Sandia National Laboratories

Exceptional service in the national interest

ENABLING EVENT-DRIVEN WORKFLOWS WITH AWS AND THE SLURM API

Cory Lueninghoener, Sandia National Laboratories

Lowell Wofford, Amazon Web Services

SLUG 2024

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc. for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

SAND2024-11483C

WHOAMI

Cory Lueninghoener

- HPC Systems Guy

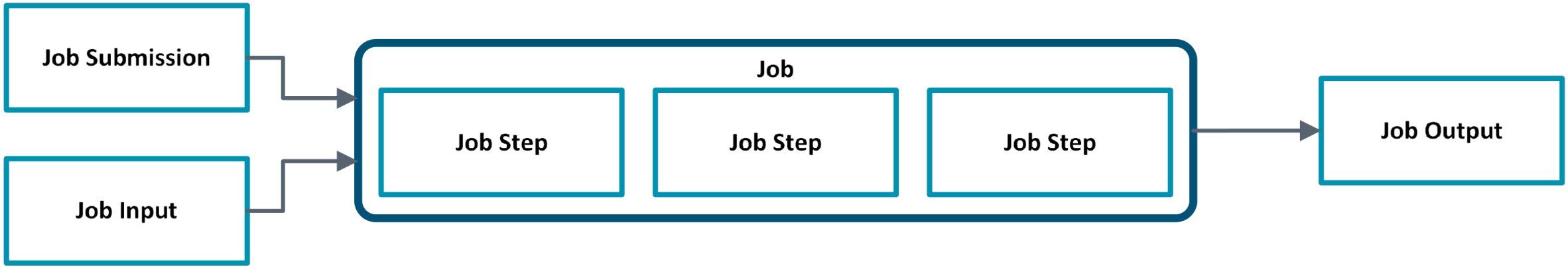
Heterogeneous Advanced Architectures Platforms Team, Sandia National Laboratories

- Hardware and software testbeds
- HPC systems design and development
- Advancing HPC operations

TODAY'S HPC JOBS

HPC jobs and workflows are still generally static or pre-defined

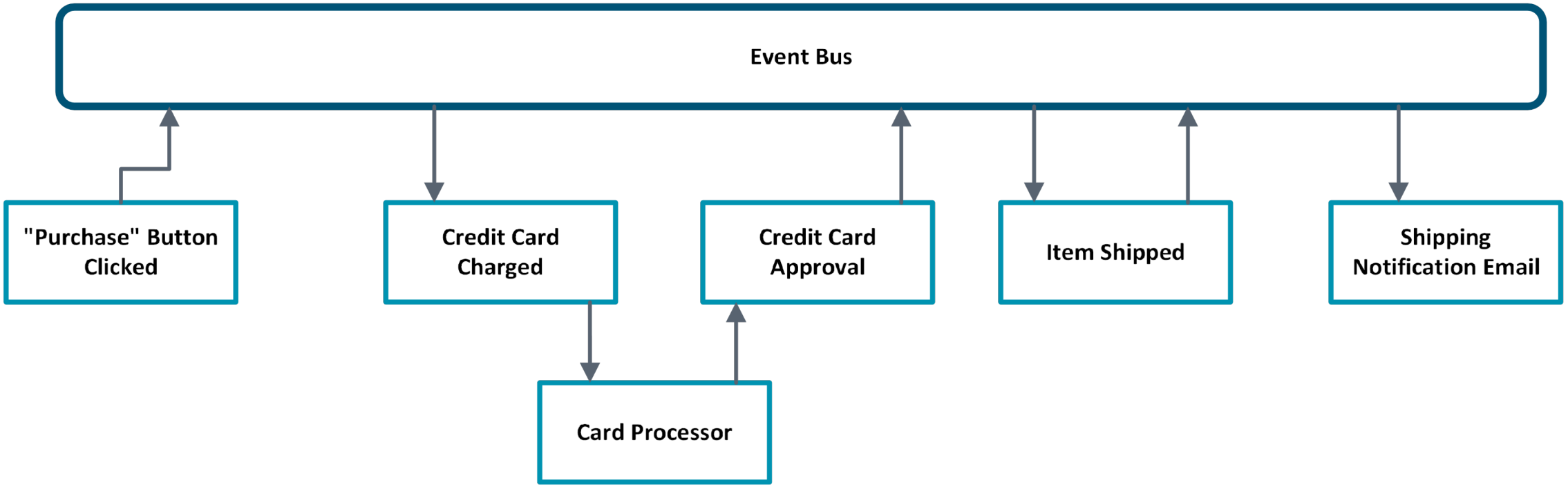
- Maybe ad-hoc or interactive
- Maybe scripted with Bash or something similar
- Maybe defined with Common Workflow Language or something similar



TODAY'S EVENT-BASED WORKFLOWS

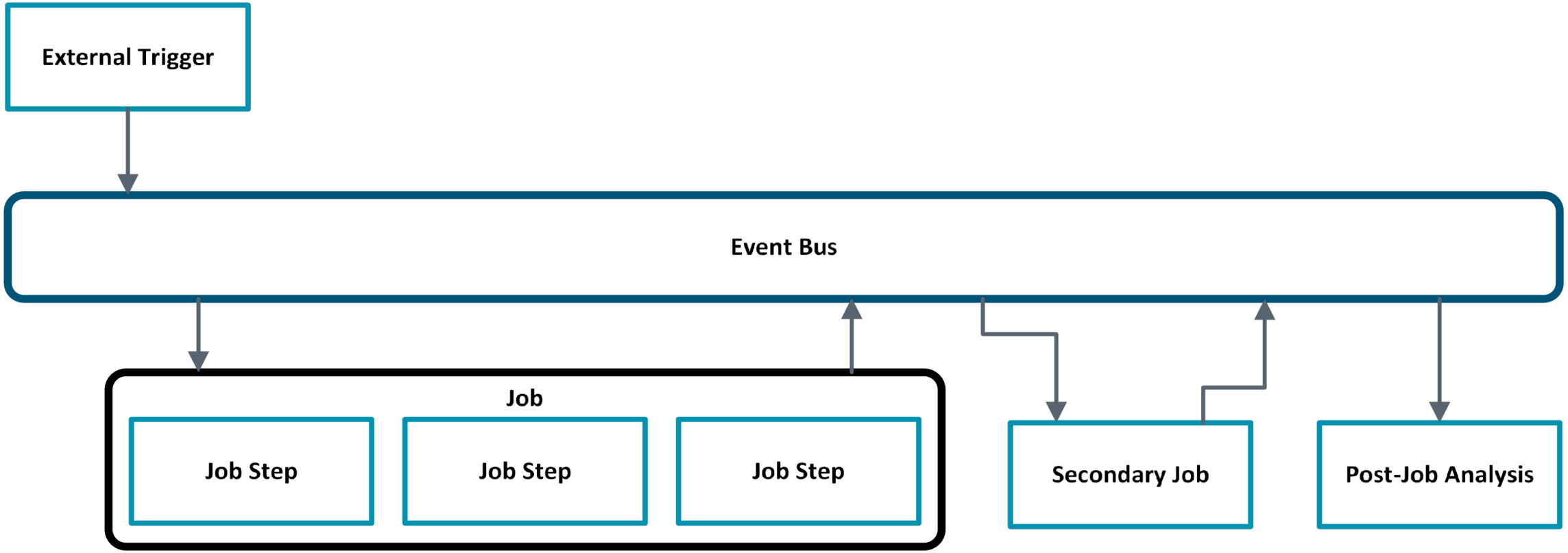
In a parallel world, asynchronous event-driven architectures are extremely common

- Individual components respond to events on an event bus
- No long-running shepherding process



CAN WE HAVE THE BEST OF BOTH WORLDS?

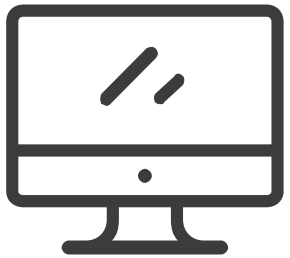
Are there things to gain from an event-driven architecture for HPC jobs?



