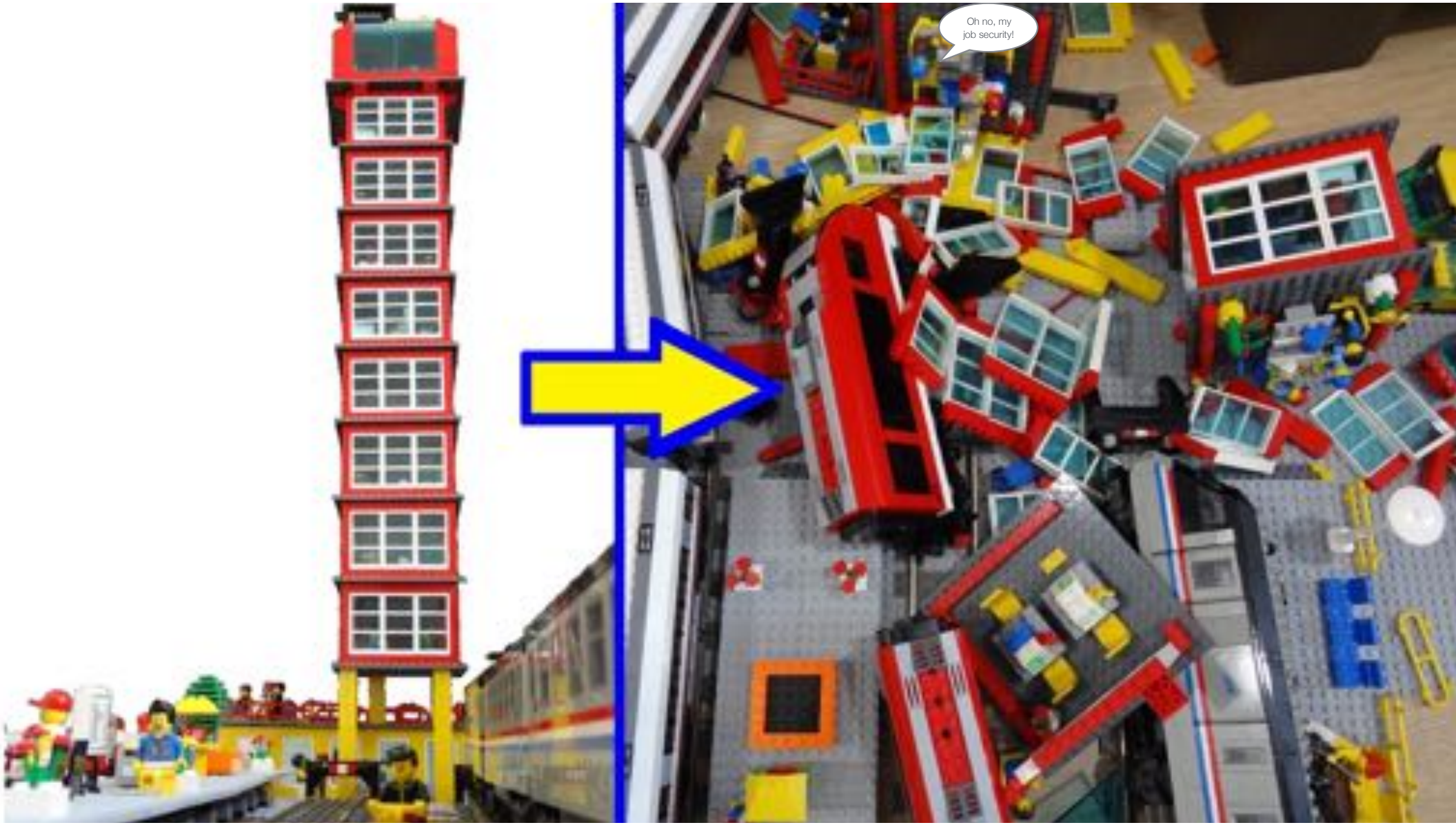
Level Up Your Web Apps with Amazon Web Services

