Stream processing with R in AWS

AWR, AWR.KMS, AWR.Kinesis (R packages) used in ECS

Gergely Daroczi

@daroczig

March 7, 2017

About me

About me

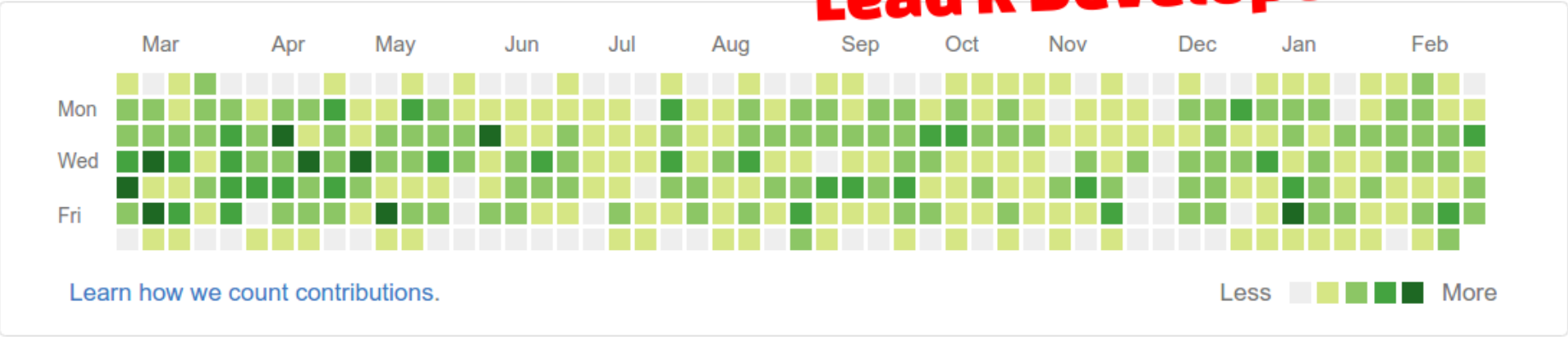
CARD.COM

About me

CARD.COM

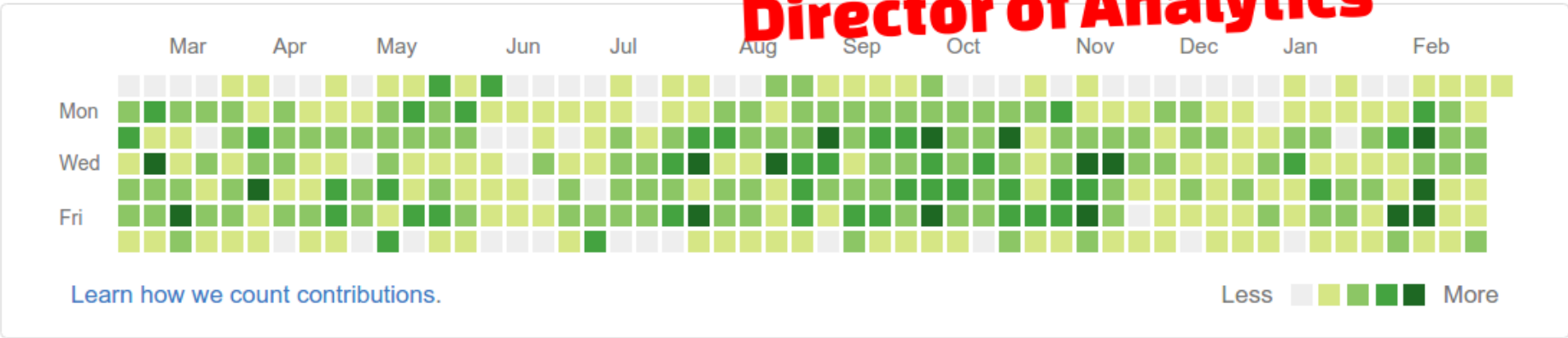
3,633 contributions in the year before last

Lead R Developer



3,470 contributions in the last year

Director of Analytics



CARD.com's View of the World

CARD.COM

Gergely Daróczi @daroczig · Apr 11

Just received my "I ♥ R" prepaid debit card from @CARD. Will be fun to use this #rstats designed card at #user2015 :)

RETWEETS 10 FAVORITES 16

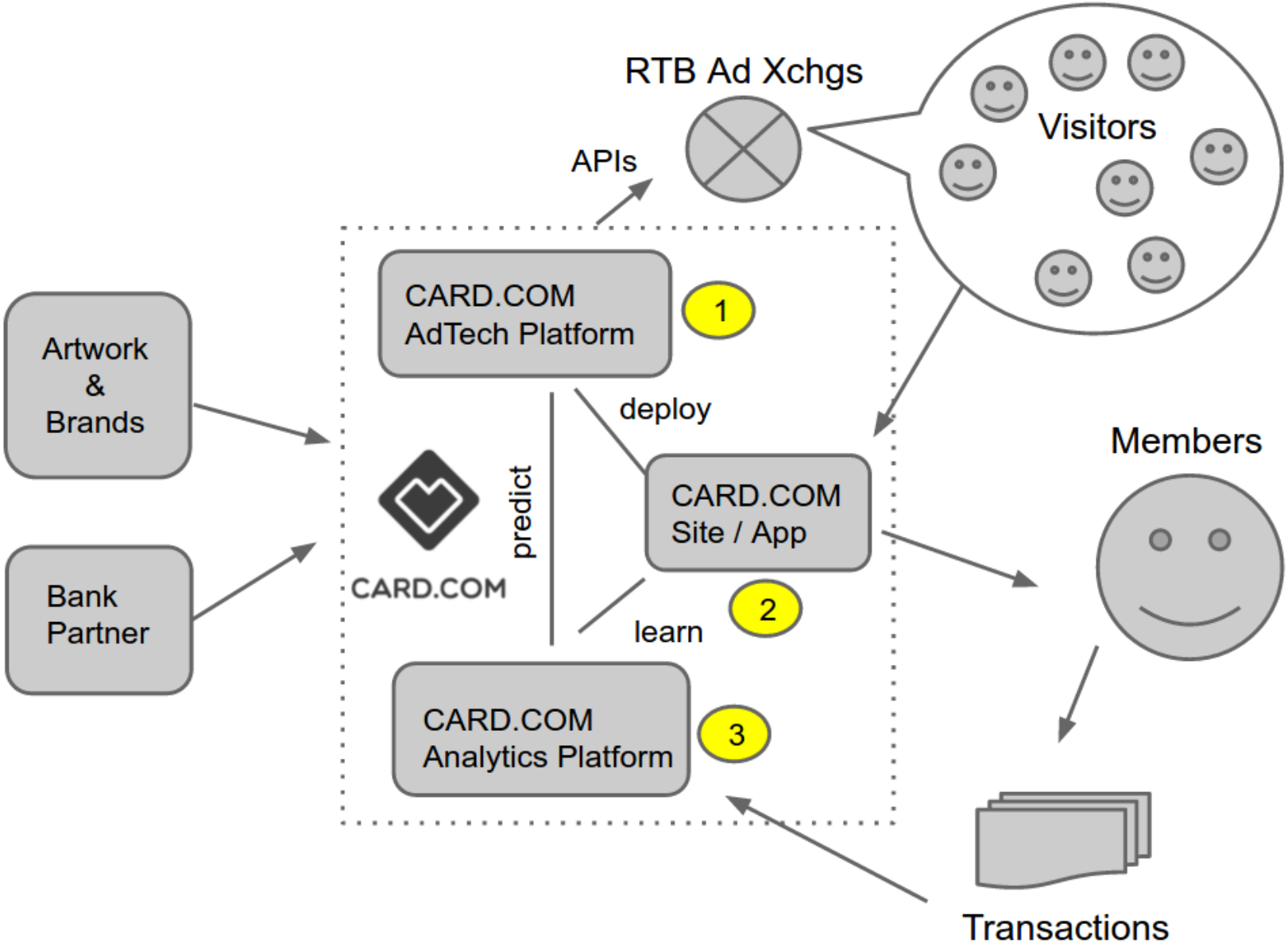
CARD.com's View of the World

CARD.COM

≠ ≠

≠

Modern Marketing at CARD.com



Further Data Partners

CARD.COM

- card transaction processors
- card manufacturers
- CIP/KYC service providers
- online ad platforms
- remarketing networks
- licensing partners
- communication engines
- others

My View on CARD.com

CARD.COM

Why not Hadoop instead of MySQL?