HOW HARD COULD IT BE?

Collaboration between AWS HPC and Sandia's HAAPs team to prove out:

- Triggering a job with an external event
- Running that job on a standard Slurm cluster
- Triggering follow-up actions with job-end events
- The ability to expand this to more complex workflows



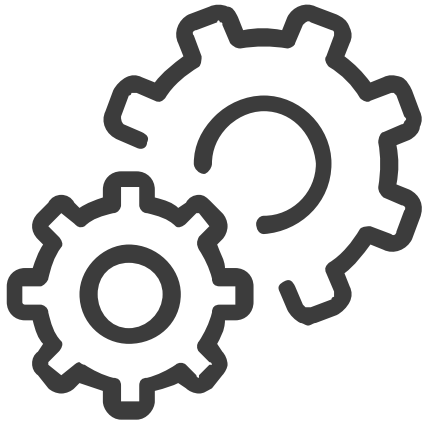
Initial implementation using AWS services

… but, aimed to be able to do this in a mixed environment or all on-prem too

EXISTING WORK IN THIS AREA

Existing projects helped inspire this work, including:

- Event-driven weather forecasting
  - https://github.com/aws-samples/event-driven-weather-forecasts
  - Based on a predefined set of steps to orchestrate the workflow
- MPI-based jobs in AWS Batch
  - https://github.com/aws-samples/batch-mpi-examples/tree/main
  - A collection of examples of running batch jobs on cloud resources
- The Slurm REST API
  - https://slurm.schedmd.com/rest.html
  - Makes Slurm-as-a-Service more feasible

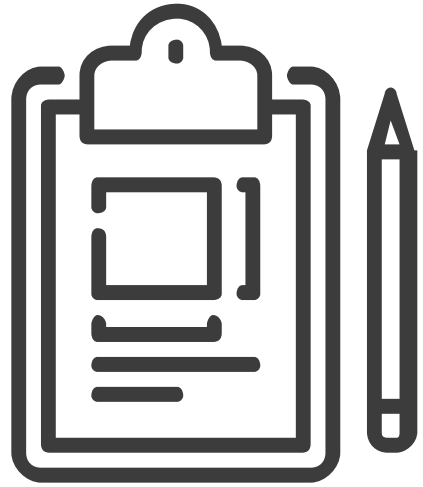


Our approach is to create a dynamic, event-driven workflow that combines the strengths of HPC and Cloud services

WHAT DO WE NEED?

As a reminder, we want to:

- Triggering a job with an external event
- Running that job on a standard Slurm cluster
- Triggering follow-up actions with job-end events
- The ability to expand this to more complex workflows



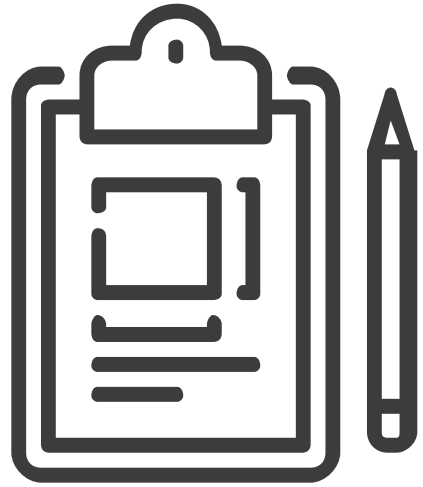
So, we need:

- An HPC cluster
- … that can start jobs via an API
- An event bus
- A way for events in that bus to trigger jobs
- A way to push an event into that bus that would start our job process
- A way to push an event into the bus when a job finishes
- A way for that event to trigger more actions after the job finishes

WHAT DO WE NEED?

As a reminder, we want to:

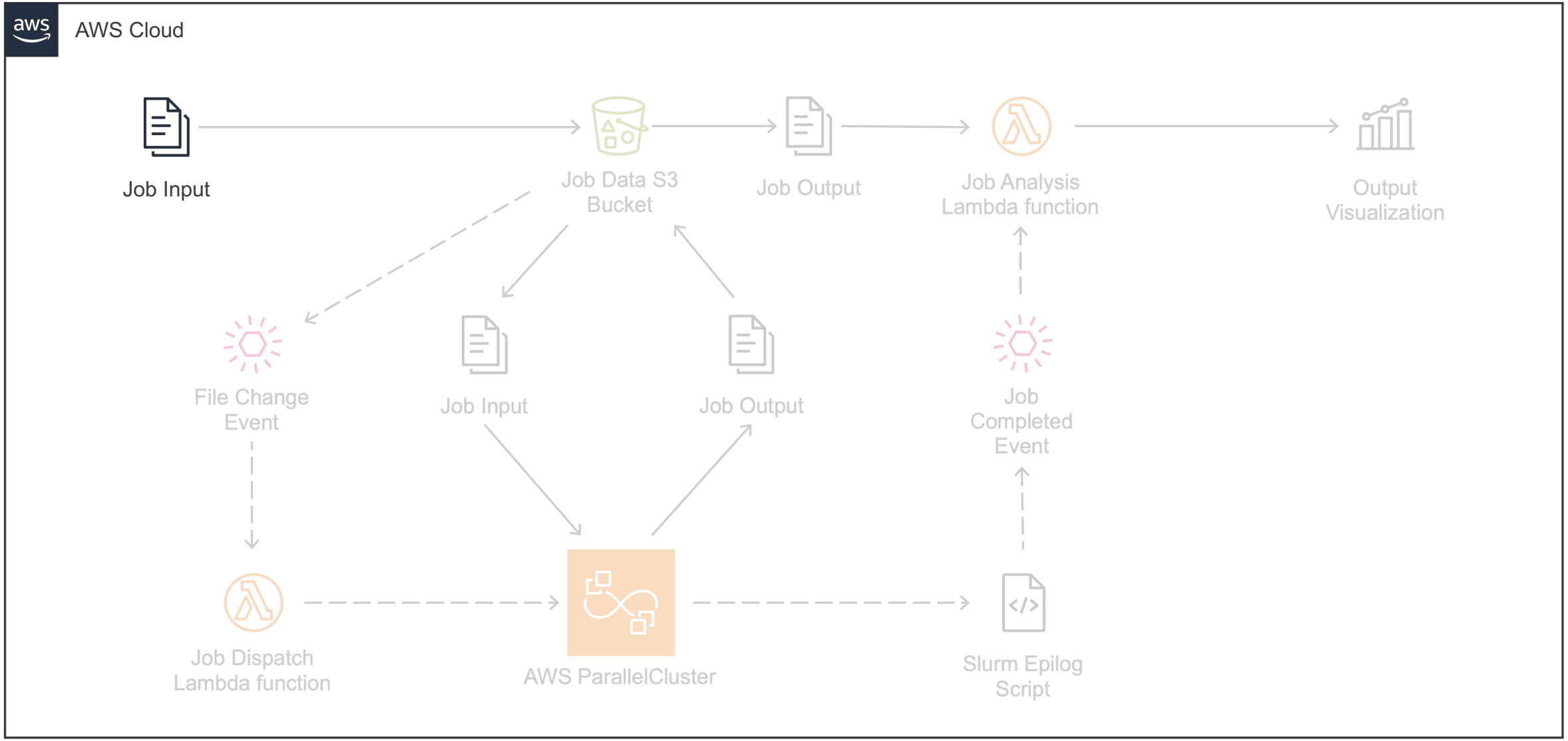
- Triggering a job with an external event
- Running that job on a standard Slurm cluster
- Triggering follow-up actions with job-end events
- The ability to expand this to more complex workflows



