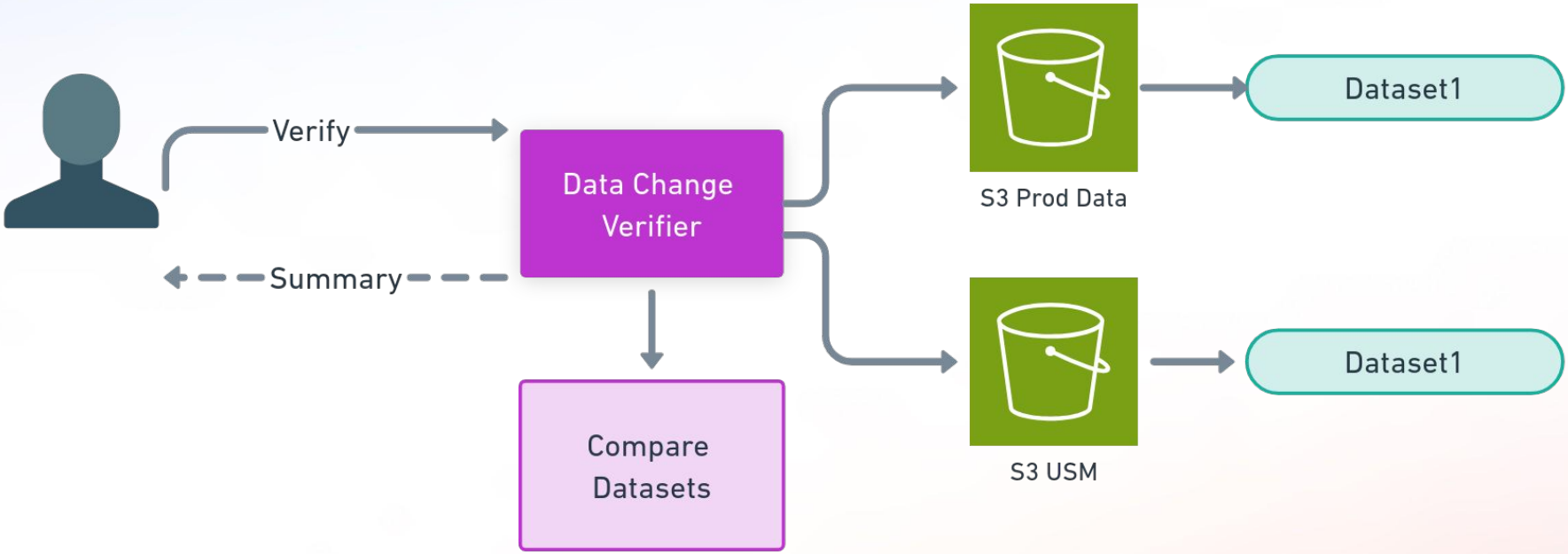
- s3://airflow-usm-my-team/ # Team specific USM bucket
  tmp/user_name/testing/output/ # User specific USM prefix
  {timestamp}/ # Timestamp of the USM run
  data-lake/paymens_transactions_dataset/2024/08/24/ # Original path to production output

(was 's3://data-lake-prod/paymens_transactions_dataset/2024/08/24/')

Evaluation: This task looks safe for USM

User Scope Mode

Datachange Verifier



User Scope Mode

Verify your change

Running verifier

```
verify-data-change --key key_to_compare_data \
--left s3://airflow-usm-my-team/tmp/user_name/testing/output/{timestamp}/output_dataset/2024/08/24/ \
--right
s3://data-lake-prod/dataset_name/2024/08/24/
```

Different options

--ignore-columns *Ignore differences in the given columns*
--schema-source *Set the source dataset to be used as the base schema during a comparison*
--rounding-scale *Instructs the diff algorithm to round doubles & floats to the desired scale, so that small differences in precision do not yield diffs*
--keys *compare records by forming a unique key from multiple fields*