CARD.COM

Infrastructure

The goal of Rapporter was to provide a front-end to *R* in **all modern browsers** running on **various platforms** [A] – let it be a desktop, a notebook, tablet or a mobile phone. Users can access their data, reports and statistical tools stored in the Rapporter cloud from any place over the Internet and even do it collaboratively with other fellows and contributors.

A minor but useful part of the infrastructure is hosted at *Zendesk* that provides an extensible **knowledge base** and **support forum** [B].

All the requests and data packets sent by the clients to Rapporter servers hit our *Content Delivery Network* provider [C] first that would return all static content of the webapp **cache**d at several locations around the world for improved response times. The CDN also operates as a front-line **firewall** and filters out some unwanted and potentially dangerous packets and queries [D] beside minimizing the risk of (Distributed) Denial-of service attacks. Users can optionally use Rapporter over a **secure channel** by HTTPS protocol [E], as the data transmitted to and from CloudFlare is encrypted on demand for improved security.

The dynamic content is mainly served by our *Ruby on Rails* [F] workers in the means of a cluster of *thin* servers [G] running inside of our private internal network. This **content management system** is made of several separate threads replying to user request via a **load balancing** reverse proxy [H] that also serves static content, plus JavaScript, CSS and image assets.

Although we try to do our best with deploying working code on production servers, we also collect possible Rails **error messages** with *Errbit* [I].

Another major part of our setup are the *HAProxy* cluster [I] of *rApache* based *R* workers [J] running within an enforced *AppArmor* [K] profile and optional *AppArmor* hats based on user privileges. This latter Linux kernel **security module** ensures that the users could not directly touch the disks or make connections to our databases – even if some malicious code would somehow escape our in-house developed **sandbox** called *sandboxR*. The dynamic hat option allows fine-grained control over the **hardware resources** on a per-user basis in the means of e.g. CPU power and memory limit, or network access. Please see some further **security considerations** below.

As we are using *R* for creating **complex** or one-time and temporary **reports** via a *Graphical User Interface* or the recently introduced *Application Programming Interface* called *Rapplications*, our home-made internal *R* functions do not deal with any statistical problems, but rather provides an environment for the users to easily implement those. Rapporter is basically made of our open-source ***rapport* and *pander* packages** (please see below) beside the above described Rails front-end and hardened security tools, and the data, methods and results all bundled in various JSON driven databases [M].

All our servers are running *Ubuntu LTS* [N] on 64 bit with a decent amount of memory and CPU cores optionally dedicated to VIP customers, and continuously **monitored** 24 hours a day, 7 days a week via the public [9] *Pingdom* availability monitor [O] and a more detailed and technical, enterprise-class monitoring solution with thousands of metrics, called *Zabbix* [P] – beside Google Analytics of course.

Data storage [M]

Although administering and maintaining several similar database engines might not make much sense in most setup, we use two NoSQL databases for **improved performance**. *CouchDB* is awesome for its disk-based B-tree views, simple attachment concept and eventual consistency schema, while *MongoDB* makes the Rails models a lot more convenient to work with.
Gluster is a network filesystem that stores R generated images on a **replicated** and optionally distributed storage attached to the highly available Rails servers.

Infrastructure

CARD.COM

Why R?

CARD.COM

User Defined Java Class

Step name: Shorten Date

Classes and code fragments:

- Classes
- Code Snippits
- Input fields
  - Getting fields...please wait
- Info fields
  - Getting fields...please wait
- Output fields
  - Getting fields...please wait

Class code

Processor

```
import java.text.SimpleDateFormat;
import java.util.Date;
import java.text.ParseException;
import java.util.TimeZone;

private SimpleDateFormat df1 = new SimpleDateFormat("yyyy/MM/dd HH:mm:ss.SSS");

private SimpleDateFormat df2 = new SimpleDateFormat("yyyy-MM-dd HH");

public boolean processRow(StepMetaInterface smi, StepDataInterface sdi) throws KettleException, ParseException
{

    Object[] r = getRow();
    if (r == null) {
        setOutputDone();
        return false;
    }

    if (first)
    {
        first = false;
    }

    // It is always safest to call createOutputRow() to ensure that your output row's Object[] is large
    // enough to handle any new fields you are creating in this step.
    r = createOutputRow(r, data.outputRowMeta.size());

    df2.setTimeZone(TimeZone.getTimeZone("America/Los_Angeles"));
```

Line #: 0

Fields | Parameters | Info steps | Target steps

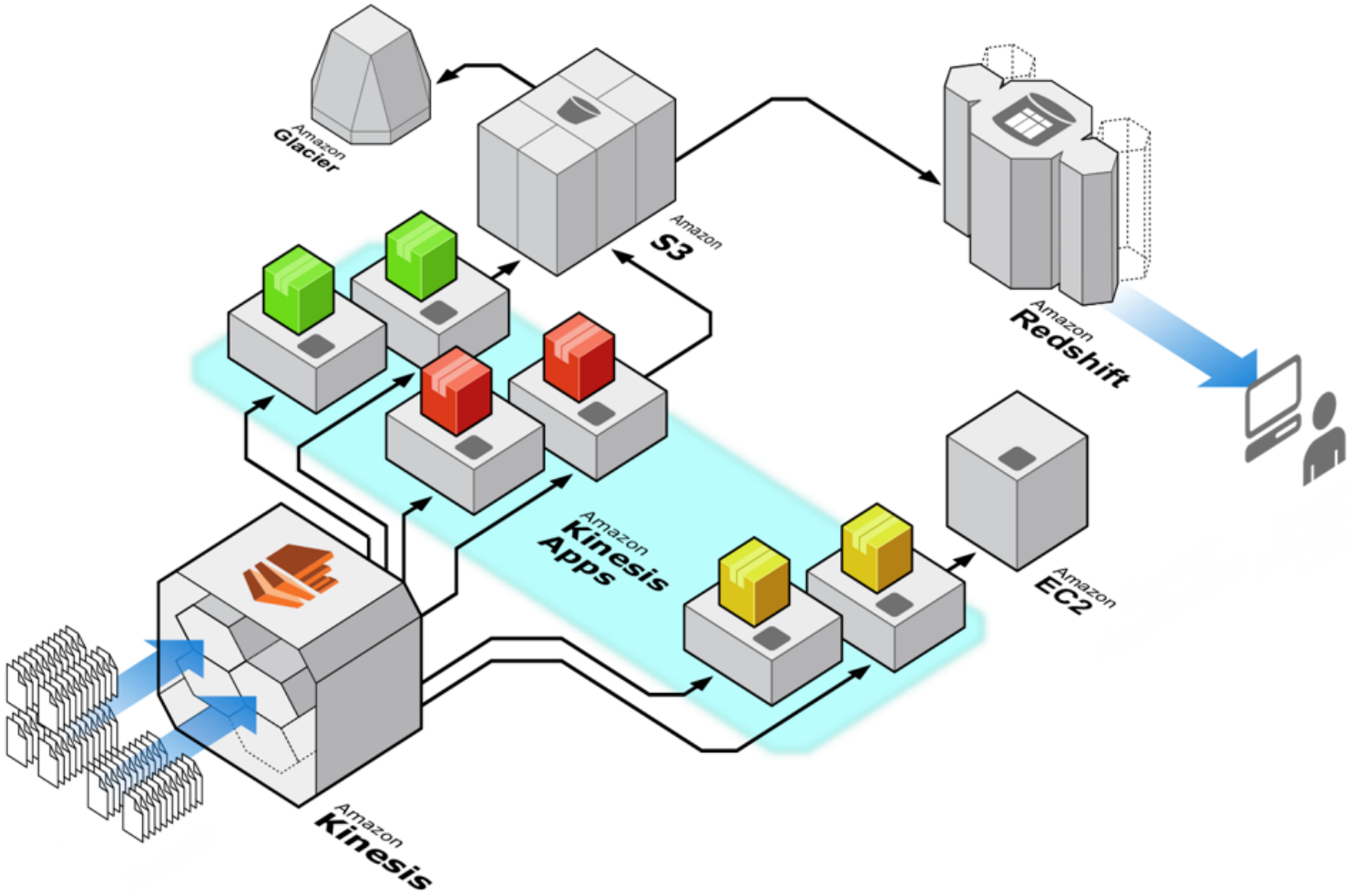
Fields

Clear the result fields?