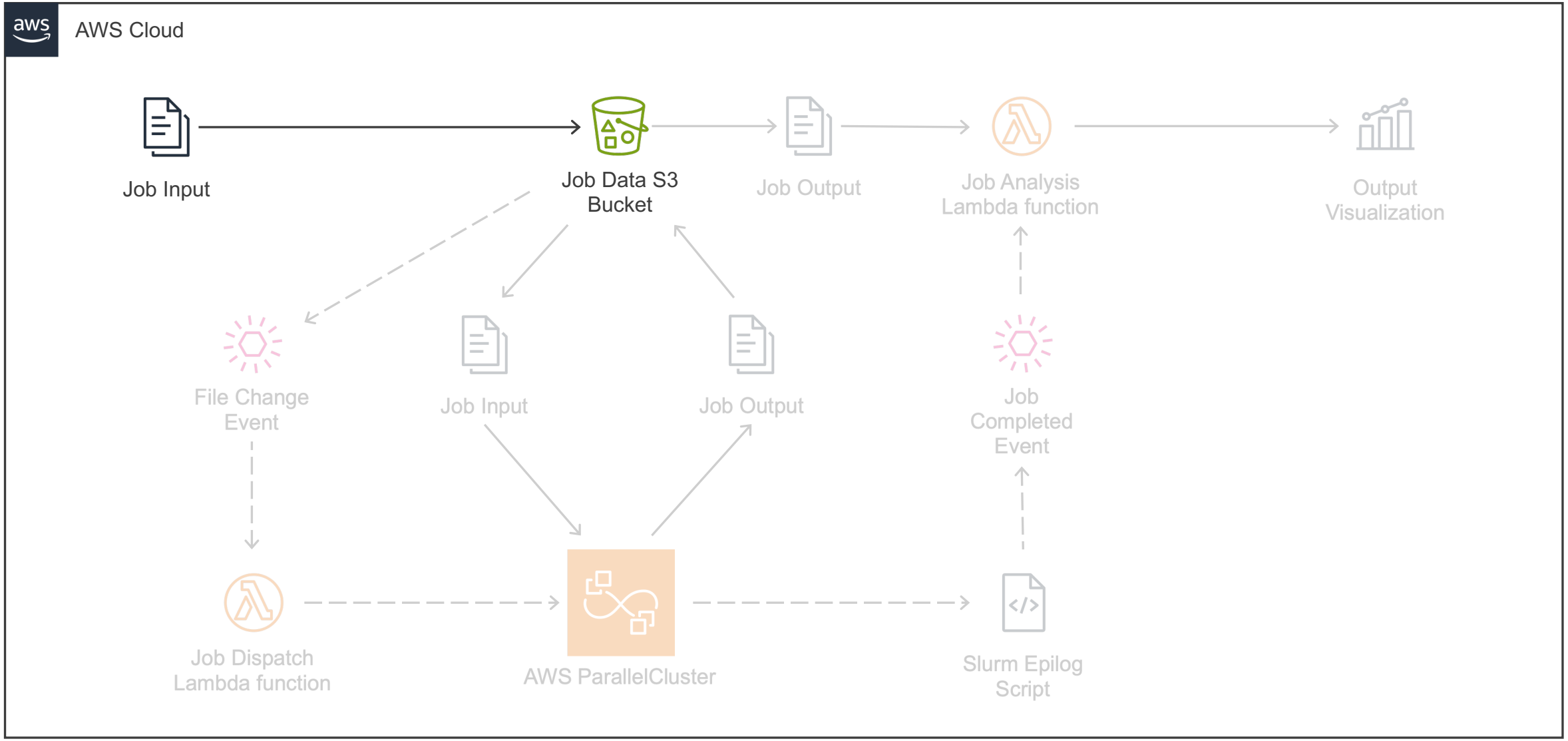
So, we need:

- An HPC cluster ➔ *ParallelCluster or existing on-prem cluster*
- … that can start jobs via an API ➔ *Slurm's REST API*
- An event bus ➔ *EventBridge*
- A way for events in that bus to trigger jobs ➔ *Lambda Function*
- A way to push an event into that bus that would start our job process ➔ *S3*
- A way to push an event into the bus when a job finishes ➔ *Slurm Epilog Script*
- A way for that event to trigger more actions after the job finishes ➔ *Lambda Function*