Brian Klaas

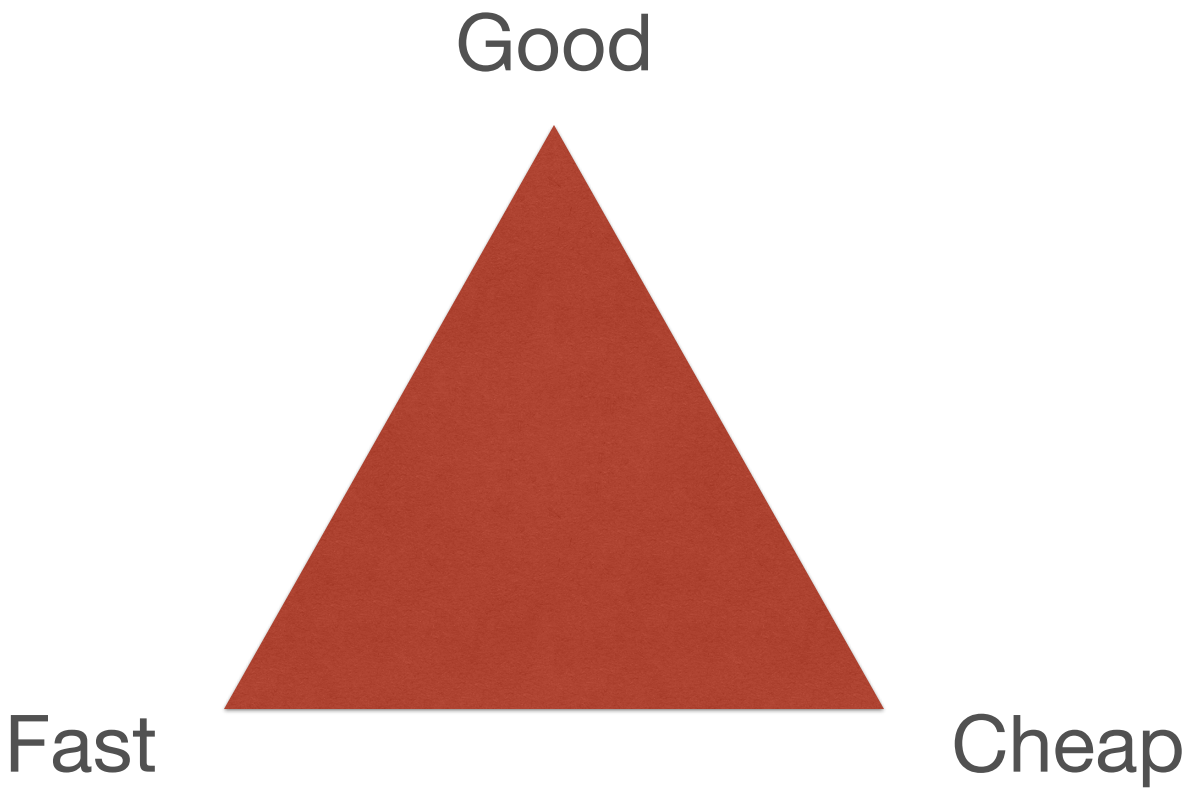
bklaas@jhu.edu

@brian_klaas

Hello

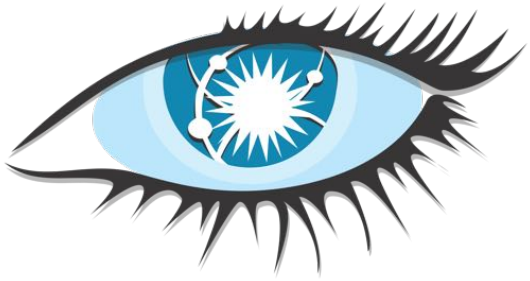

Oh no, my
job security!

Hello

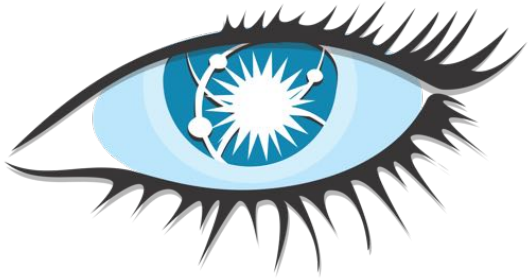


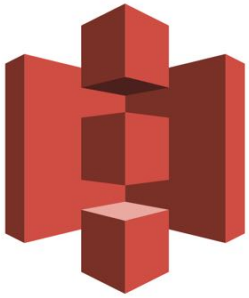
Hello

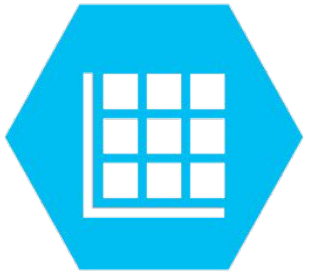
Hello



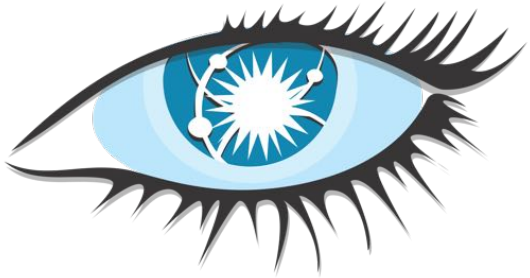
Hello

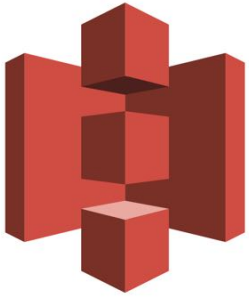


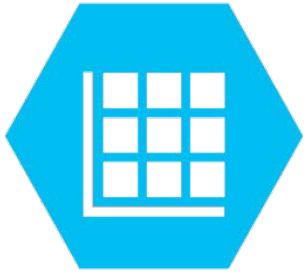




Hello







Hello

Hello


Cloud Front
Glacier
S3
Storage Gateway
Snowball
Dynamo DB
Elasti Cache
RDS
Elastic Map Reduce
Redshift
Kinesis
Aurora
Athena
Elasticsearch
Cloud Search
Elastic Transcoder
SES
SNS
SQS
SWF
FPS (Payment)
Mobile Analytics
Cognito
App Stream
WorkMail and Docs
Work Spaces
AWS Config
VPC
IAM
Coud HSM
Route 53
Direct Connect
Cloud Trail
Cloud Watch
Data Pipeline
Elastic Container Service
Batch
Lightsail
Xray
Code Build
Code Pipeline
EC2
Elastic Beanstalk
Lambda
Step Functions
API Gateway
Cloud Formation
OpsWorks
Code Commit
Code Deploy
Hello

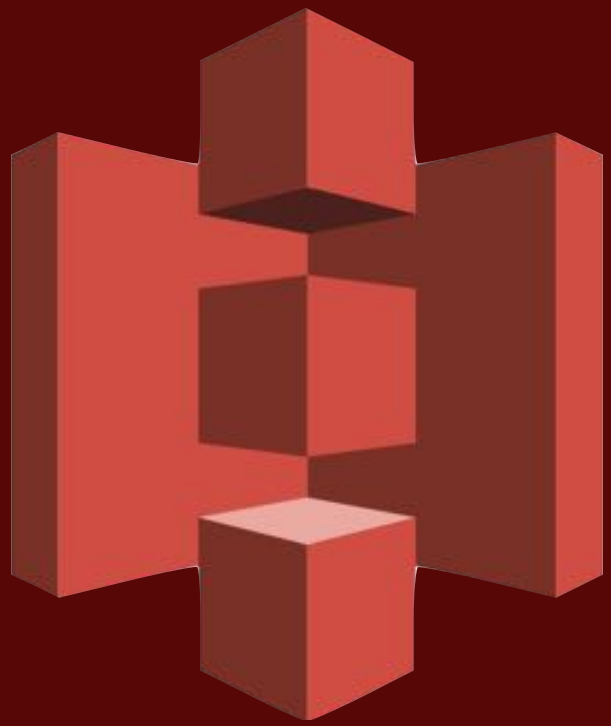
Hello

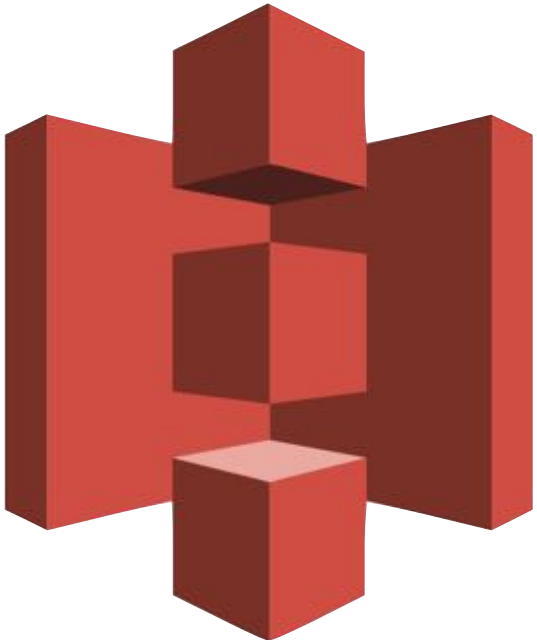
Things you have to do on your own:

1. Go play in the console
2. Learn about IAM roles and permissions

Hello

S3





Store all the things.