#	Fieldname	Type	Length	Precision
1	RPT_DATE_SHORT	String		

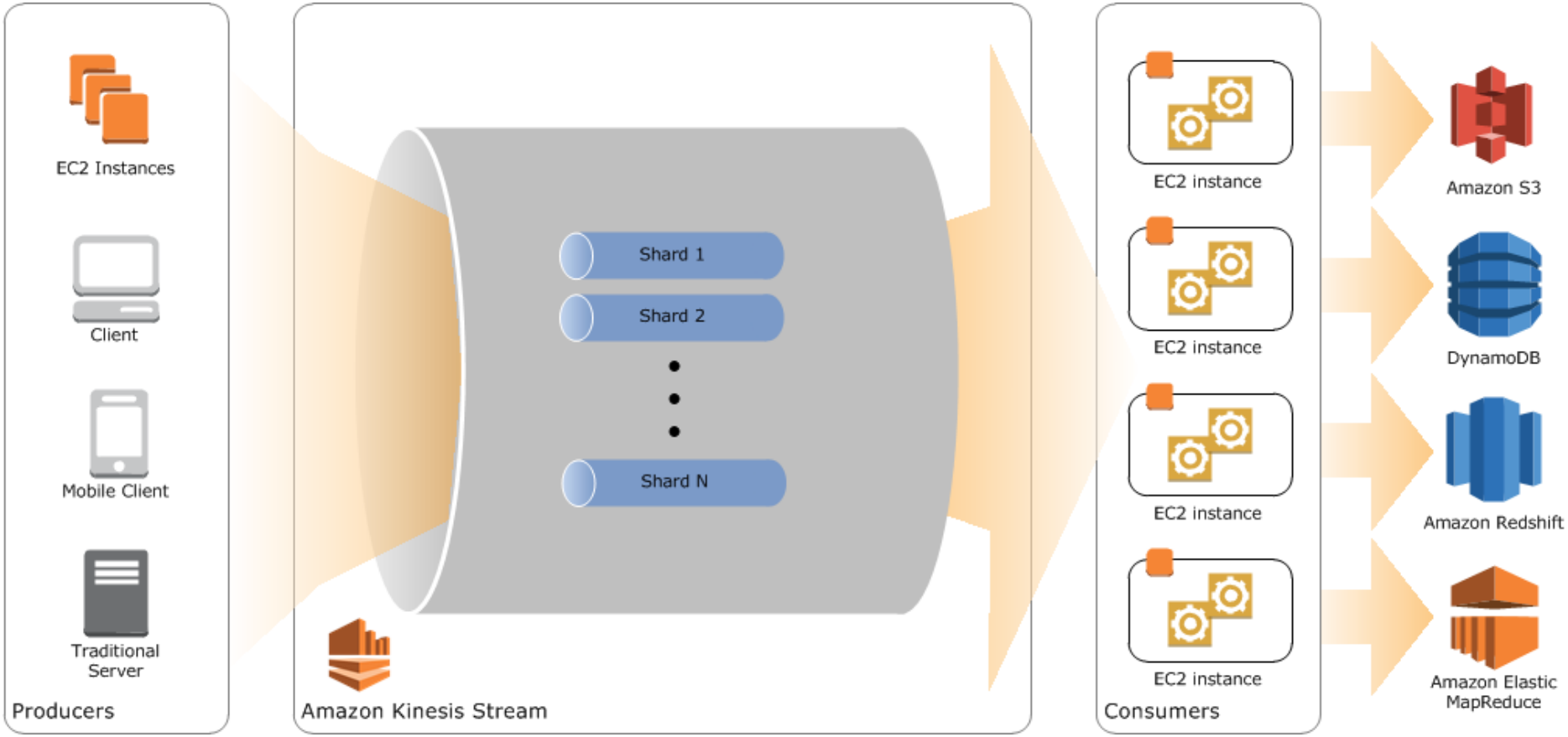
Help | OK | Cancel | Test class

Why Amazon Kinesis?