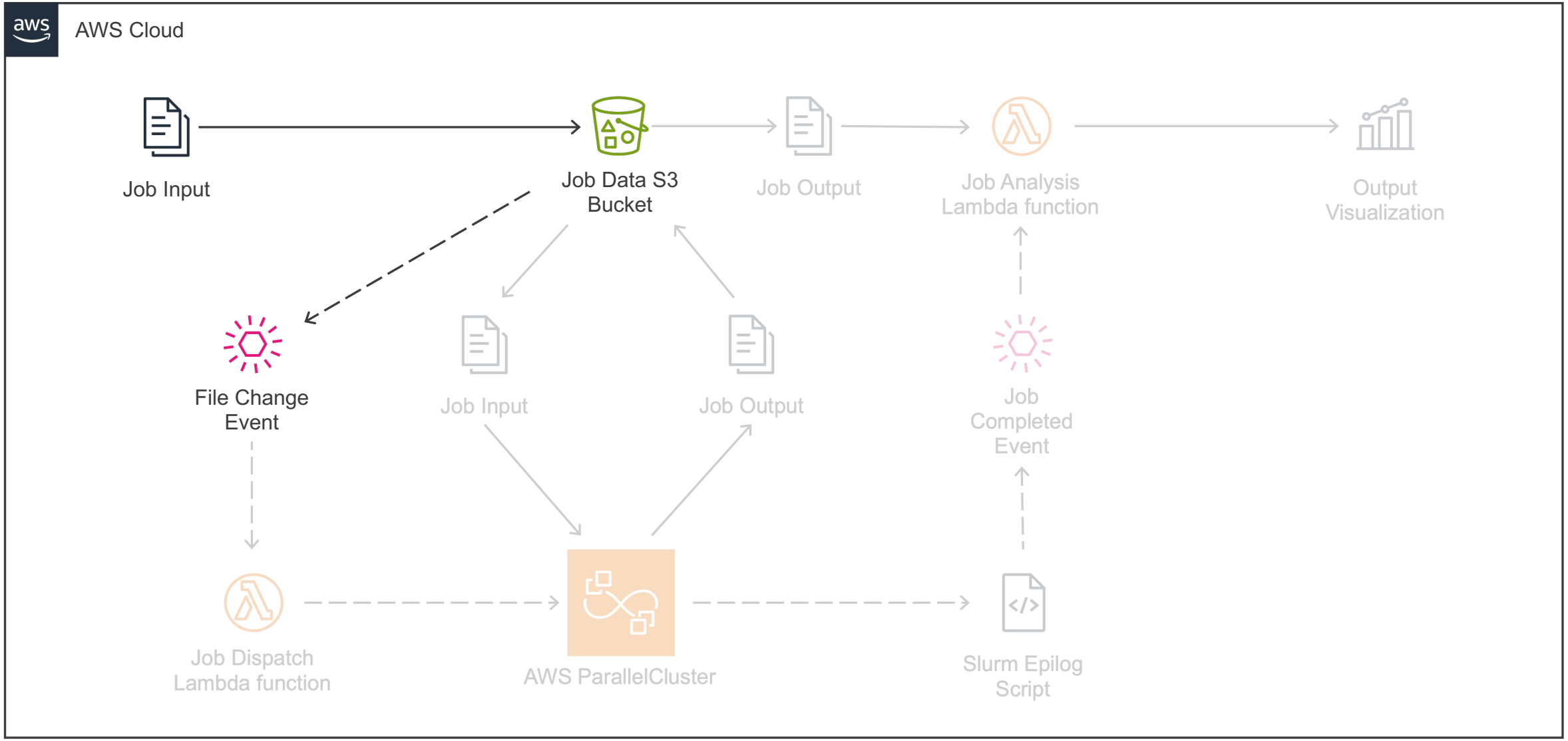
EVENT-DRIVEN HPC



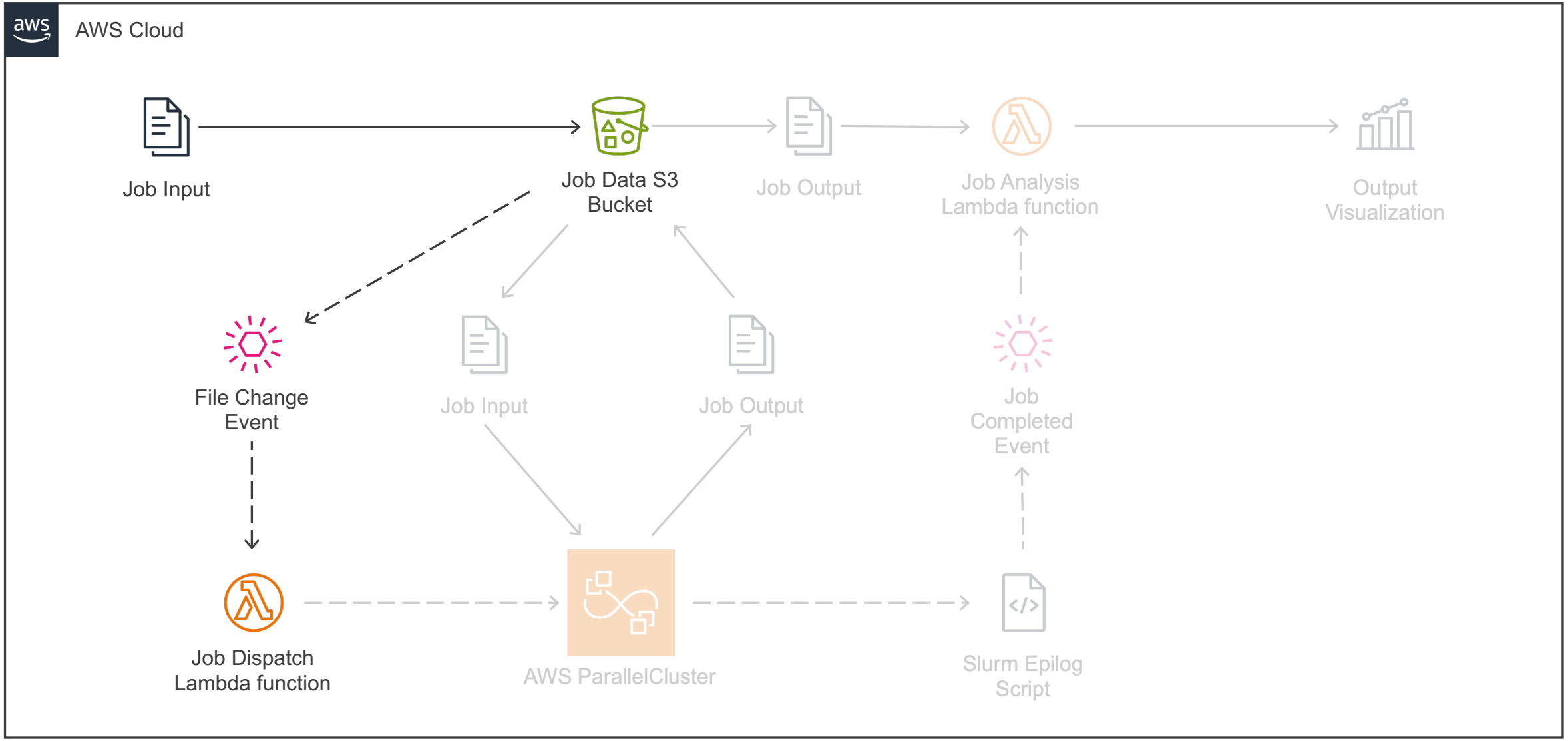
EVENT-DRIVEN HPC



EVENT-DRIVEN HPC



EVENT-DRIVEN HPC



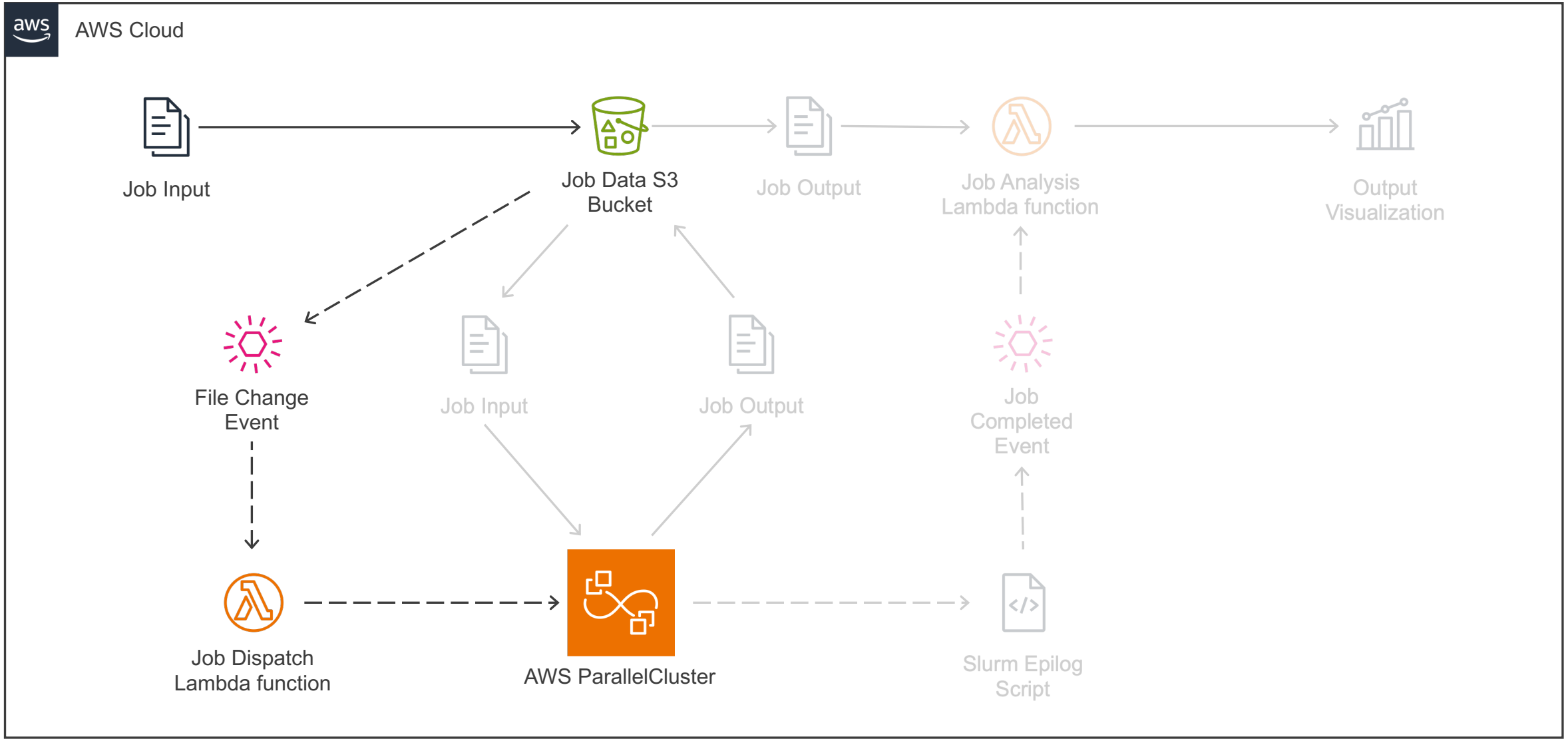
JOB SUBMISSION LAMBDA FUNCTION

- Job submitted via Slurm's REST API
- Lambda function written in Python
- Uses OpenAPI-generated Slurm library
- Library lives in a Lambda filesystem layer