99.999999999% durability

$0.023 per GB stored

$0.004 per 10,000 GET

$0.005 per 1,000 PUT

$0.10 per GB out after 1GB

Bucket

myfiles.s3.amazonaws.com

http://mybucket.s3.amazonaws.com/
path/to/file.png

~~C:/~~

s3://

```
cffile( variable="fileData", file="s3://
somebucket/somefile.txt", action="read" );
```

```
cfdirectory( directory="s3://somebucket/
someDirectory", action="list" );
```

```
if ( !directoryExists("s3://
somebucket.s3.amazonaws.com") ) {
    perms = [ {group="all", permission="read"},
    {id="canonicalIDofYourAWSAccount",
     permission=“full_control"} ];
    cfdirectory( directory="s3://
somebucket.s3.amazonaws.com", storeacl=perms,
action="create" );
}

fileWrite("s3://somebucket.s3.amazonaws.com/
myFile.txt", "#someOutput#");

files = directoryList("s3://
somebucket.s3.amazonaws.com");
```

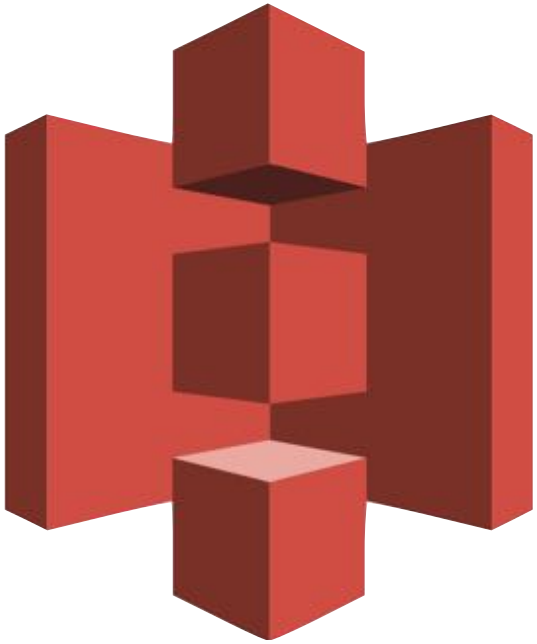
IAM Credentials Required

Inline:

```
cffile( variable="fileData", file="s3://
accessKey:secretKey@somebucket/somefile.txt",
action="read" );
```

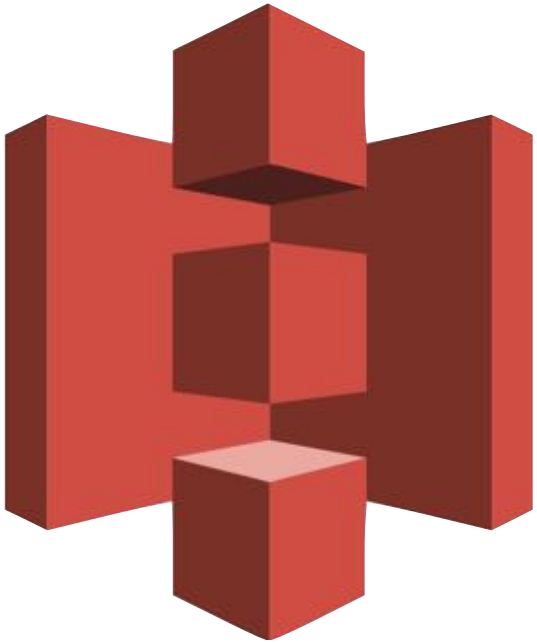
Or, in application.cfc:

```
this.s3.accessKeyId="accessKey";
this.s3.awsSecretKey="secretKey";
```



- Expire URLs
- Change properties on a per-request basis
- Upload to S3 from browser

Requires request signing.



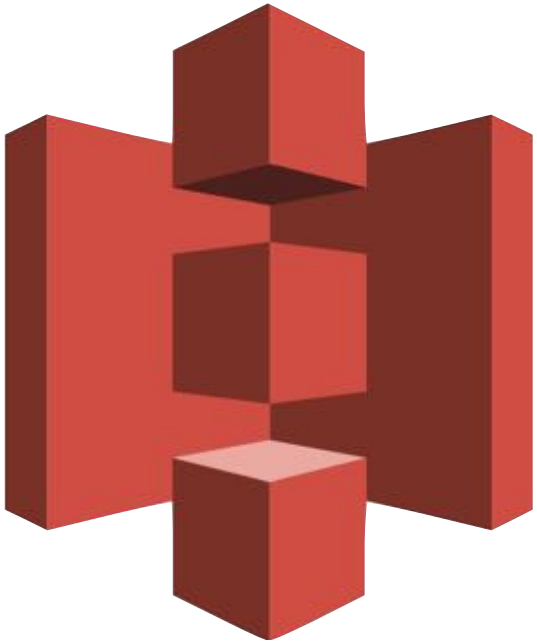
S3 Request Signing Utils

github.com/brianklaas/ctlS3utils

S3 is storage, not a file system.

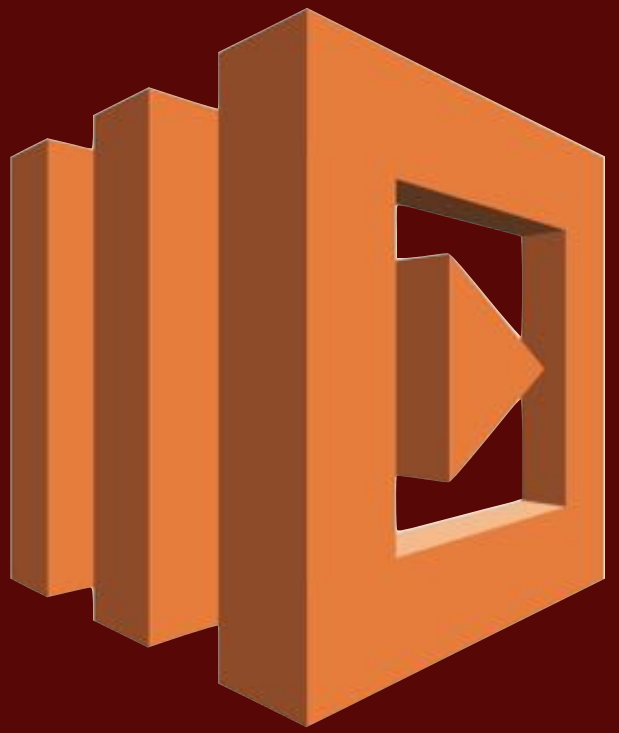
Can get basic file info with

```
cfhttp( url="http://bucket.s3.amazonaws.com/filename",
                    method="head" );
```

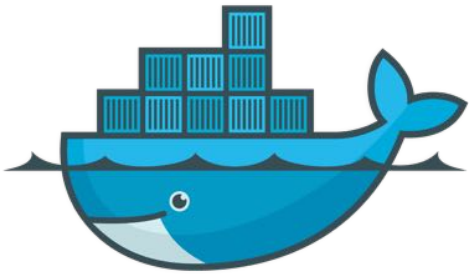


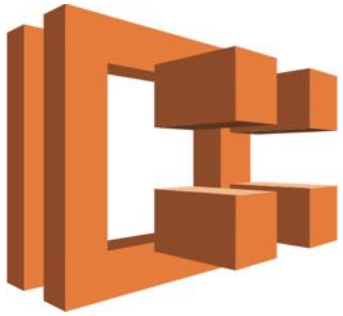
Store all the things.