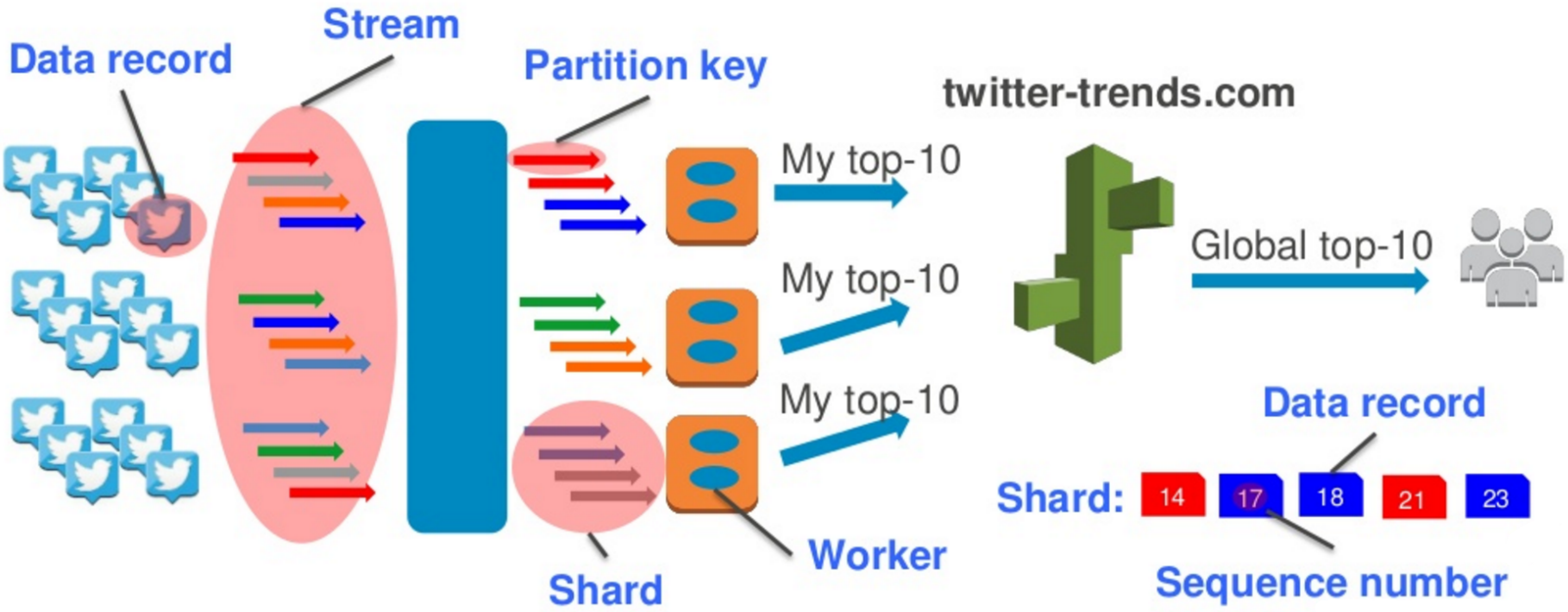
Source: Kinesis Product Details

Intro to Amazon Kinesis Streams



Source: Kinesis Developer Guide

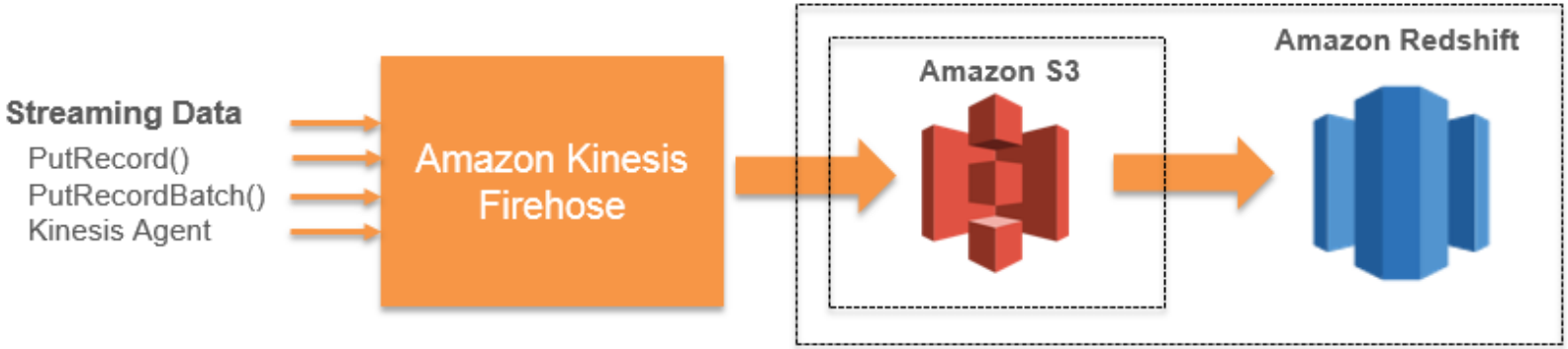
Intro to Amazon Kinesis Shards



Source: AWS re:Invent 2013

Deep Learning

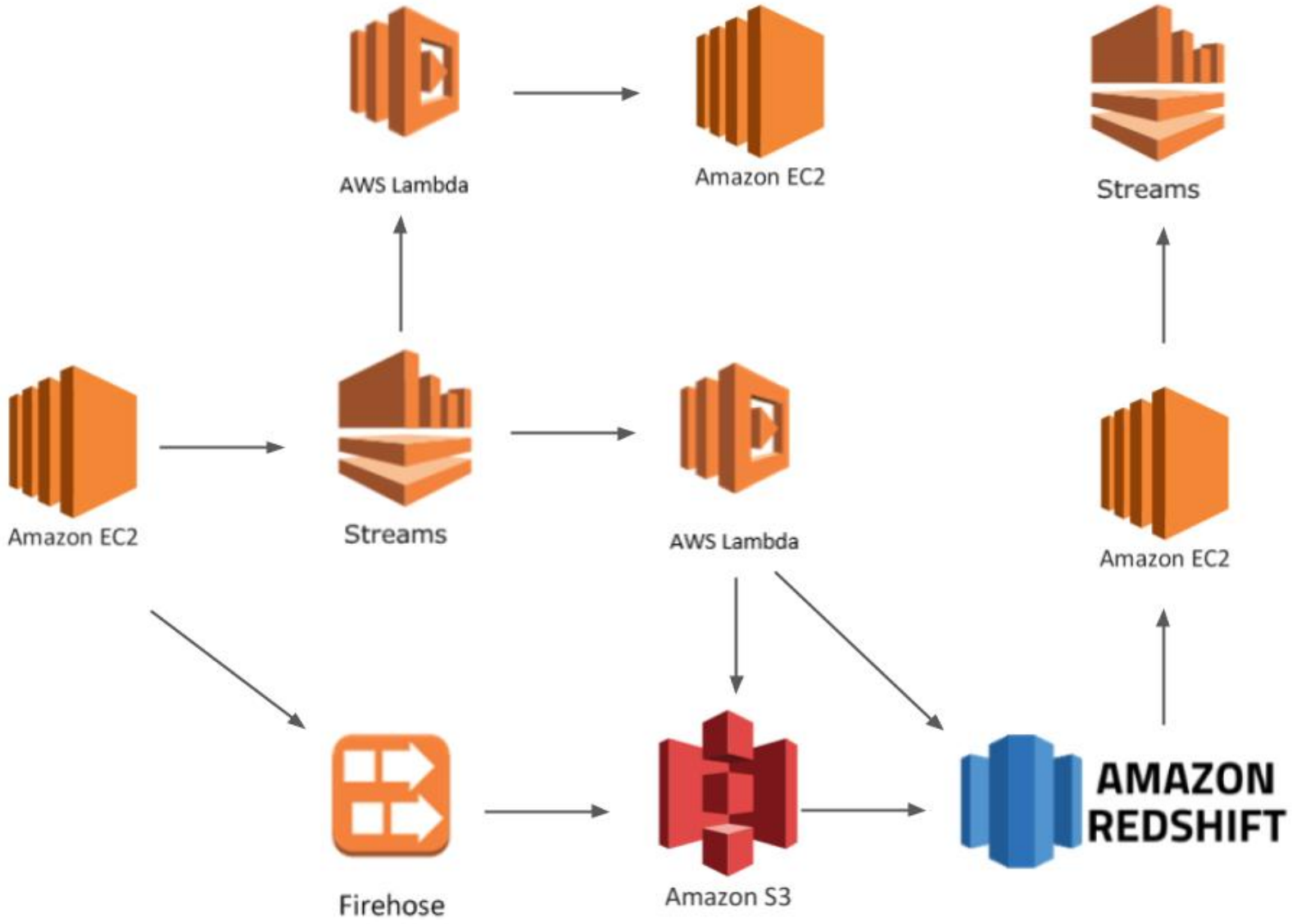
Deep Learning



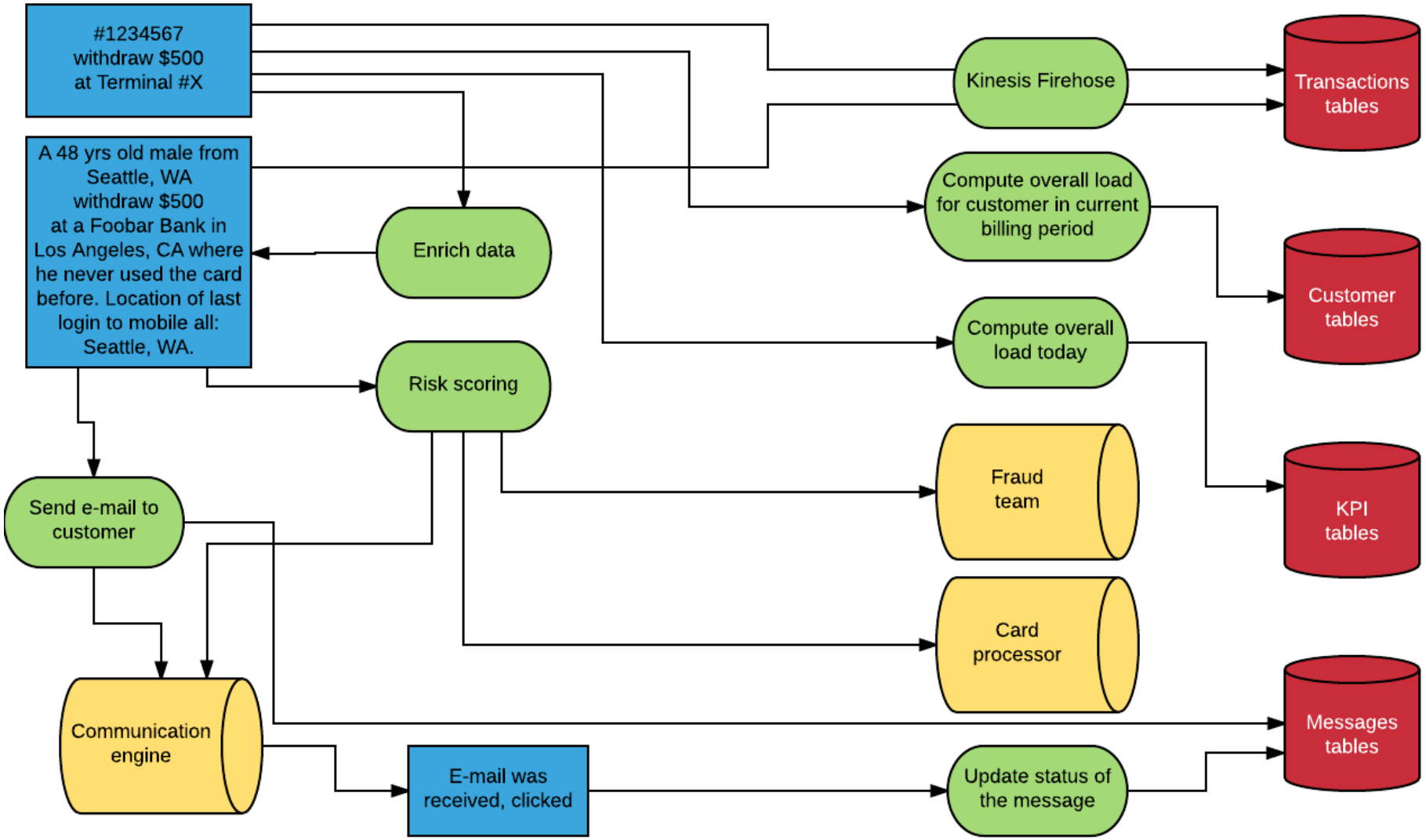
The S3 Object System

```
> x <- 3.14
> attr(x, 'class') <- 'standard'

> print.standard <- function(x, ...) {
+     ## SLA
+     if (runif(1) * 100 > 99.9) {
+         Sys.sleep(20)
+     }
+     futile.logger::flog.info(x)
+ }

> while (TRUE) print(x)
INFO [2017-03-03 22:27:57] 3.14
INFO [2017-03-03 22:27:57] 3.14
INFO [2017-03-03 22:27:57] 3.14
INFO [2017-03-03 22:28:17] 3.14
INFO [2017-03-03 22:28:17] 3.14
```

