[bytesource]

Backups at Scale @ Bitpanda

A story by Jörg Herzinger and Daniel Linhart

Objectives

- Standardisation of Bitpandas database snapshot solution
- Snapshots should be copied to a secure account outside of our AWS Organization and to a different region
- The solution should deal with databases of various sizes and criticality

History | Motivation

- When it comes to Backups, a little paranoia is good

Taking Snapshots of your Aurora Cluster

Copying your snapshots to another account

Copying your snapshots to another region

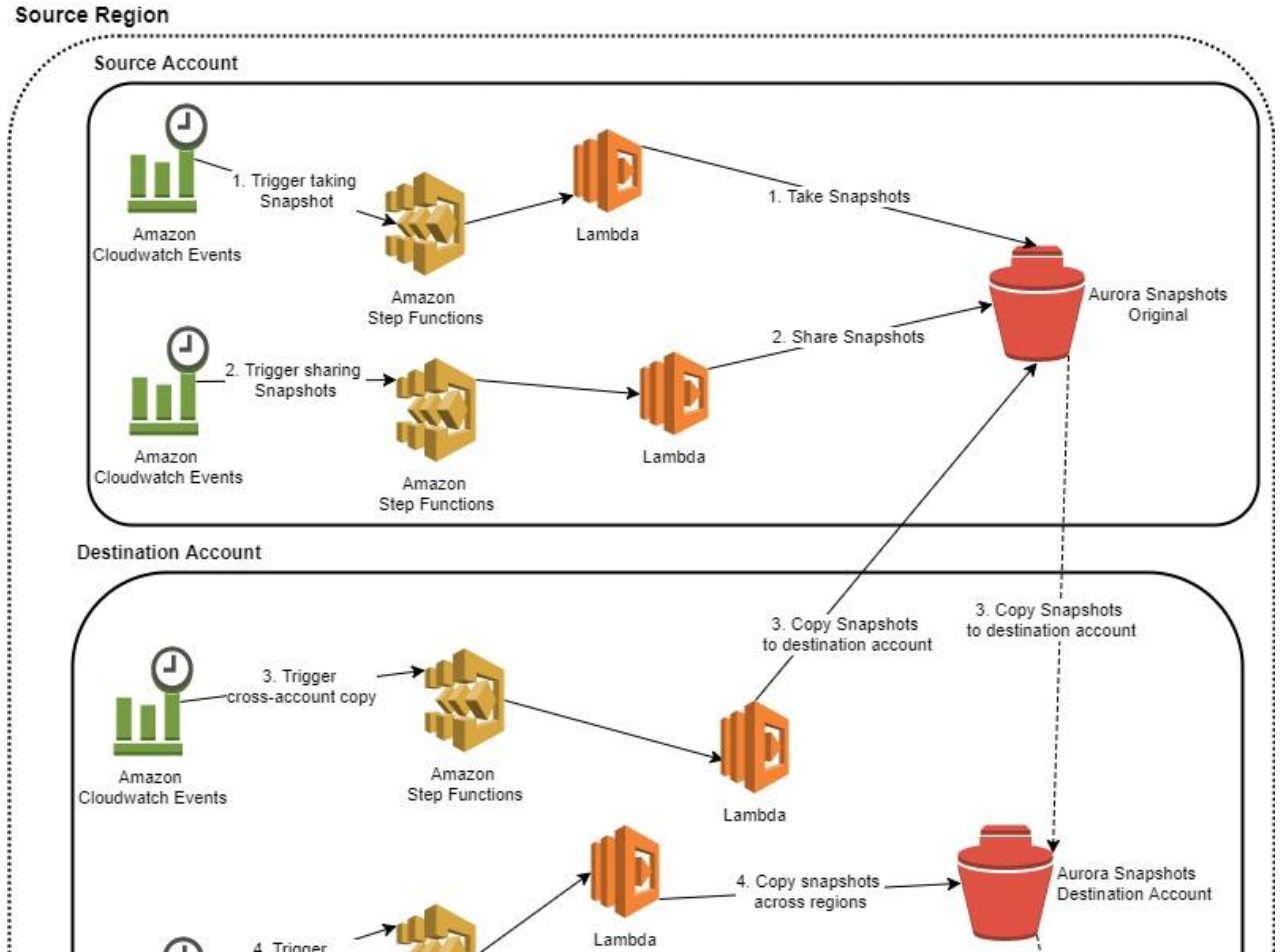
Copying your snapshots to another cloud provider