Lambda



Lambda



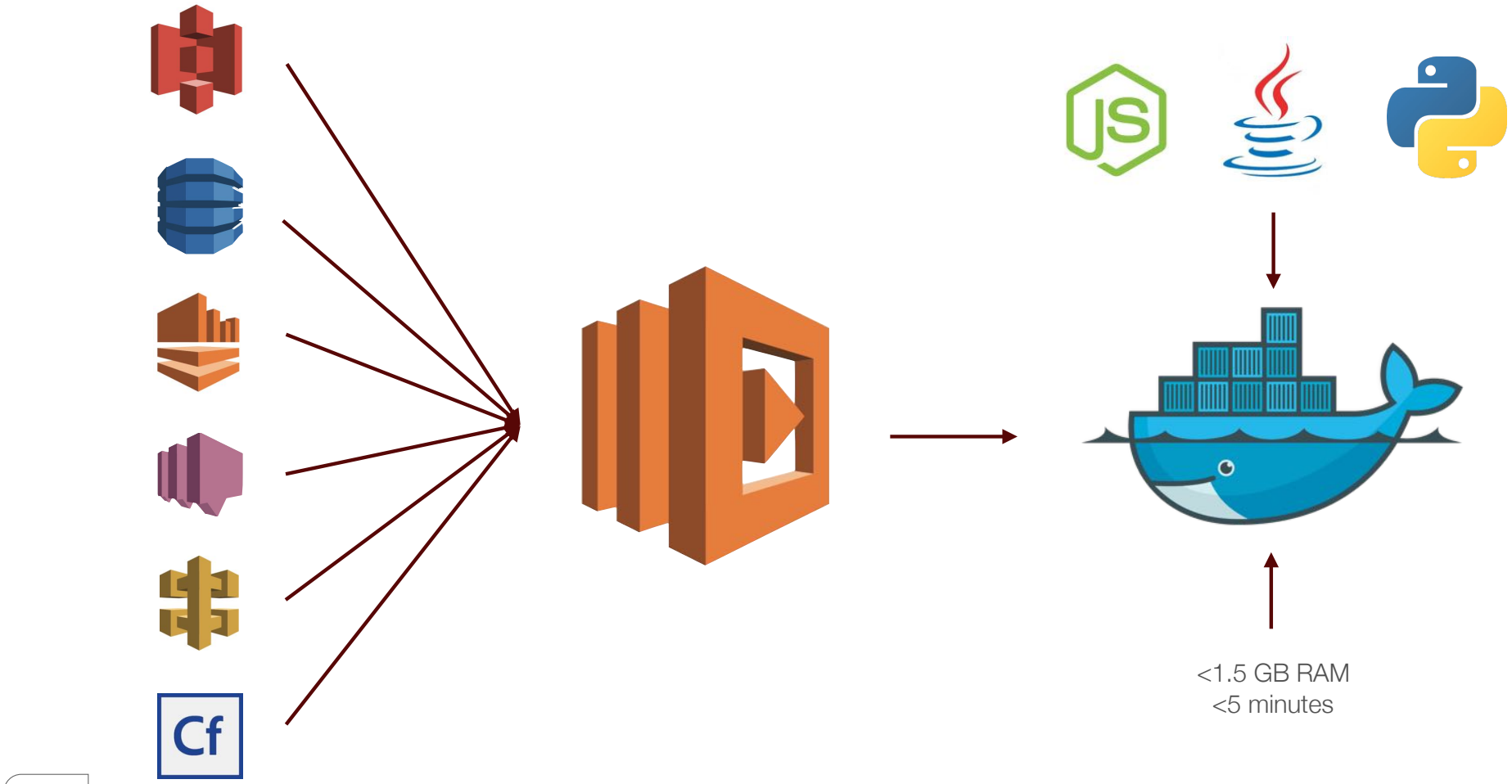


Lambda

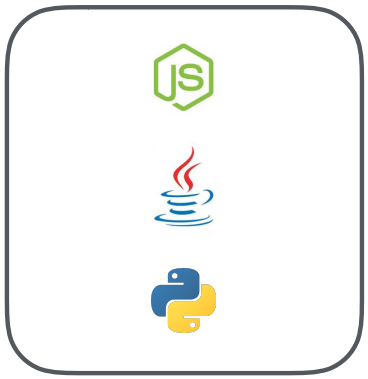
No server is easier to manage than no server.

= Event–driven computing

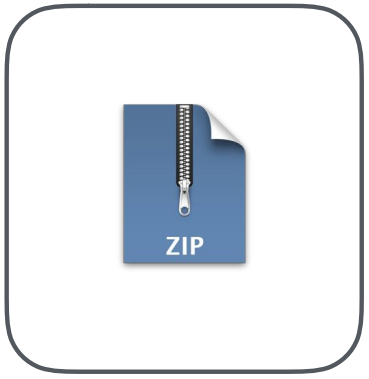
Lambda



Lambda

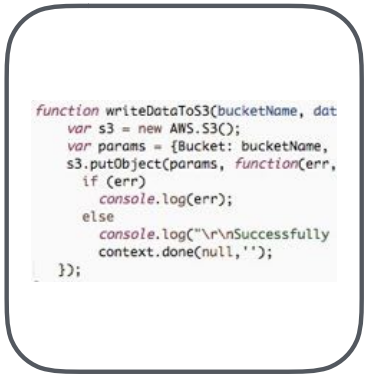


1. Write a handler function



2. Upload ZIP

3. Invoke an event



```
function writeDataToS3(bucketName, dat
    var s3 = new AWS.S3();
    var params = {Bucket: bucketName,
    s3.putObject(params, function(err,
      if (err)
        console.log(err);
      else
        console.log("\r\nSuccessfully
        context.done(null,'');
  });
```

4. Handler runs

Lambda

= Microservices infrastructure without having to worry about running containers or scaling your infrastructure!