S4: Multiple Dispatch



Example use-case



How to Communicate with Kinesis

CARD.COM

Writing data to the stream:

- Amazon Kinesis Streams API, SDK
- Amazon Kinesis Producer Library (KPL) from Java
- flume-kinesis
- Amazon Kinesis Agent

Reading data from the stream:

- Amazon Kinesis Streams API, SDK
- Amazon Kinesis Client Library (KCL) from Java, Node.js, .NET, Python, Ruby

Managing streams:

- Amazon Kinesis Streams API (!)

Now We Need an R Client!

CARD.COM

```
> library(rJava)
> .jinit(classpath = list.files('~/Projects/AWR/inst/java/', full.names = TRUE))

> kc <- .jnew('com.amazonaws.services.kinesis.AmazonKinesisClient')
> kc$setEndpoint('kinesis.us-west-2.amazonaws.com', 'kinesis', 'us-west-2')

> sir <- .jnew('com.amazonaws.services.kinesis.model.GetShardIteratorRequest')
> sir$setStreamName('test_kinesis')
> sir$setShardId(.jnew('java/lang/String', '0'))
> sir$setShardIteratorType('TRIM_HORIZON')
> iterator <- kc$getShardIterator(sir)$getShardIterator()

> grr <- .jnew('com.amazonaws.services.kinesis.model.GetRecordsRequest')
> grr$setShardIterator(iterator)
> kc$getRecords(grr)$getRecords()
[1] "Java-Object{[{SequenceNumber: 49562894160449444332153346371084313572324361665
ApproximateArrivalTimestamp: Tue Jun 14 09:40:19 CEST 2016,
Data: java.nio.HeapByteBuffer[pos=0 lim=6 cap=6],PartitionKey: 42}]}"

> sapply(kc$getRecords(grr)$getRecords(),
+        function(x)
+        rawToChar(x$getData()$array()))
[1] "foobar"
```

Managing Shards via the Java SDK

CARD.COM

Let's merge two shards:

```
> ms <- .jnew('com.amazonaws.services.kinesis.model.MergeShardsRequest')
> ms$setShardToMerge('shardId-000000000000')
> ms$setAdjacentShardToMerge('shardId-000000000001')
> ms$setStreamName('test_kinesis')
> kc$mergeShards(ms)
```

What do we have now?

```
> kc$describeStream(StreamName = 'test_kinesis')$getStreamDescription()$getShards()
[1] "Java-Object{[
{ShardId: shardId-000000000000,HashKeyRange: {StartingHashKey: 0,EndingHashKey: 170
SequenceNumberRange: {
StartingSequenceNumber: 49562894160427143586954815717376297430913467927668719618,
EndingSequenceNumber: 49562894160438293959554081028945856364232263390243848194}},
{ShardId: shardId-000000000001,HashKeyRange: {StartingHashKey: 170141183460469231731
SequenceNumberRange: {
StartingSequenceNumber: 49562894160449444332153346340517833149186116289174700050,
EndingSequenceNumber: 49562894160460594704752611652087392082504911751749828626}},
{ShardId: shardId-000000000002,
ParentShardId: shardId-000000000000,
AdjacentParentShardId: shardId-000000000001,
HashKeyRange: {StartingHashKey: 0,EndingHashKey: 340282366920938463463374607431768
SequenceNumberRange: {StartingSequenceNumber: 49562904991497673099704924344727019522
```

Amazon Kinesis Client Library

CARD.COM

- An *easy-to-use* programming model for processing data

```
java -cp amazon-kinesis-client-1.7.3.jar \
  com.amazonaws.services.kinesis.multilang.MultiLangDaemon \
  app.properties
```

- *Scalable* and *fault-tolerant* processing (checkpointing via DynamoDB)
- Logging and metrics in CloudWatch
- The **MultiLangDaemon** spawns processes written in any language, communication happens via JSON messages sent over stdin/stdout
- Only a few events/methods to care about in the consumer application:
  1. initialize
  2. processRecords
  3. checkpoint
  4. shutdown

Messages from the KCL

CARD.COM

1. **initialize**:
   - Perform initialization steps
   - Write "status" message to indicate you are done
   - Begin reading line from STDIN to receive next action
2. **processRecords**:
   - Perform processing tasks (you may write a checkpoint message at any time)
   - Write "status" message to STDOUT to indicate you are done.
   - Begin reading line from STDIN to receive next action
3. **shutdown**:
   - Perform shutdown tasks (you may write a checkpoint message at any time)
   - Write "status" message to STDOUT to indicate you are done.
   - Begin reading line from STDIN to receive next action
4. **checkpoint**:
   - Decide whether to checkpoint again based on whether there is an error or not.

Again: Why R?

CARD.COM

User Defined Java Class

Step name: Shorten Date

Classes and code fragments:

- Classes
- Code Snippits
- Input fields
  - Getting fields...please wait
- Info fields
  - Getting fields...please wait
- Output fields
  - Getting fields...please wait

Class code

Processor

```
import java.text.SimpleDateFormat;
import java.util.Date;
import java.text.ParseException;
import java.util.TimeZone;

private SimpleDateFormat df1 = new SimpleDateFormat("yyyy/MM/dd HH:mm:ss.SSS");

private SimpleDateFormat df2 = new SimpleDateFormat("yyyy-MM-dd HH");

public boolean processRow(StepMetaInterface smi, StepDataInterface sdi) throws KettleException, ParseException
{

    Object[] r = getRow();
    if (r == null) {
        setOutputDone();
        return false;
    }

    if (first)
    {
        first = false;
    }

    // It is always safest to call createOutputRow() to ensure that your output row's Object[] is large
    // enough to handle any new fields you are creating in this step.
    r = createOutputRow(r, data.outputRowMeta.size());

    df2.setTimeZone(TimeZone.getTimeZone("America/Los_Angeles"));
```

Line #: 0

Fields | Parameters | Info steps | Target steps

Fields

Clear the result fields?