User Scope Mode

Check results

Change summary

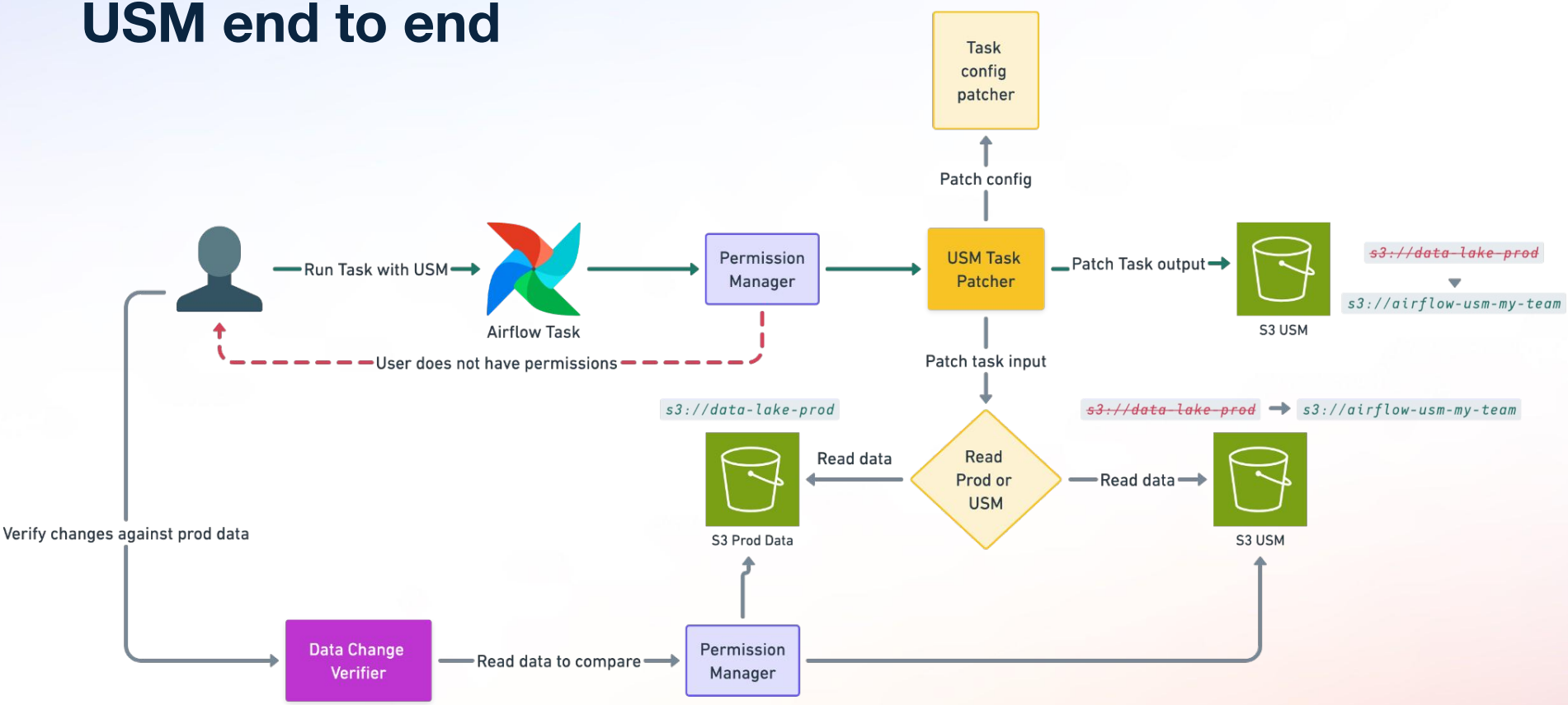

```
{
    "change_count" : 51,
    "left_only_count" : 10,
    "right_only_count" : 0,
    "diff_count" : 41,
    "schema_matches" : true,
    "content_matches" : false,
    "schema_diff_details" : [
    ],
    ...
}
```


Data difference

```
transaction_id = 123
merchant_id = 567
country = US
capture_amount_usd = 233
source_side = left

transaction_id = 123
merchant_id = 567
country = US
capture_amount_usd = 133
sourse_side = right
```