Lambda

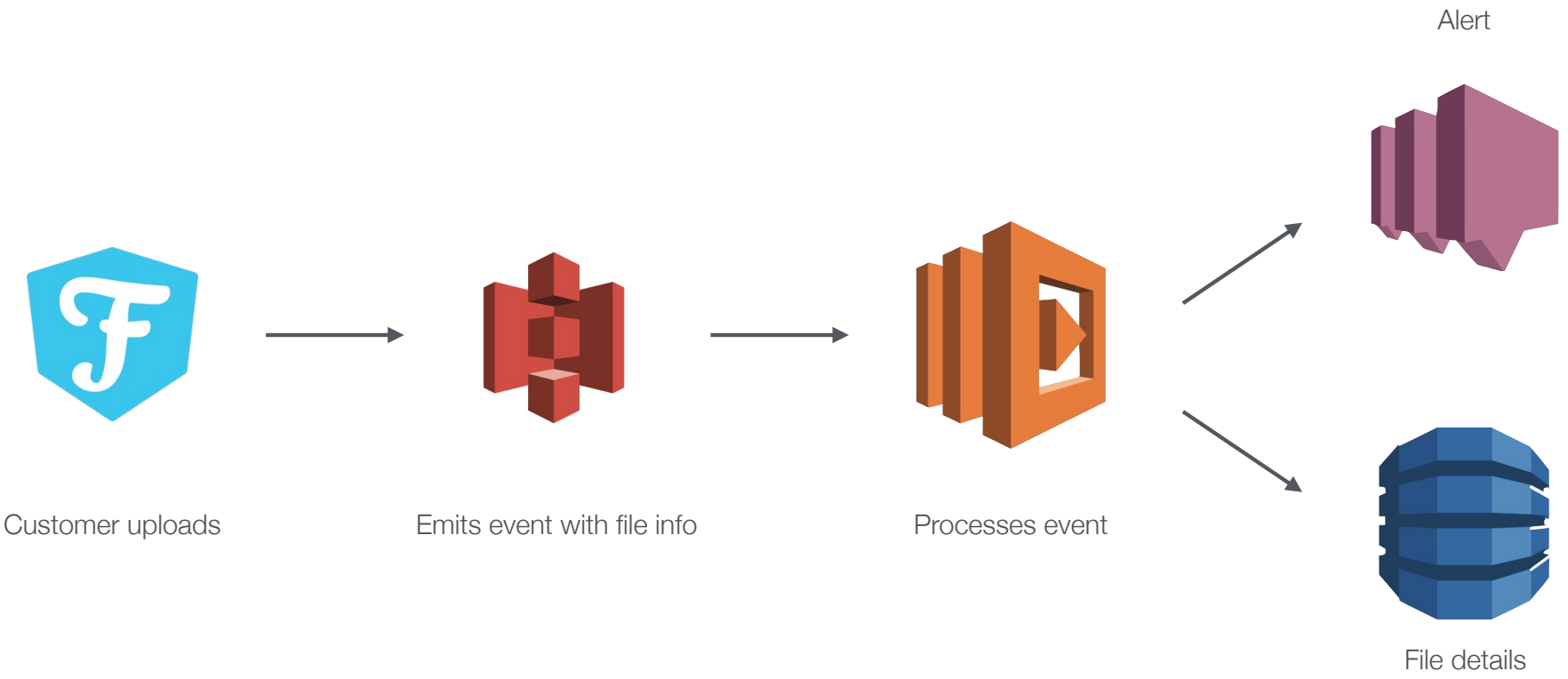
Logging and Monitoring Third–Party Services