#	Fieldname	Type	Length	Precision
1	RPT_DATE_SHORT	String		

Help | OK | Cancel | Test class

R Script Interacting with KCL

CARD.COM

```
#!/usr/bin/r -i

while (TRUE) {

    ## read and parse JSON messages
    line <- fromJSON(readLines(n = 1))

    ## nothing to do unless we receive records to process
    if (line$action == 'processRecords') {

        ## process each record
        lapply(line$records, function(r) {

            business_logic(fromJSON(rawToChar(base64_dec(r$data))))
            cat(toJSON(list(action = 'checkpoint', checkpoint = r$sequenceNumber)))

        })
    }

    ## return response in JSON
    cat(toJSON(list(action = 'status', responseFor = line$action)))

}
```

R Script Interacting with KCL

CARD.COM

```
#!/usr/bin/r -i

while (TRUE) {

    ## read and parse JSON messages
    line <- fromJSON(readLines(n = 1))

    ## nothing to do unless we receive records to process
    if (line$action == 'processRecords') {

        ## process each record
        lapply(line$records, function(r) {

            business_logic(fromJSON(rawToChar(base64_dec(r$data))))
            cat(toJSON(list(action = 'checkpoint', checkpoint = r$sequenceNumber)))

        })
    }

    ## return response in JSON
    cat(toJSON(list(action = 'status', responseFor = line$action)))

}
```

Get rid of the bugs and the boilerplate

```
> install.packages('AWR.Kinesis')
also installing the dependency ‘AWR’

trying URL 'https://cloud.r-project.org/src/contrib/AWR_1.11.89.tar.gz'
Content type 'application/x-gzip' length 3125 bytes

trying URL 'https://cloud.r-project.org/src/contrib/AWR.Kinesis_1.7.3.tar.gz'
Content type 'application/x-gzip' length 3091459 bytes (2.9 MB)

* installing *source* package ‘AWR’ ...
** testing if installed package can be loaded
trying URL 'https://gitlab.com/cardcorp/AWR/repository/archive.zip?ref=1.11.89'
downloaded 58.9 MB
* DONE (AWR)

* installing *source* package ‘AWR.Kinesis’ ...
* DONE (AWR.Kinesis)
```

Add content to the boilerplate

Business logic coded in R (demo_app.R):

```
library(AWR.Kinesis)
kinesis_consumer(processRecords = function(records) {
    flog.info(jsonlite::toJSON(records))
})
```

Add content to the boilerplate

Business logic coded in R (demo_app.R):

```
library(AWR.Kinesis)
kinesis_consumer(processRecords = function(records) {
    flog.info(jsonlite::toJSON(records))
})
```

Note

This is not something you should run in RStudio.

Add content to the boilerplate

CARD.COM

Business logic coded in R (demo_app.R):

```
library(AWR.Kinesis)
kinesis_consumer(processRecords = function(records) {
    flog.info(jsonlite::toJSON(records))
})
```

Config file for the MultiLangDaemon (demo_app.properties):

```
executableName = ./demo_app.R
streamName = demo_stream
applicationName = demo_app
```

Start the MultiLangDaemon:

```
/usr/bin/java -cp AWR/java/*:AWR.Kinesis/java/*:./ \
    com.amazonaws.services.kinesis.multilang.MultiLangDaemon \
    ./demo_app.properties
```

''Advanced'' AWR.Kinesis features

CARD.COM

```
library(futile.logger)
library(AWR.Kinesis)

kinesis_consumer(

    initialize     = function()
        flog.info('Hello'),

    processRecords = function(records)
        flog.info(paste('Received', nrow(records), 'records from Kinesis')),

    shutdown       = function()
        flog.info('Bye'),

    updater        = list(
        list(1, function()
            flog.info('Updating some data every minute')),
        list(1/60*10, function()
            flog.info(paste(
                'This is a high frequency updater call',
                'running every 10 seconds')))),

    checkpointing = 1,
    logfile = '/logs/logger.log')
```

Let's run it locally!

CARD.COM

Note

In theory you could, but this is not something you should run in RStudio.

1. Create a Kinesis Stream
2. Create an IAM user with DynamoDB and Kinesis permissions
3. Write data to the Stream
4. Run the MultiLangDaemon referencing the properties file

Let's run it locally!

CARD.COM

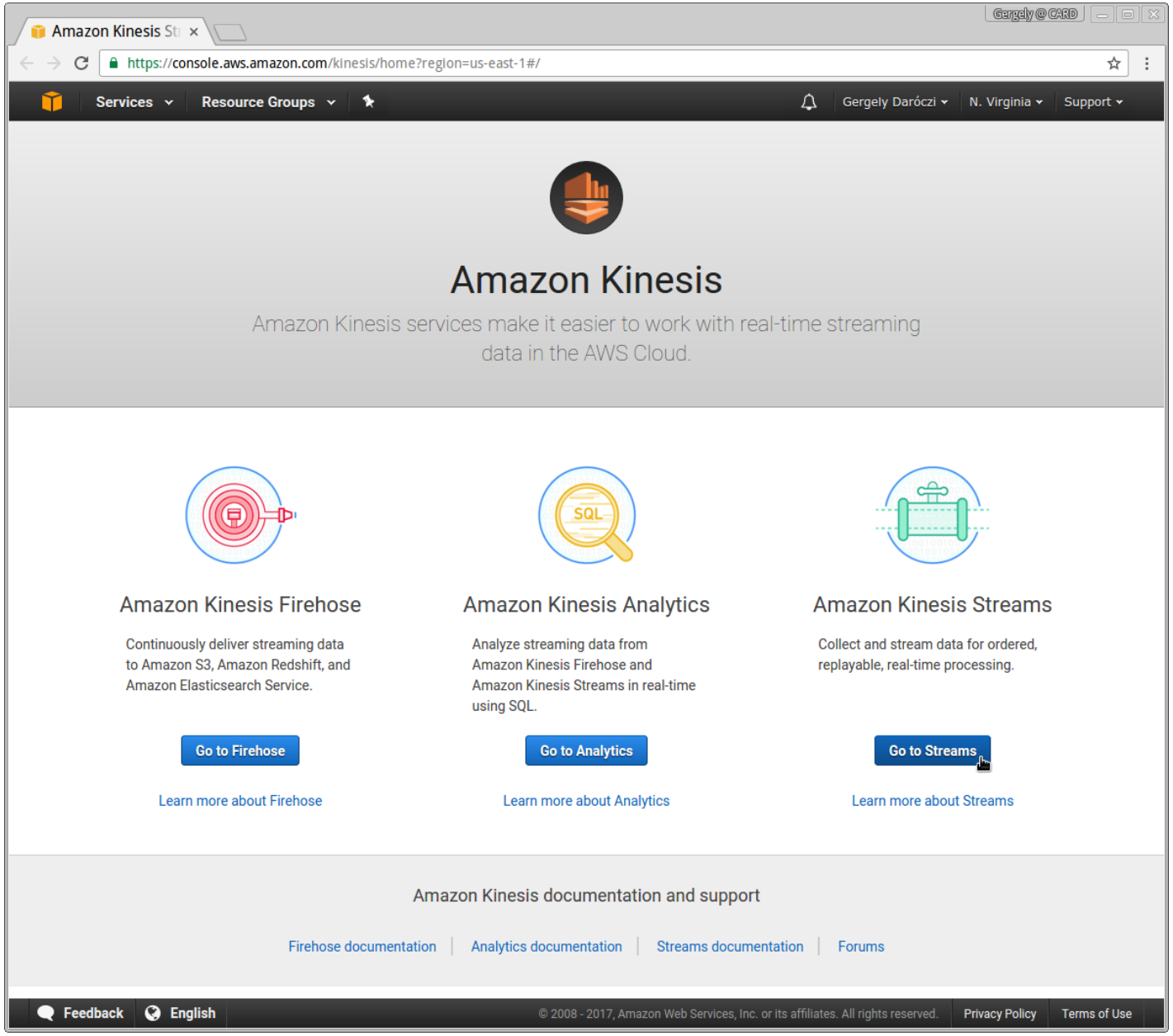
Note

In theory you could, but this is not something you should run in RStudio.