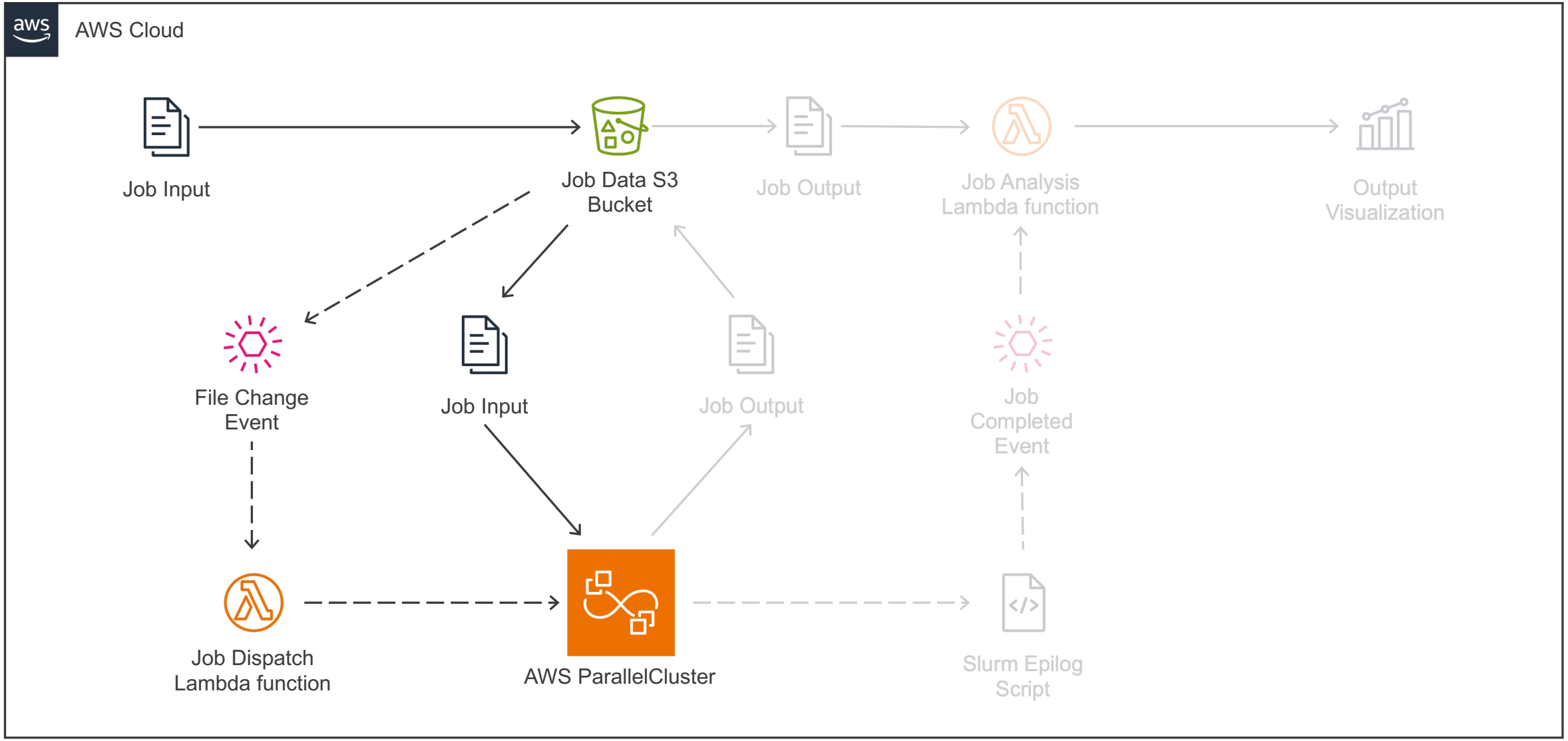
Job Dispatch Lambda function

```
import slurm

config = slurm.Configuration(...)

def lambda_handler(event, context):
  with slurm.ApiClient(config) as api_client:
    api_inst = slurm.SlurmApi(api_client)
    job_sub = slurm.V0039JobSubmission()
    job = slurm.V0039JobDescMsg(
      name = "eventdrivenjob",
      tasks = 4,
      script = "...",
      ...
    )
    job_sub.job = job
    resp = api_inst.slurm_v0039_submit_job(job_sub)

  return { 'response': str(resp) }
```