Lambda

Logging and Monitoring Third–Party Services

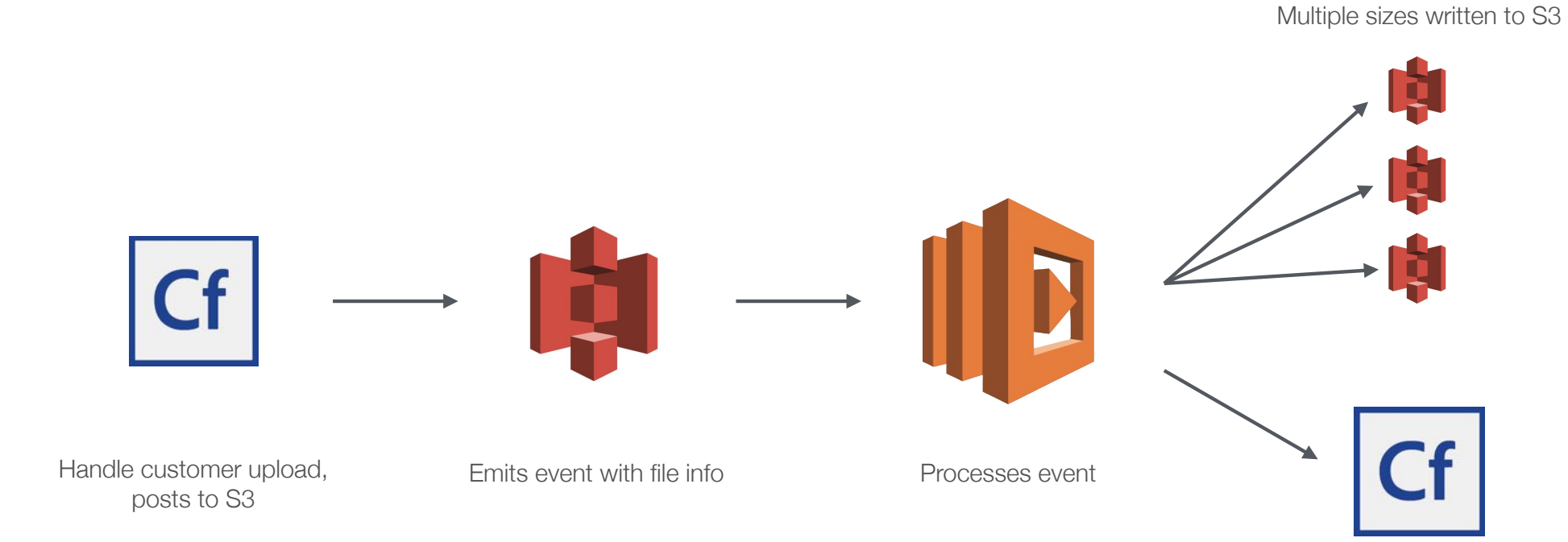
Lambda

Logging and Monitoring Third–Party Services



Lambda

Async Image Resizing



Video Production Workflow

Upload to ingest bucket

Processes event

Launches Elastic Transcoder job

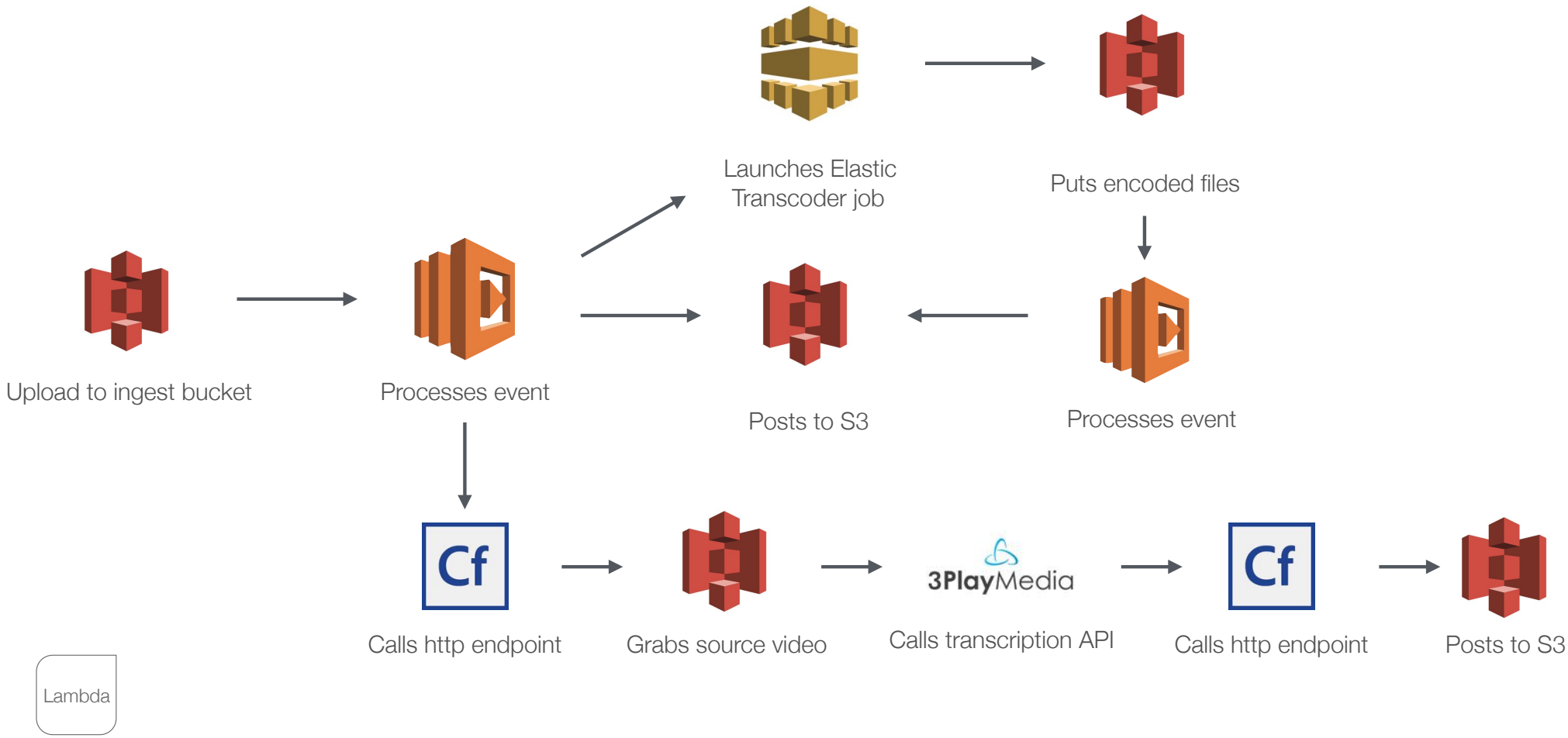
Puts encoded files

Processes event

Posts to S3

Lambda

Video Production Workflow (v2)



Demo

Invoking Lambda functions from 

github.com/brianklaas

= Focus on building apps,
not infrastructure

Lambda

Step Functions

(with some Rekognition)

Visual serverless orchestration

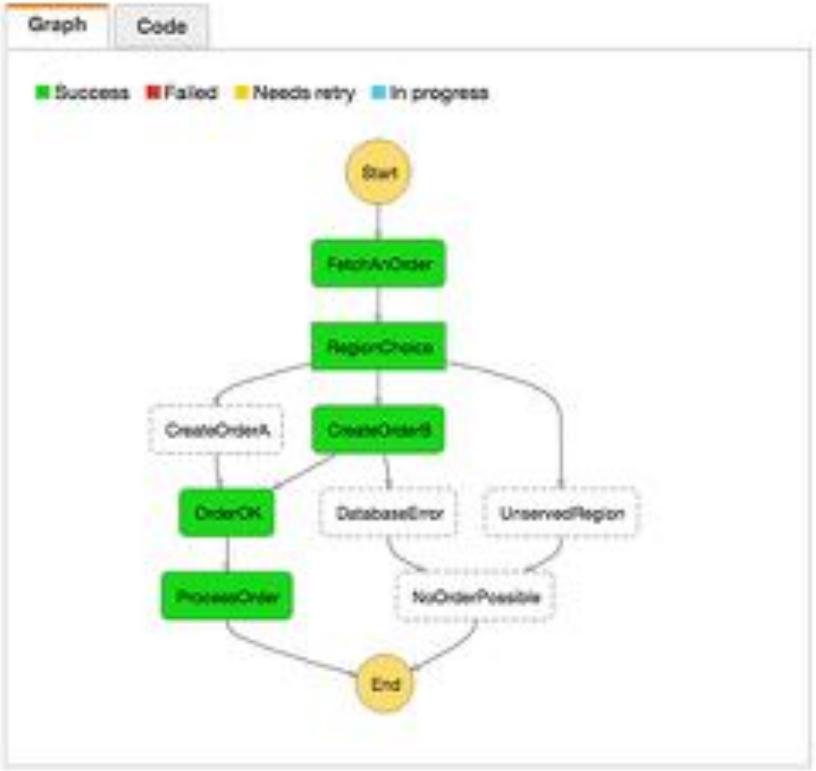
Automated, multi-step, asynchronous, serverless workflows in AWS