1. Create a Kinesis Stream
2. Create an IAM user with DynamoDB and Kinesis permissions
3. Write data to the Stream
4. Run the MultiLangDaemon referencing the properties file

Create a Kinesis Stream

CARD.COM



Create a Kinesis Stream

CARD.COM

Amazon Kinesis Str ×

https://console.aws.amazon.com/kinesis/home?region=us-east-1#/streams/create

Services | Resource Groups | Gergely Daróczi | N. Virginia | Support

Amazon Kinesis

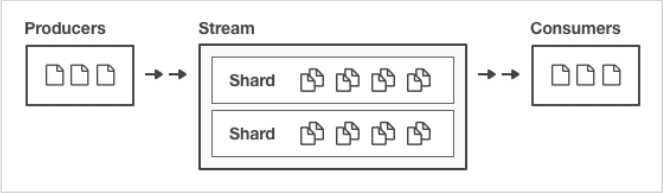
- Streams
- Firehose
- Analytics

Create stream

Stream name* test-AWR

Shards

A shard is a unit of throughput capacity. Each shard ingests up to 1MB/sec and 1000 records/sec, and emits up to 2MB/sec. To accommodate for higher or lower throughput, the number of shards can be modified after the stream is created using the API. Learn more



Estimate the number of shards you'll need

Number of shards* 1

You can provision up to 48 more shards before hitting your account limit of 50.
Learn more or request a shard limit increase for this account

Total stream capacity Values are calculated based on the number of shards entered above.

Write	1	MB per second
	1000	Records per second
Read	2	MB per second

* Required

Cancel | Create stream

Check the Kinesis Stream

CARD.COM

Amazon Kinesis Str

https://console.aws.amazon.com/kinesis/home?region=us-east-1#/streams/details?streamName=test-AWR

Services | Resource Groups | Gergely Daróczi | N. Virginia | Support

Amazon Kinesis
- Streams
- Firehose
- Analytics

Streams > test-AWR

test-AWR

Delete stream

Details | Monitoring | Tags

Time range: Last hour Statistic: Average Period: 1 minute

Get Records (bytes)

4.0M, 3.0M, 2.0M, 1.0M, 0.0

16:00, 16:15, 16:30, 16:45

Get Records Iterator Age (ms)

1.0, 0.5, 0.0

16:00, 16:15, 16:30, 16:45

Get Records Latency (ms)

14.0, 13.0, 12.0, 11.0, 10.0

13.5

03/05 16:40
10.3

03/05 16:40

16:00, 16:15, 16:30, 16:45

Get Records (records/second)

2.0, 1.5, 1.0, 0.5, 0.0

16:00, 16:15, 16:30, 16:45

Get Records (success count)

Incoming Data (bytes)

Create an IAM user

CARD.COM

IAM Management

https://console.aws.amazon.com/iam/home?region=us-east-1#/users/AWR

Services | Resource Groups | Gergely Daróczi | Global | Support

Search IAM

- Dashboard
- Groups
- Users
- Roles
- Policies
- Identity providers
- Account settings
- Credential report

- Encryption keys

Users > AWR

Users: AWR

User ARN	arn:aws:iam:: [redacted] :user/AWR
Path	/
Creation time	2017-02-13 16:04 PST

Permissions | Groups (0) | Security credentials | Access Advisor

Add permissions

Number of attached policies 5

- AmazonEC2ContainerRegistryFullAccess - AWS Managed policy
- CloudWatchFullAccess - AWS Managed policy
- cloudwatch-putmetrics - Managed policy
- AmazonDynamoDBFullAccess - AWS Managed policy
- AmazonKinesisFullAccess - AWS Managed policy

Add inline policy

Feedback | English | © 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved. | Privacy Policy | Terms of Use

Write Data to the Stream from R

```
library(rJava)
.jcall("java/lang/System", "S", "setProperty", "aws.profile", "personal")

library(AWR.Kinesis)
library(jsonlite)
library(futile.logger)
library(nycflights13)
while (TRUE) {

    ## pick a ~car~flight
    flight <- flights[sample(1:nrow(flights), 1), ]

    ## prr <- .jnew('com.amazonaws.services.kinesis.model.PutRecordRequest')
    ## prr$setStreamName('test1')
    ## prr$setData(J('java.nio.ByteBuffer')$wrap(.jbyte(charToRaw(toJSON(car)))))
    ## prr$setPartitionKey(rownames(car))
    ## kc$putRecord(prr)

    res <- kinesis_put_record(stream = 'test-AWR', region = 'us-east-1',
                              data = toJSON(flight), partitionKey = flight$dest)
    flog.info(paste('Pushed a new flight to Kinesis:', res$sequenceNumber))

}
```