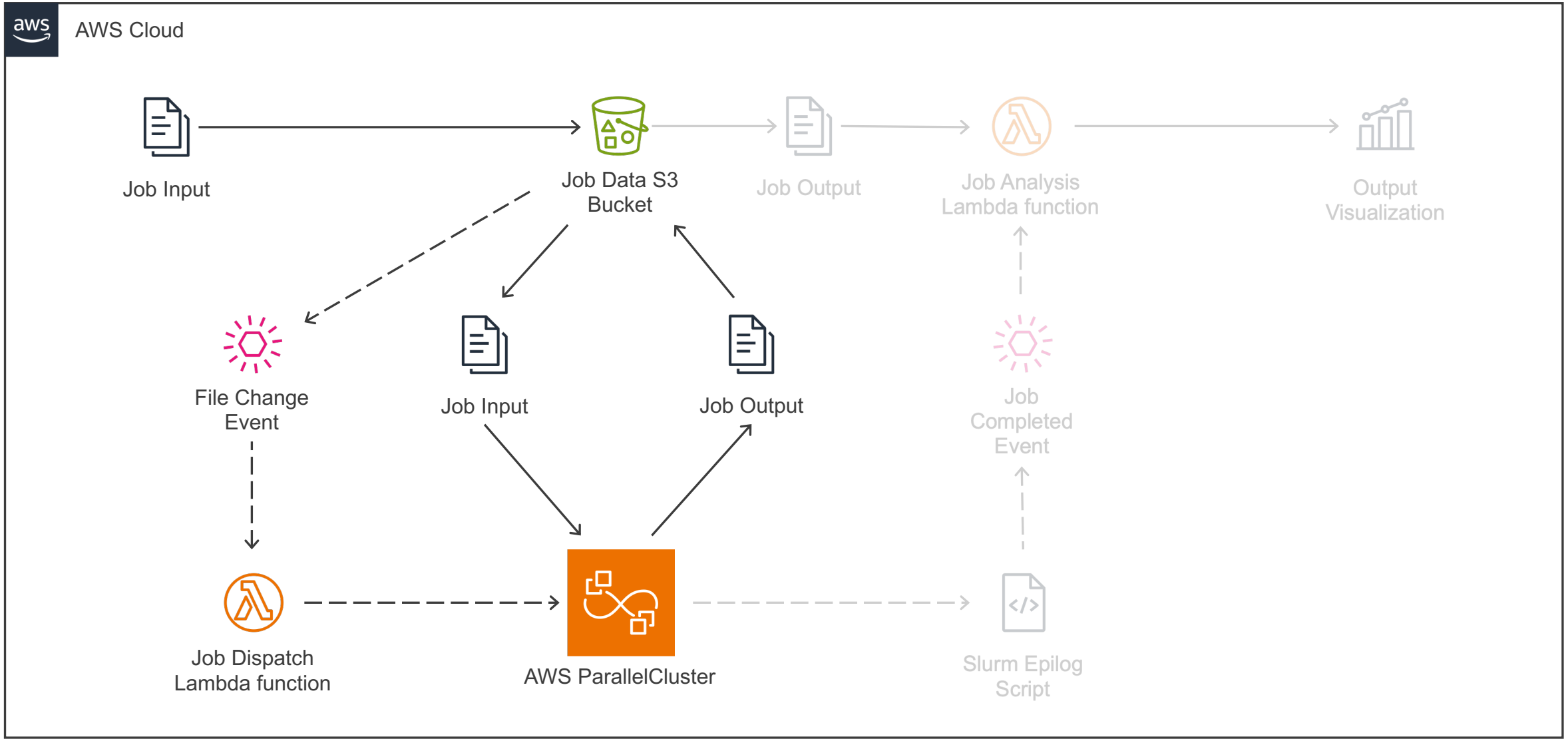
EVENT-DRIVEN HPC



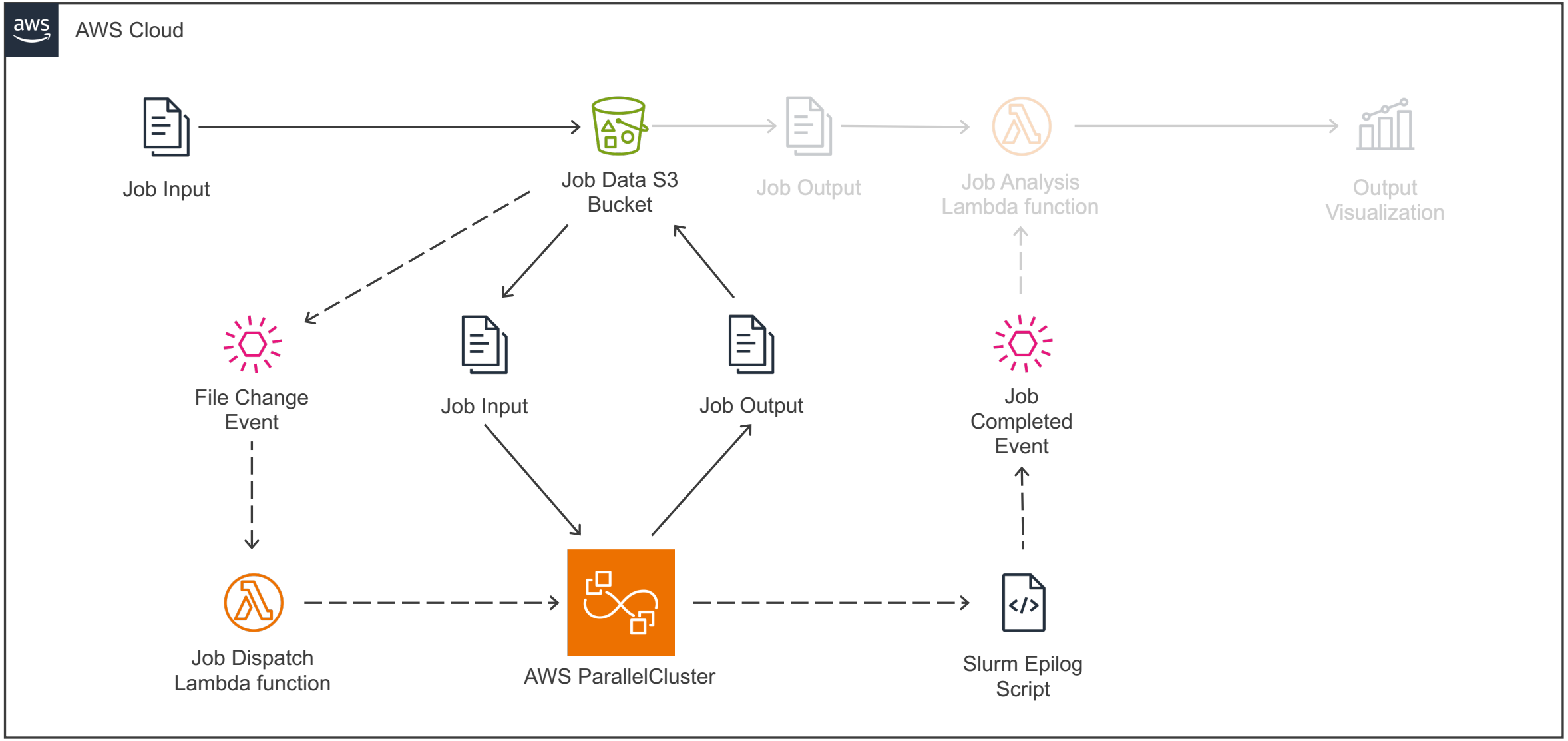
EVENT-DRIVEN HPC



EVENT-DRIVEN HPC



EVENT-DRIVEN HPC



JOB START AND END EVENTS

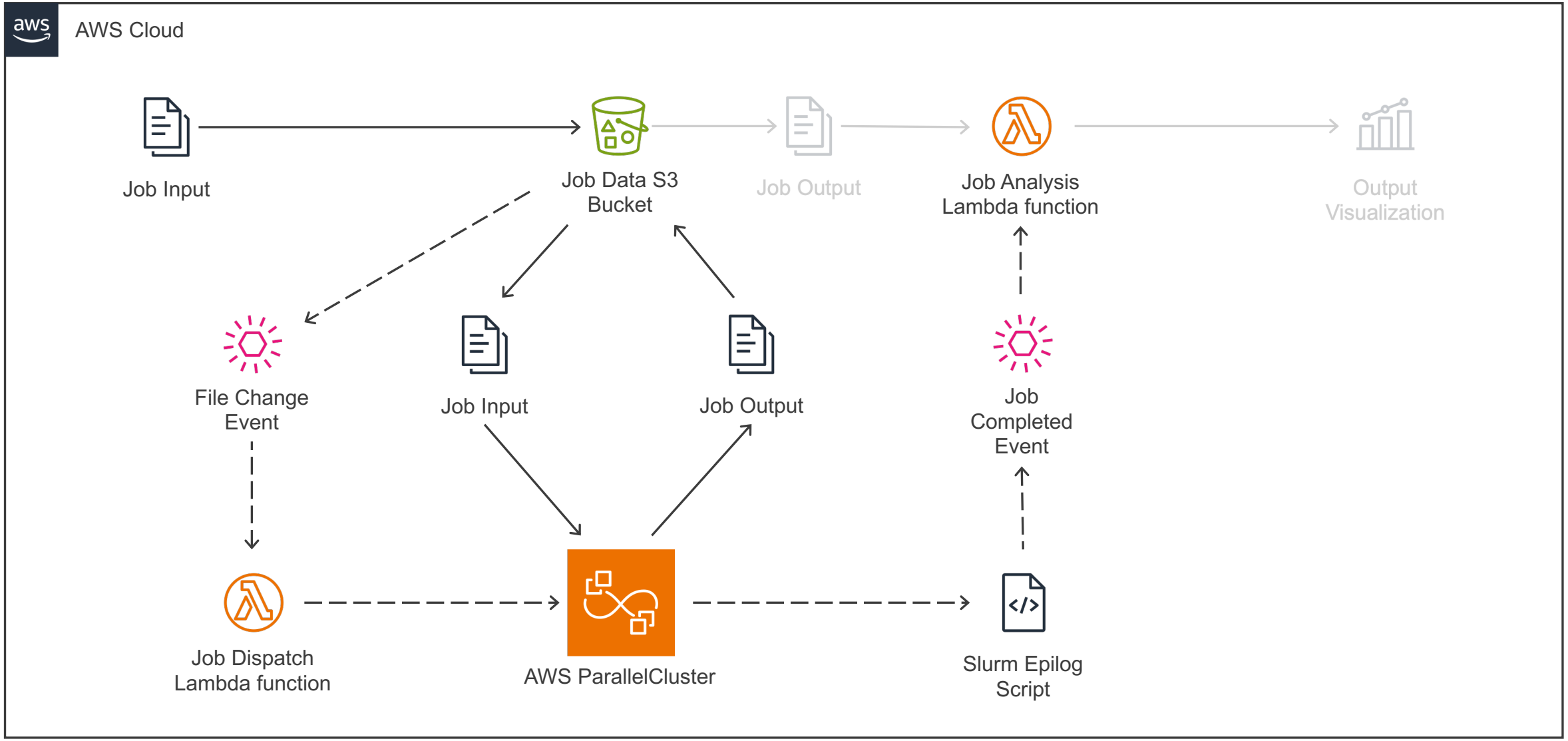
Prolog and Epilog scripts inject job start/end events into the message queue

- Specifically, PrologSlurmctld and EpilogSlurmctld scripts
- Events can contain arbitrary key/value pairs
- We picked a few that are useful for this work

Job Completed Event

```
{
  "Source": "pcluster.${SLURM_CLUSTER_NAME}",
  "Detail": {
    "SLURM_JOB_USER": "${SLURM_JOB_USER}",
    "SLURM_JOB_ID": "${SLURM_JOB_ID}",
    "SLURM_JOB_NAME": "${SLURM_JOB_NAME}"
  },
  "DetailType": "pcluster_job_end"
}
```