What does that mean

- A separate account that's "secure"
  - Not in Bitpandas AWS Organisation
  - Restricted access to the Account to only a few people (not accounts)
- Multi Region Backup
- Multi Cloud Backup
- Different Backup Schedules based on "Criticality"

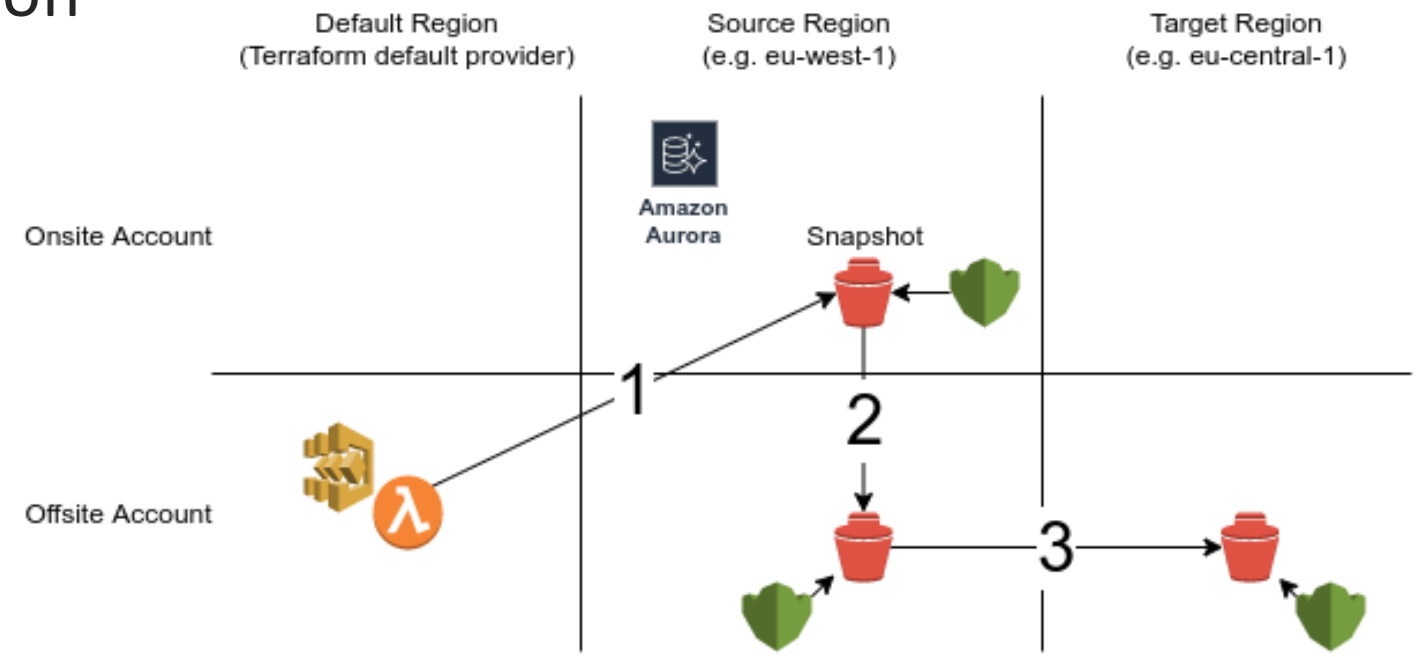
Initial idea and setup

- Reuse aurora-snapshot-tool available on GitHub from awslabs



Architecture

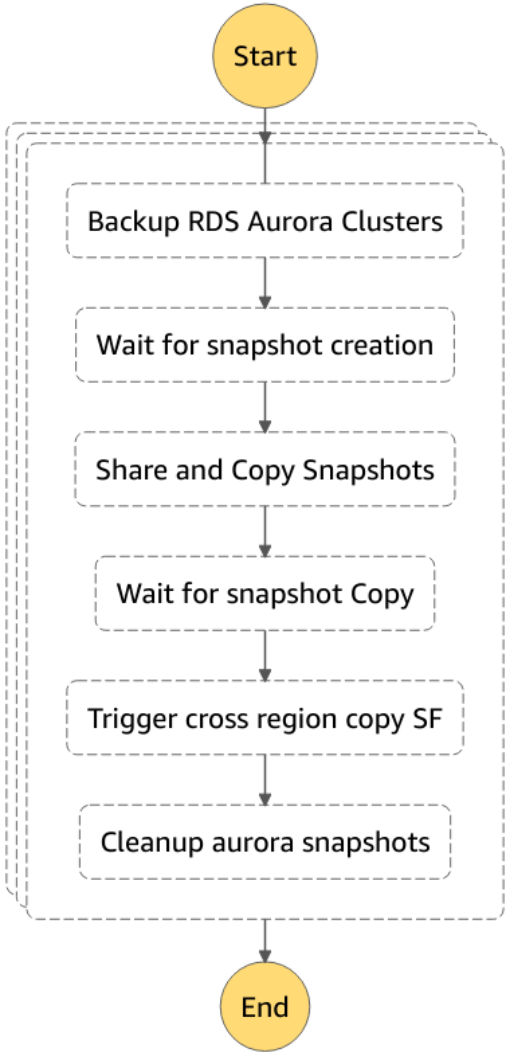
- Central step function controlling backup flow
- KMS per account and region
- Pull based
- First cross account copy
- Second cross region copy



Architecture

- Starting point simple step function

Don't wait in Lambda. Execution time costs money and has a 15-minute timeout!
Retry in the StepFunction. It's free of charge and retrys can be controlled.