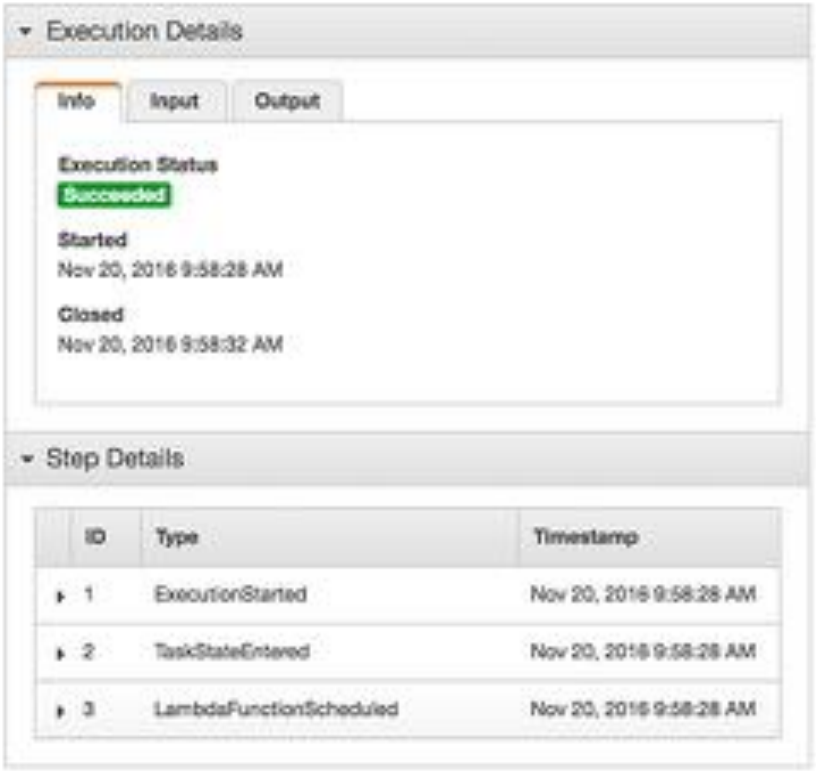
Avoid rebuilding the monolith

(Or the microlith)

Step Functions = “state machine”