EVENT-DRIVEN HPC



JOB ANALYSIS LAMBDA FUNCTION

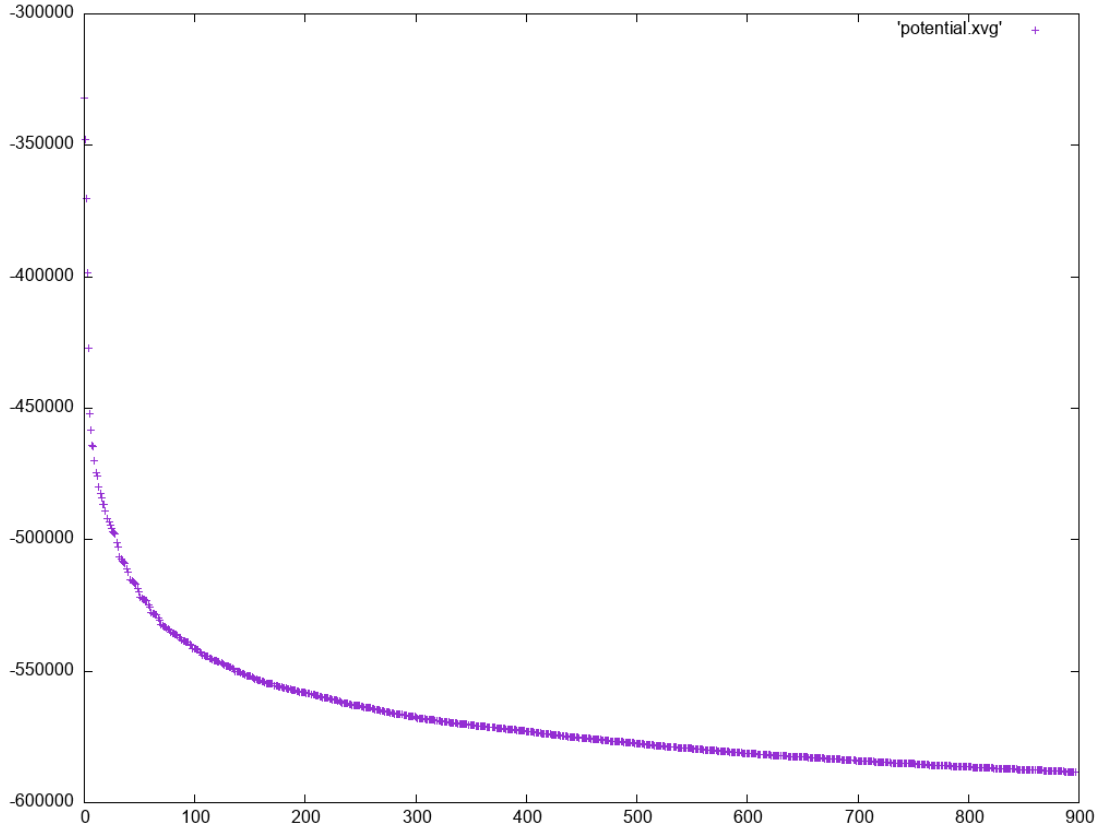
Triggered by job end event

Plots job data

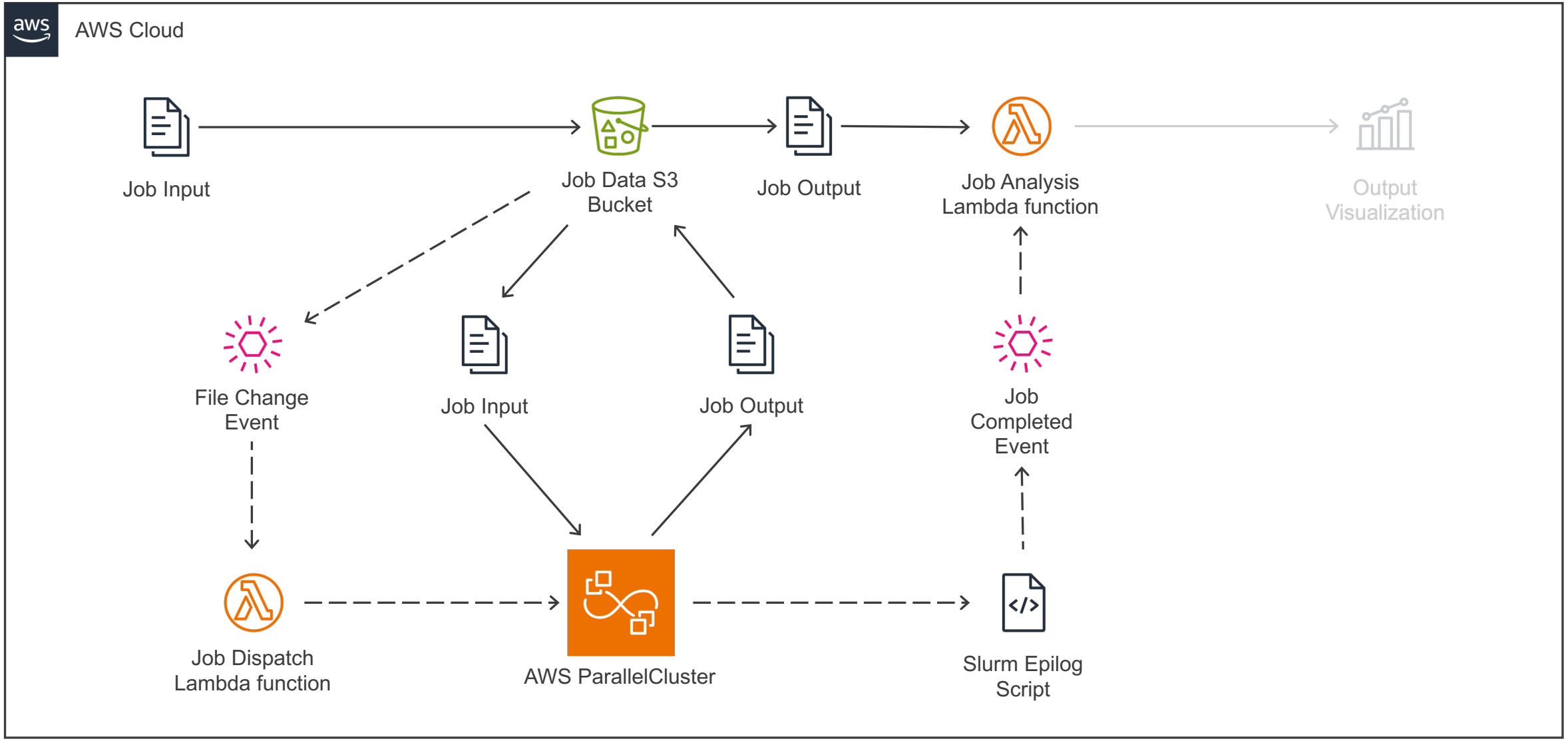
- Pulls job output from S3
- Generates graphs using gnuplot
- Pushes graphs back into S3