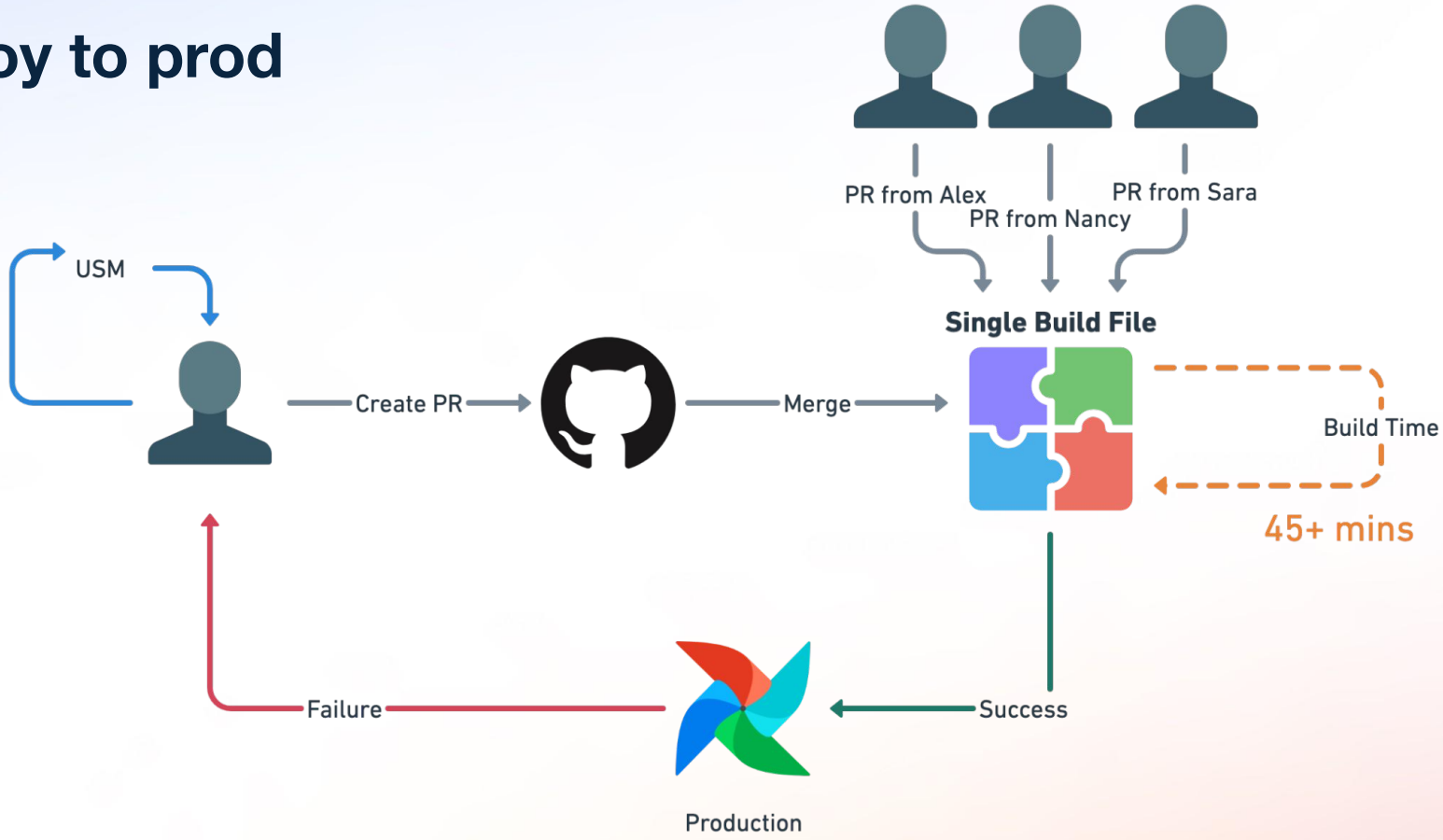
User Scope Mode

USM end to end



stripe

Deploy to prod



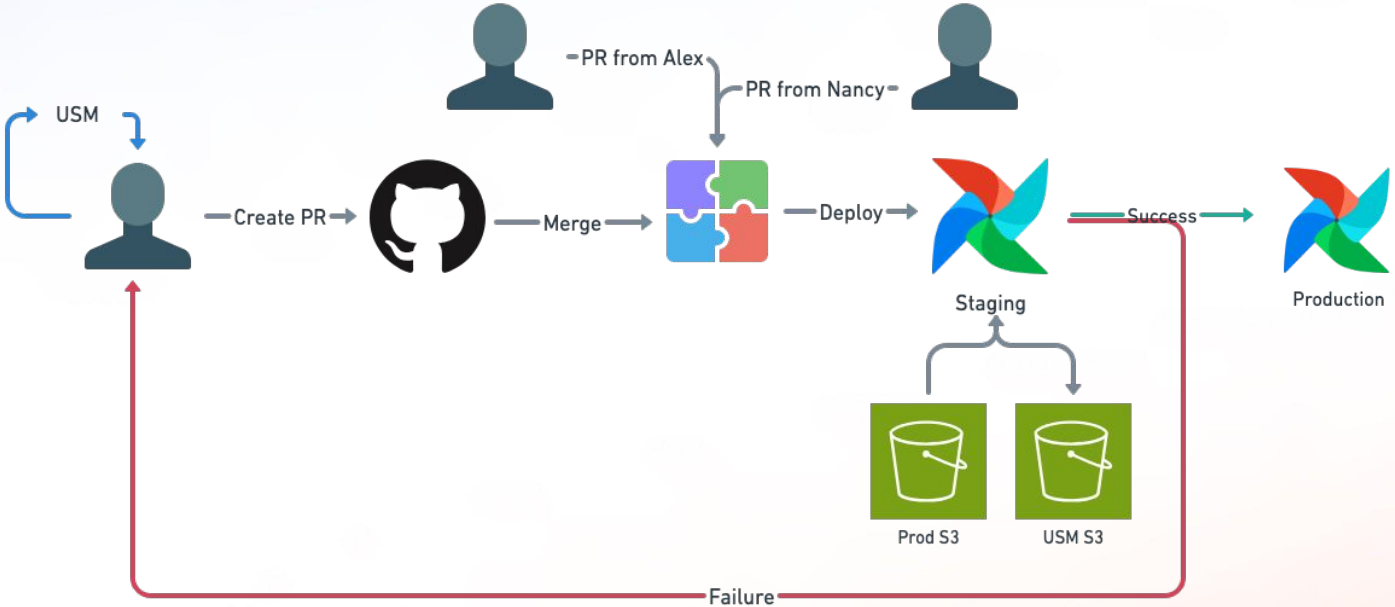
How can we do better?

Coming soon!

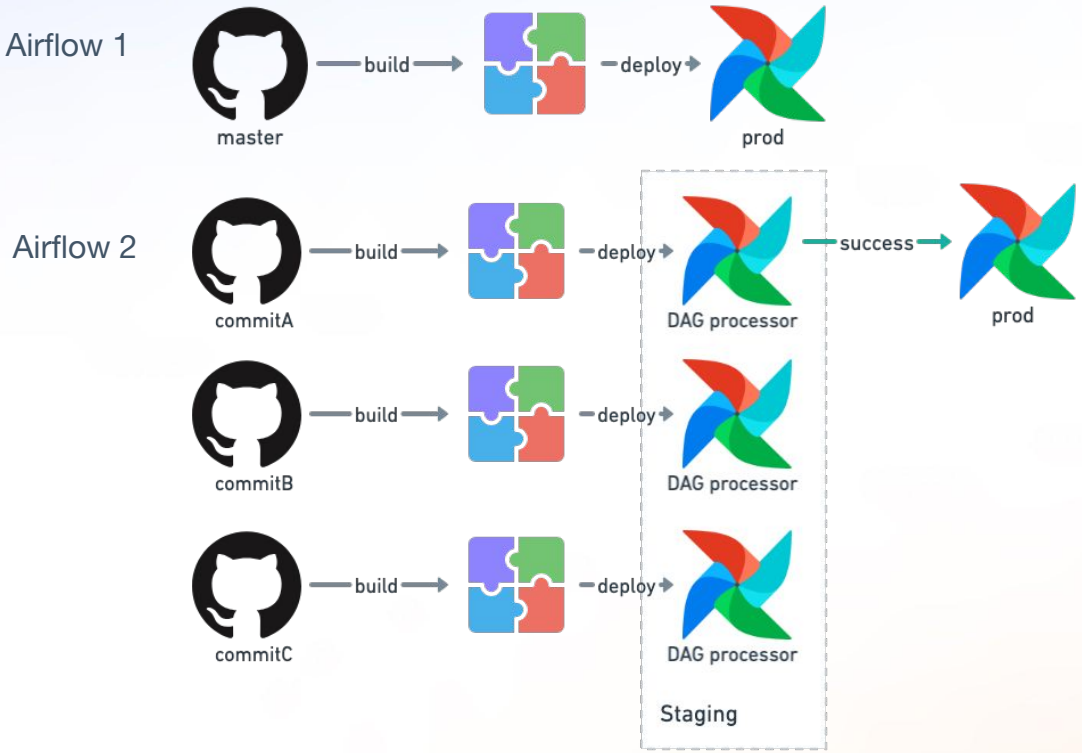
Downsides of USM Today

- Very manual process
- Does not closely match production behavior
- PRs that are tested separately may conflict when merged together and deployed
- Difficult to test long running pipelines or large numbers of tasks

Staging Environment



Independent Deploys



stripe

Questions?

We are hiring - US, US-Remote, Canada-remote