Amazon States Language (JSON)
https://states-language.net/spec.html





Step Functions

Task
Sequence
Parallel
Branching
Waiting
Error Handling
Retries

Demo

Invoking Step Functions from 

github.com/brianklaas

Step Functions

More on Rekognition in the awsPlaybox

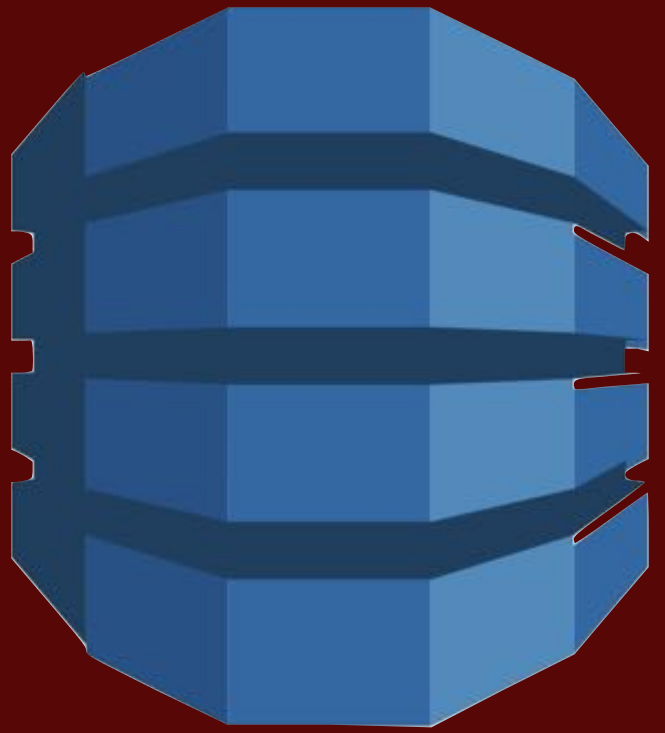
Step Functions

Automated, multi-step, asynchronous, serverless workflows in AWS

Use statelint to validate your ASL JSON

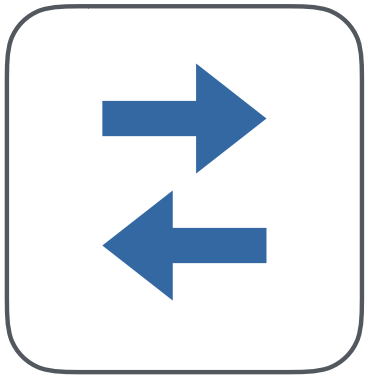
https://github.com/awslabs/statelint

DynamoDB



= Hugely scalable, high-write throughput document data store

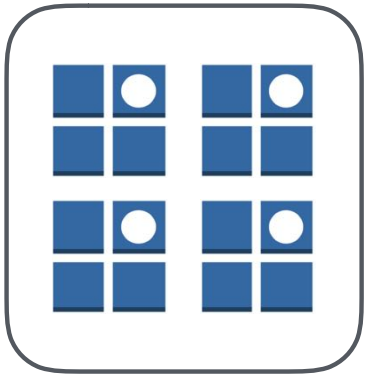
Dynamo DB



1. Set read/write capacity

2. Set primary and sort keys

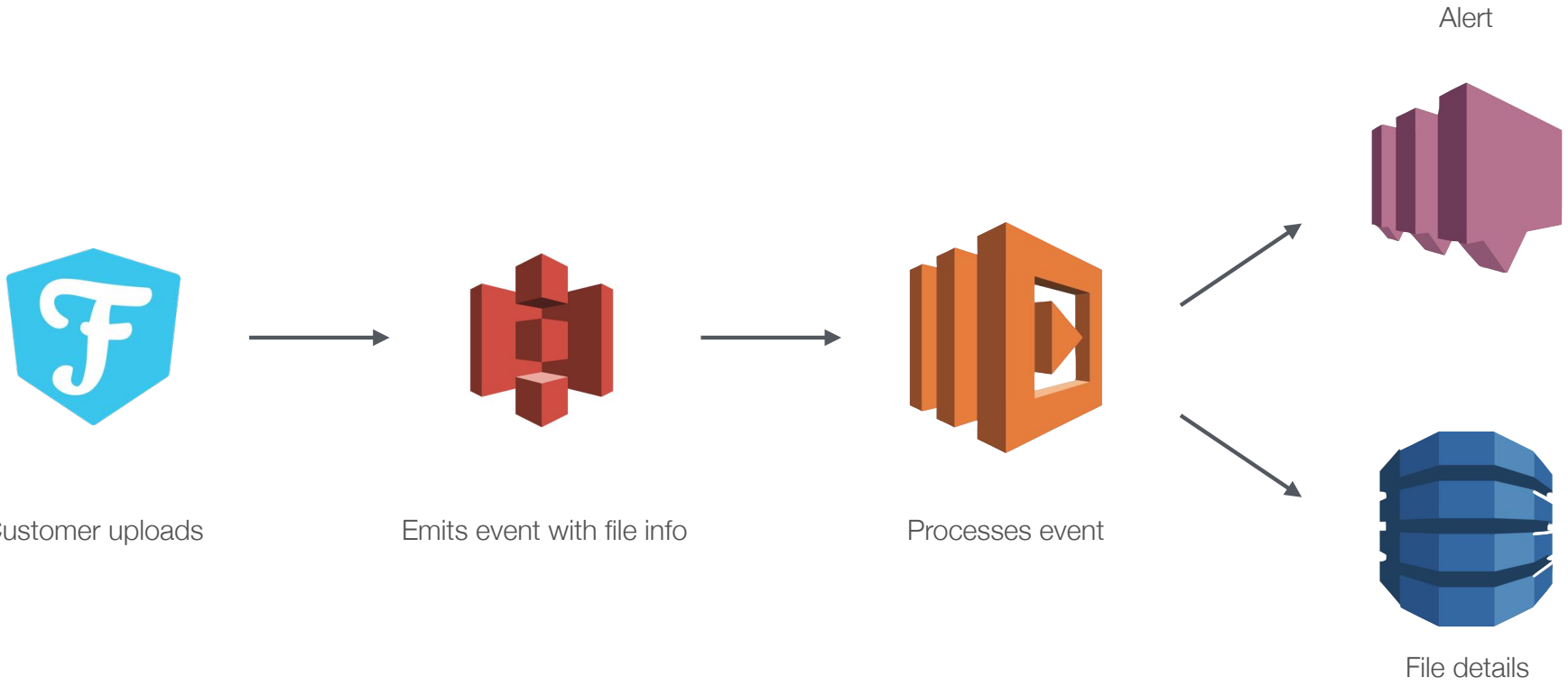
3. Set secondary indexes



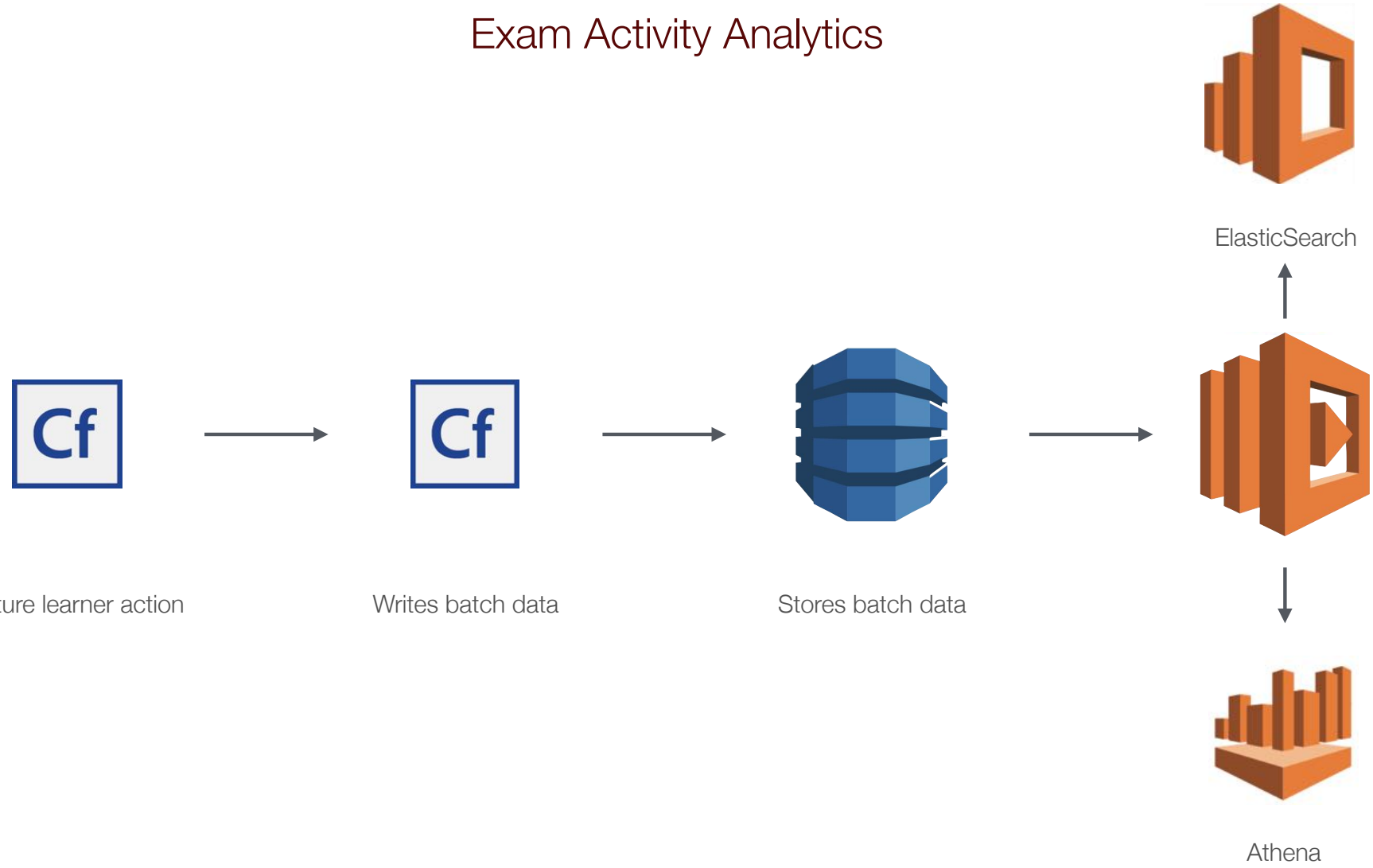
4. Write

Dynamo DB

Logging and Monitoring Third–Party Services



Exam Activity Analytics



Demo

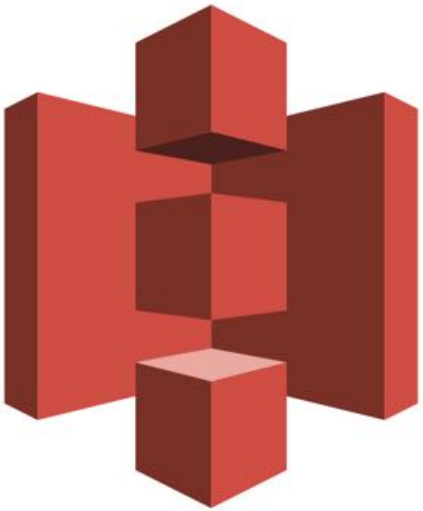
List, Put, Scan, Filter from 

github.com/brianklaas

Dynamo DB

Preparing for Outages

The Great S3 Outage of 2017



Outages

Plan for outages.

(Blame Amazon)

Batch upload from your servers

Use multiple regions

Shut off services

Have a plan.


Go Do!

AWS Playbox

github.com/brianklaas/awsplaybox