Can potentially do more sophisticated analysis

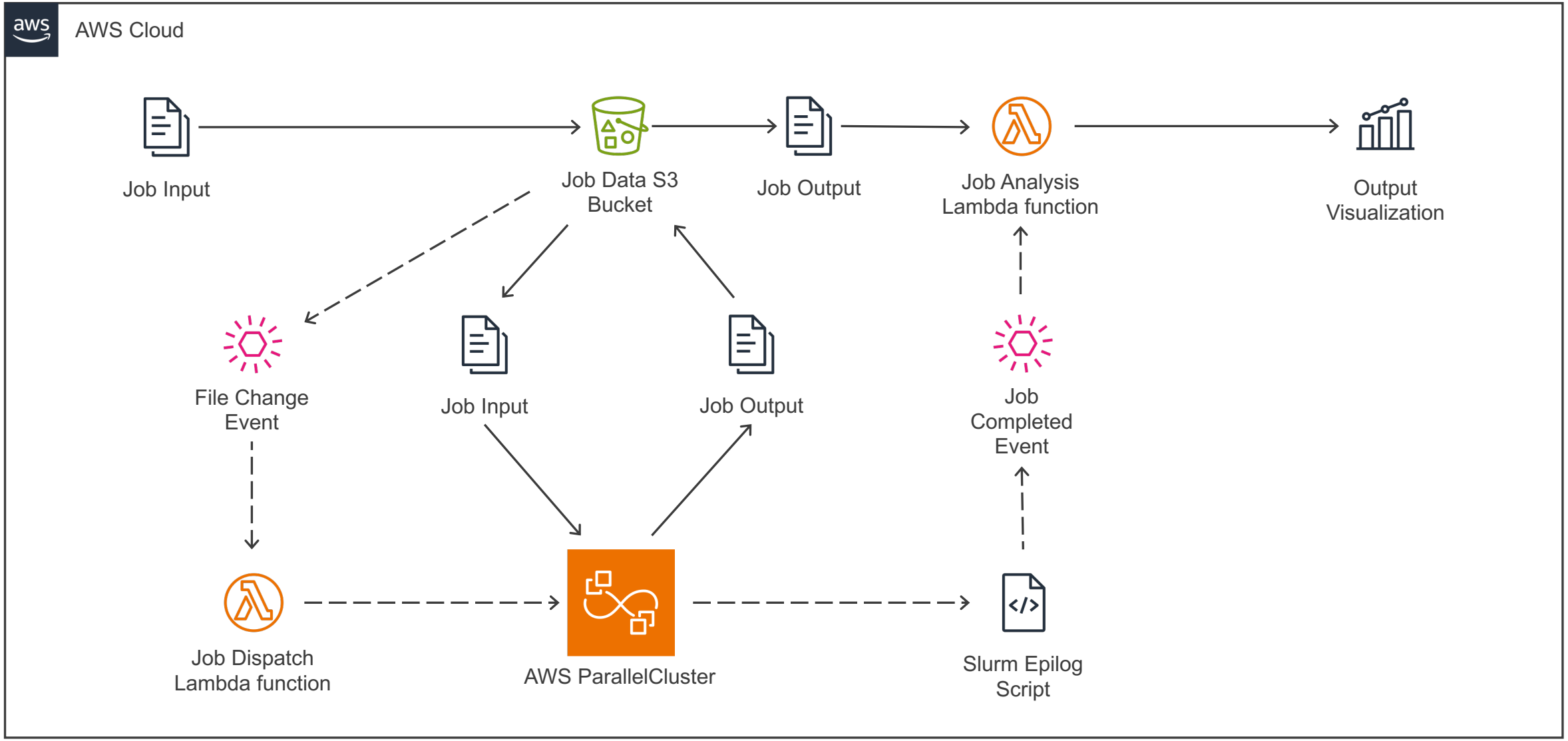
Job Analysis Lambda function



EVENT-DRIVEN HPC



EVENT-DRIVEN HPC



SO, HOW HARD COULD IT BE?

Not that bad!

- Slurm REST API made cluster-as-a-service possible
- Prototyping on AWS was quick and straightforward
- Building in a hybrid on/off-prem environment is very doable
- Building totally on-prem would require several other services, plus the people to run them

But…

- but we still did a lot of relatively manual things, like writing custom lambda scripts
- It’s clear that a general-purpose event driven HPC architecture is a bigger undertaking

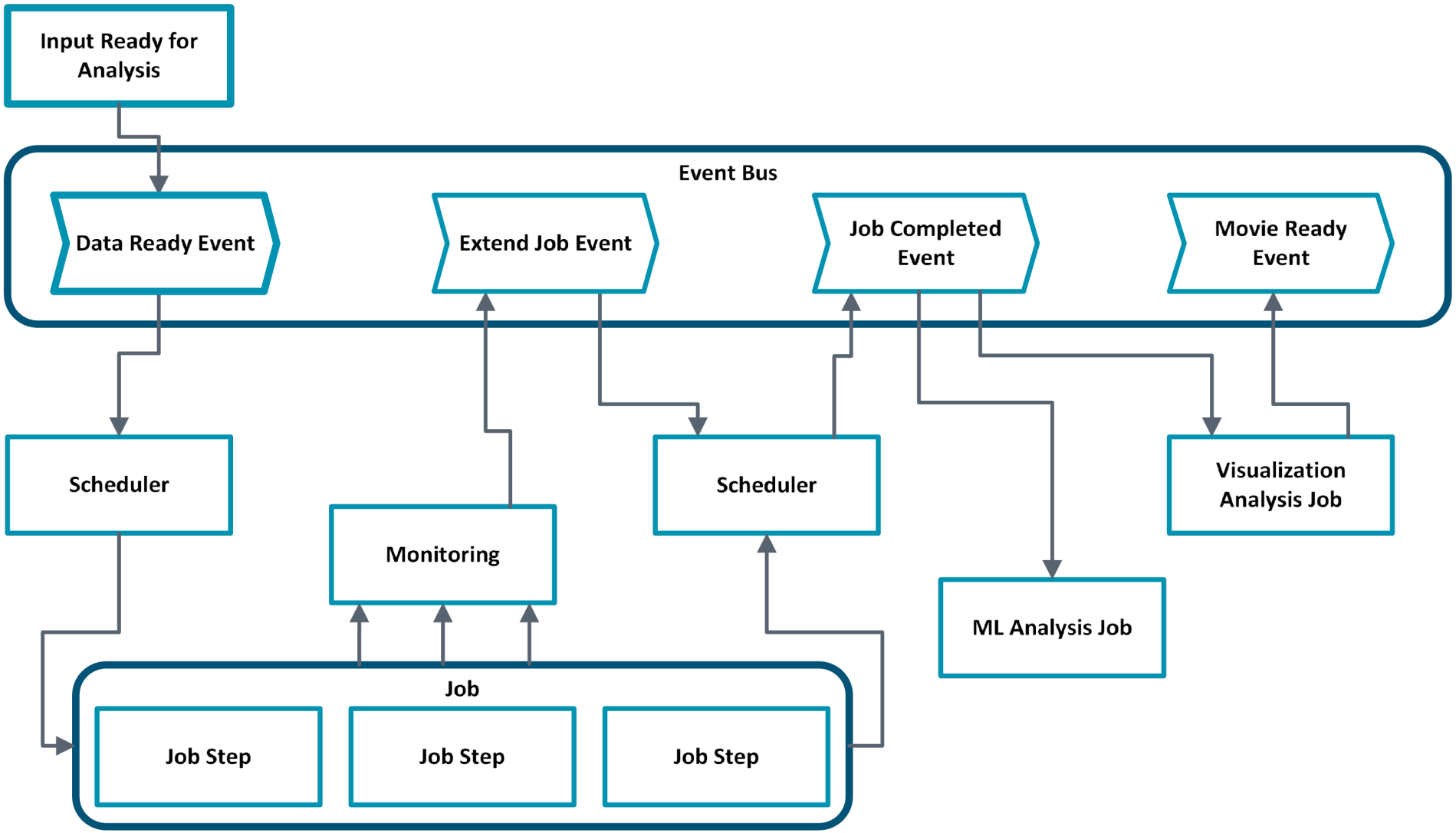
There are a lot of other event-driven actions that could be taken over time

FUTURE WORK

There are a number of future directions we can go with this

- This prototype was built all in the cloud; we want to make it hybrid (hpc stuff local, control stuff in the cloud)
- Work with a application team to apply this approach to a real workflow
- Generalize the process to make it usable by multiple applications
- Embed a start/end event generator into slurm, perhaps as a spank plugin
- Integrate monitoring and feedback loops into the job portion

WHAT IF ... ?


Input Ready for Analysis
Event Bus
Data Ready Event
Extend Job Event
Job Completed Event
Movie Ready Event
Scheduler
Monitoring
Scheduler
Visualization Analysis Job
ML Analysis Job
Job
Job Step
Job Step
Job Step

THANKS!

Sandia National Laboratories

aws