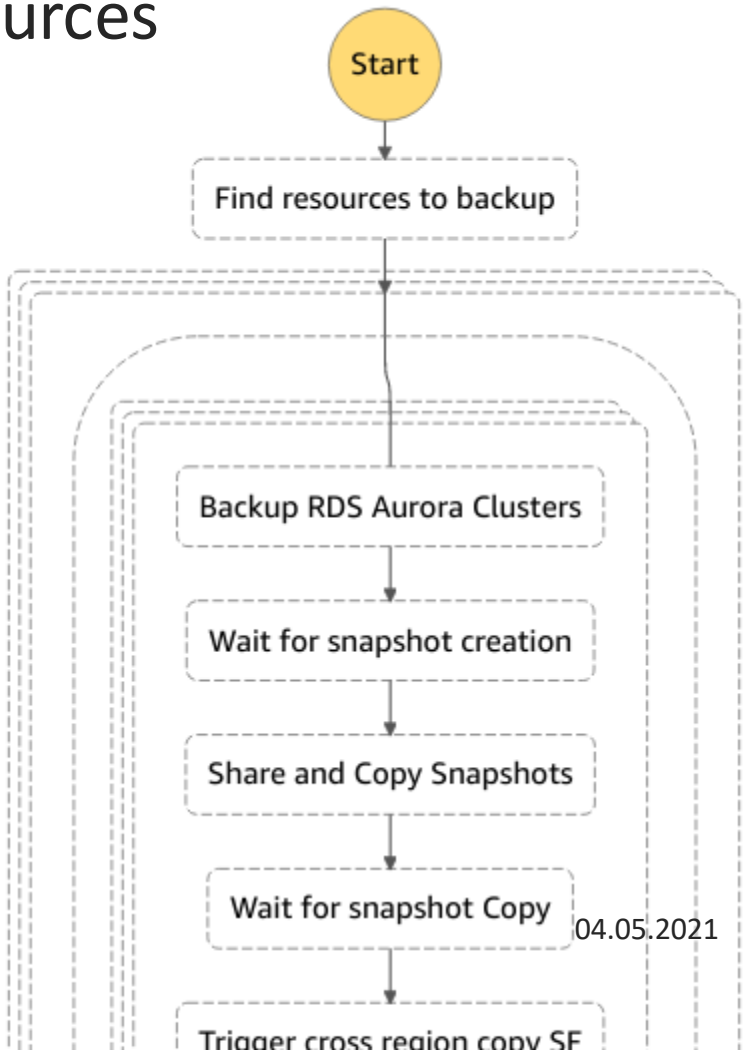


Architecture

- Extend by business logic that selects resources
- Tags are your Friends

Any operation breaks down to:
Assume Role -> Init Client -> Call function

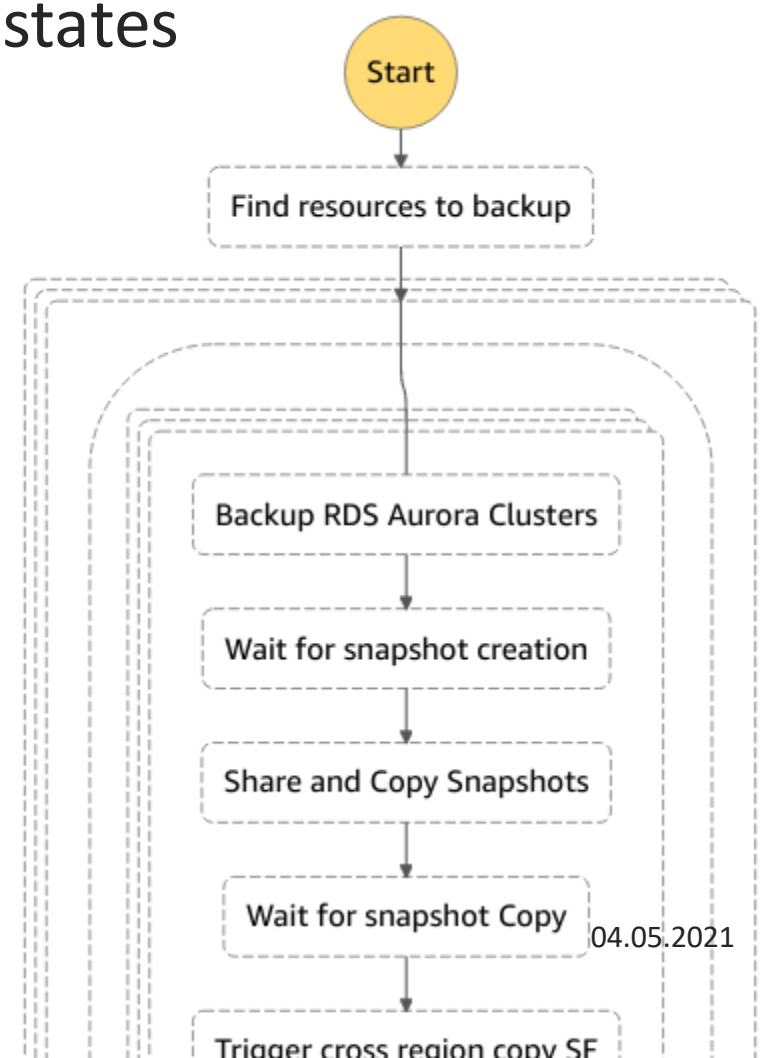
Find your abstraction for that flow and put it in a library



Architecture

- Parallelism built in by StepFunction map states

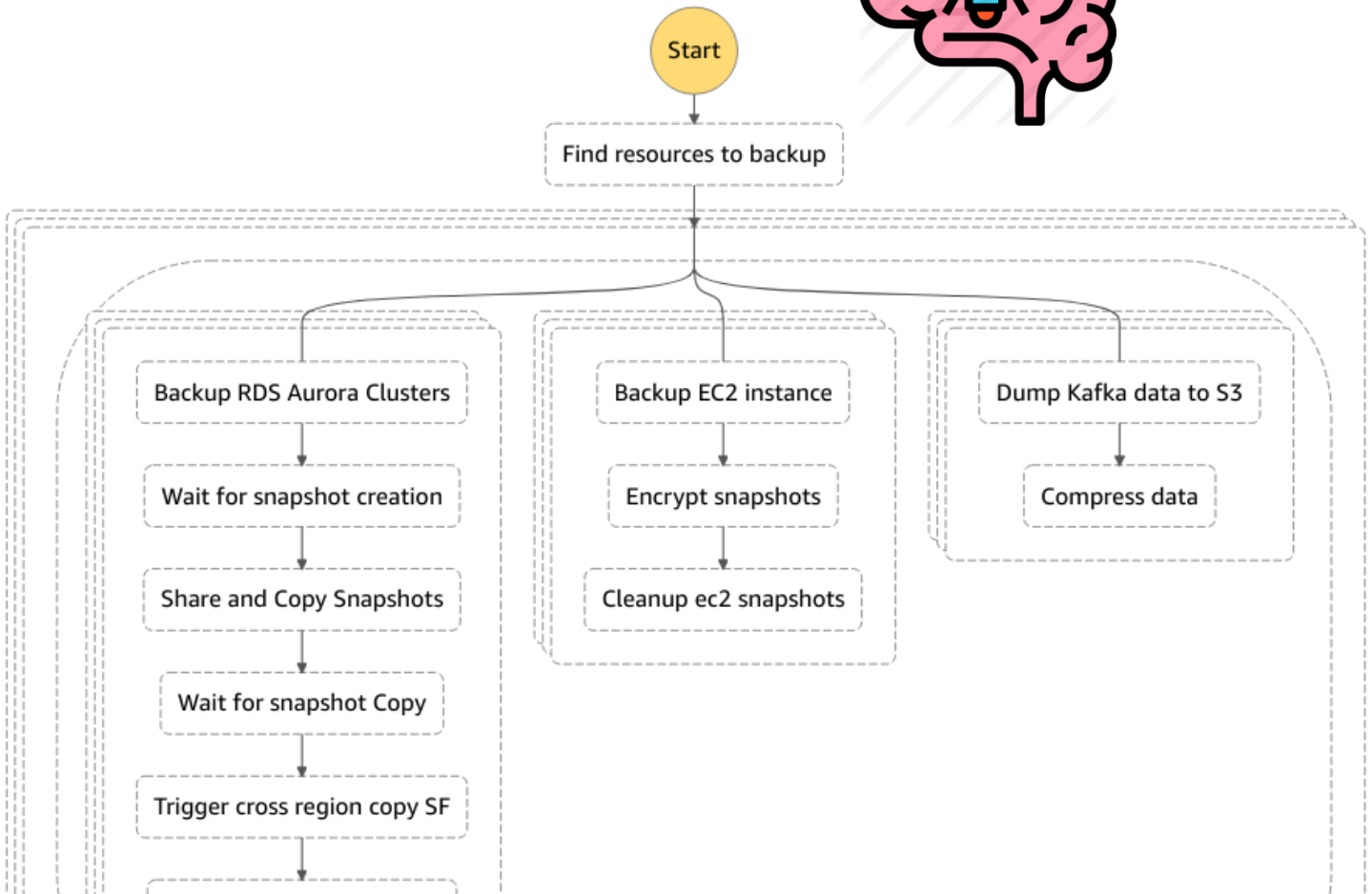
```
{
  "accounts": [
    {
      "account_id": "999999999999",
      "resources": {
        "aurora": [ "panda-cluster-1", "panda-cluster-2" ]
      }
    },
    {
      "account_id": "888888888888",
      "resources": {
        "aurora": [ "panda-cluster-3" ]
      }
    }
  ]
}
```



Architecture

- Plan for future extensions

Build your StepFunction dynamically in Terraform



Initial development flow

- Single Terraform repo for
  - Step function definitions
  - Lambda code
  - Terraform resources
- Deploy and test via "terraform plan && terraform apply"

Keep your initial development as fast as possible and worry about releases later. Ensure fast deployment cycles and allow all developers full access to an AWS dev account.

What we had to learn (hard way)

Service Limits

5 parallel Aurora snapshots per account and region maximum

Region Errors

5 parallel cross region copies maximum

Copy Timings

Inter region – few minutes

Cross region – up to 1 hour

Impossible Errors

"We currently do not have sufficient capacity in the region you requested. Our system will be working on provisioning additional capacity. You can avoid getting this error by temporarily reducing your request rate. (Service: AWSLambda; Status Code: 500; Error Code: ServiceException; Request ID: aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee; Proxy: null)"

[bytesource]

aws partner network

Advanced Consulting Partner

- Solution Provider
- Well Architected
- Immersion Day Partner
- Public Sector Partner

ATLASSIAN

Platinum Solution Partner
ENTERPRISE

Questions?

Contact us: contact@bytesource.net

[bytesource]

aws partner network

Advanced Consulting Partner

- Solution Provider
- Well Architected
- Immersion Day Partner
- Public Sector Partner

ATLASSIAN

Platinum Solution Partner
ENTERPRISE

Questions?

Contact us: contact@bytesource.net