Write Data to the Stream from R

CARD.COM

Minibuf-1

File Edit Options Buffers Tools Minibuf Help

```
> library(rJava)
> .jcall("java/lang/System", "S", "setProperty", "aws.profile", "personal")
> library(AWR.Kinesis); library(jsonlite); library(futile.logger); library(nycflights13)
> while (TRUE) {
+     flight <- flights[sample(1:nrow(flights), 1), ]
+     res <- kinesis_put_record('test-AWR', region = 'us-east-1', data = toJSON(flight),
+                               partitionKey = flight$dest)
+     flog.info(paste('Pushed a new flight to Kinesis:', res$sequenceNumber))
+ }
INFO [2017-03-06 21:47:16] Pushed a new flight to Kinesis: 49571082550613725758991014186133813750336290428328869938
INFO [2017-03-06 21:47:17] Pushed a new flight to Kinesis: 49571082550613725758991014186380434617537674848688406578
INFO [2017-03-06 21:47:17] Pushed a new flight to Kinesis: 49571082550613725758991014186610130523264454391882580018
INFO [2017-03-06 21:47:17] Pushed a new flight to Kinesis: 49571082550613725758991014186870049574481599664444407858
INFO [2017-03-06 21:47:18] Pushed a new flight to Kinesis: 49571082550613725758991014187125132922420286489026887730
INFO [2017-03-06 21:47:18] Pushed a new flight to Kinesis: 49571082550613725758991014187394723380194348794986364978
INFO [2017-03-06 21:47:19] Pushed a new flight to Kinesis: 49571082550613725758991014187636508544117274698647076914
INFO [2017-03-06 21:47:19] Pushed a new flight to Kinesis: 49571082550613725758991014187857741969106751837618307122
INFO [2017-03-06 21:47:20] Pushed a new flight to Kinesis: 49571082550613725758991014188075348616637385157784895538
INFO [2017-03-06 21:47:20] Pushed a new flight to Kinesis: 49571082550613725758991014188250642860481506388117291058
INFO [2017-03-06 21:47:21] Pushed a new flight to Kinesis: 49571082550613725758991014188505726208420193143980294194
INFO [2017-03-06 21:47:21] Pushed a new flight to Kinesis: 49571082550613725758991014188814002292421923652249845810
INFO [2017-03-06 21:47:21] Pushed a new flight to Kinesis: 49571082550613725758991014189015892904297566724425777202
INFO [2017-03-06 21:47:22] Pushed a new flight to Kinesis: 49571082550613725758991014189266140548957795032309432370
INFO [2017-03-06 21:47:22] Pushed a new flight to Kinesis: 49571082550613725758991014189516388193618023271473610802
INFO [2017-03-06 21:47:23] Pushed a new flight to Kinesis: 49571082550613725758991014189794441132129388119094984754
INFO [2017-03-06 21:47:23] Pushed a new flight to Kinesis: 49571082550613725758991014190015674557118865258066214962
INFO [2017-03-06 21:47:24] Pushed a new flight to Kinesis: 49571082550613725758991014190342084528414815135236882482
INFO [2017-03-06 21:47:24] Pushed a new flight to Kinesis: 49571082550613725758991014190545184066110072905306996786
INFO [2017-03-06 21:47:25] Pushed a new flight to Kinesis: 49571082550613725758991014190772462120197623190151757874
INFO [2017-03-06 21:47:25] Pushed a new flight to Kinesis: 49571082550613725758991014190965890251335963858104746034
INFO [2017-03-06 21:47:25] Pushed a new flight to Kinesis: 49571082550613725758991014191170198714850836257349566514
INFO [2017-03-06 21:47:26] Pushed a new flight to Kinesis: 49571082550613725758991014191397476768938386542194327602
INFO [2017-03-06 21:47:26] Pushed a new flight to Kinesis: 49571082550613725758991014191629590526304395412457390130
INFO [2017-03-06 21:47:27] Pushed a new flight to Kinesis: 49571082550613725758991014191910061316454989380989222962
INFO [2017-03-06 21:47:27] Pushed a new flight to Kinesis: 49571082550613725758991014192179651774229051686948700210
INFO [2017-03-06 21:47:28] Pushed a new flight to Kinesis: 49571082550613725758991014192373079905367392423621165106
INFO [2017-03-06 21:47:28] Pushed a new flight to Kinesis: 49571082550613725758991014192595522256176484191767101490
INFO [2017-03-06 21:47:28] Pushed a new flight to Kinesis: 49571082550613725758991014192772025425840220119993679922
INFO [2017-03-06 21:47:29] Pushed a new flight to Kinesis: 49571082550613725758991014192981169592633550967217848370
INFO [2017-03-06 21:47:29] Pushed a new flight to Kinesis: 49571082550613725758991014193206029795081871993713197106
INFO [2017-03-06 21:47:30] Pushed a new flight to Kinesis: 49571082550613725758991014193413965036055588280482136114
INFO [2017-03-06 21:47:30] Pushed a new flight to Kinesis: 49571082550613725758991014193609811018833158206784536626
INFO [2017-03-06 21:47:30] Pushed a new flight to Kinesis: 49571082550613725758991014193818955185626489122728181810
INFO [2017-03-06 21:47:31] Pushed a new flight to Kinesis: 49571082550613725758991014194043815388074810149223530546
INFO [2017-03-06 21:47:31] Pushed a new flight to Kinesis: 49571082550613725758991014194267466664703516546544173106
INFO [2017-03-06 21:47:32] Pushed a new flight to Kinesis: 49571082550613725758991014194544310677395266696271364146
INFO [2017-03-06 21:47:32] Pushed a new flight to Kinesis: 49571082550613725758991014194752245918368982914320826418
INFO [2017-03-06 21:47:33] Pushed a new flight to Kinesis: 49571082550613725758991014195169325326136030048313933874
INFO [2017-03-06 21:47:34] Pushed a new flight to Kinesis: 49571082550613725758991014195478810335957375117038714930
INFO [2017-03-06 21:47:34] Pushed a new flight to Kinesis: 49571082550613725758991014195897098669544036880206528562
INFO [2017-03-06 21:47:35] Pushed a new flight to Kinesis: 49571082550613725758991014196216255085922298982328959026
INFO [2017-03-06 21:47:35] Pushed a new flight to Kinesis: 49571082550613725758991014196406056439601795831477305394
INFO [2017-03-06 21:47:36] Pushed a new flight to Kinesis: 49571082550613725758991014196589813164183219466032644146
INFO [2017-03-06 21:47:36] Pushed a new flight to Kinesis: 49571082550613725758991014196811046589172696673723351090
INFO [2017-03-06 21:47:36] Pushed a new flight to Kinesis: 49571082550613725758991014197006892571950266600025751602
INFO [2017-03-06 21:47:37] Pushed a new flight to Kinesis: 49571082550613725758991014197254722364971265580840517682
INFO [2017-03-06 21:47:37] Pushed a new flight to Kinesis: 49571082550613725758991014197475955789960742857250701362
INFO [2017-03-06 21:47:38] Pushed a new flight to Kinesis: 49571082550613725758991014197724994508801356467240173618
INFO [2017-03-06 21:47:38] Pushed a new flight to Kinesis: 49571082550613725758991014197965570746904667741726179378
INFO [2017-03-06 21:47:39] Pushed a new flight to Kinesis: 49571082550613725758991014198145700694027247488757399602
INFO [2017-03-06 21:47:39] Pushed a new flight to Kinesis: 49571082550613725758991014198412873300162080536367464498
INFO [2017-03-06 21:47:40] Pushed a new flight to Kinesis: 49571082550613725758991014198640151354249630889931702322
INFO [2017-03-06 21:47:41] Pushed a new flight to Kinesis: 49571082550613725758991014199353417587822262171727822898
INFO [2017-03-06 21:47:41] Pushed a new flight to Kinesis: 49571082550613725758991014199624216971415939106862006322
 * 12M U:  R-kinesis-putrecord   iESS [R]: run                                    109137 :  0   99%
M-x butterfly
```

Reading Data from the Stream

CARD.COM

```
## get an iterator
sir <- .jnew('com.amazonaws.services.kinesis.model.GetShardIteratorRequest')
sir$setStreamName('test-AWR')
sir$setShardId(.jnew('java/lang/String', '0'))
sir$setShardIteratorType('TRIM_HORIZON')
kc <- .jnew('com.amazonaws.services.kinesis.AmazonKinesisClient')
kc$setEndpoint('kinesis.us-east-1.amazonaws.com')
iterator <- kc$getShardIterator(sir)$getShardIterator()

## get records
grr <- .jnew('com.amazonaws.services.kinesis.model.GetRecordsRequest')
grr$setShardIterator(iterator)
records <- kc$getRecords(grr)$getRecords()

## transform to string
json <- sapply(records, function(x)
    rawToChar(x$getData()$array()))

## decode JSON
json[1]
fromJSON(json[1])
rbindlist(lapply(json, fromJSON))
```

Running the MultiLangDaemon locally

CARD.COM

Terminix: Default

1: daroczig@gergely-CARD: ~/Projects/card-rocker/r-kinesis-example/files

```
~/Projects/card-rocker/r-kinesis-example/files  master ?  export AWS_PROFILE=personal
~/Projects/card-rocker/r-kinesis-example/files  master ?  /usr/bin/java -cp \
"/usr/local/lib/R/site-library/AWR/java/*:/usr/local/lib/R/site-library/AWR.Kinesis/java/*:./" \
com.amazonaws.services.kinesis.multilang.MultiLangDaemon ./app.properties
Mar 05, 2017 5:32:33 PM com.amazonaws.services.kinesis.clientlibrary.config.KinesisClientLibConfigurator getConfiguration
INFO: Value of workerId is not provided in the properties. WorkerId is automatically assigned as: [redacted]
Mar 05, 2017 5:32:34 PM com.amazonaws.services.kinesis.clientlibrary.config.KinesisClientLibConfigurator withProperty
INFO: Successfully set property regionName with value us-east-1
Mar 05, 2017 5:32:34 PM com.amazonaws.services.kinesis.multilang.MultiLangDaemonConfig buildExecutorService
INFO: Using a cached thread pool.
Mar 05, 2017 5:32:34 PM com.amazonaws.services.kinesis.multilang.MultiLangDaemonConfig <init>
INFO: Running AWR-demo-app to process stream test-AWR with executable /app/app.R
Mar 05, 2017 5:32:34 PM com.amazonaws.services.kinesis.multilang.MultiLangDaemonConfig prepare
INFO: Using workerId: [redacted]
Mar 05, 2017 5:32:34 PM com.amazonaws.services.kinesis.multilang.MultiLangDaemonConfig prepare
INFO: Using credentials with access key id: [redacted]
Mar 05, 2017 5:32:34 PM com.amazonaws.services.kinesis.multilang.MultiLangDaemonConfig prepare
INFO: MultiLangDaemon is adding the following fields to the User Agent: amazon-kinesis-client-library-java-1.7.3 amazon-kinesis-multi-lang-dae
mon/1.0.1 R /app/app.R
Mar 05, 2017 5:32:34 PM com.amazonaws.services.kinesis.leases.impl.LeaseCoordinator <init>
INFO: With failover time 10000 ms and epsilon 25 ms, LeaseCoordinator will renew leases every 3308 ms, takeleases every 20050 ms, process maxi
mum of 2147483647 leases and steal 1 lease(s) at a time.
Mar 05, 2017 5:32:34 PM com.amazonaws.services.kinesis.clientlibrary.lib.worker.Worker initialize
INFO: Initialization attempt 1
Mar 05, 2017 5:32:34 PM com.amazonaws.services.kinesis.clientlibrary.lib.worker.Worker initialize
INFO: Initializing LeaseCoordinator
Mar 05, 2017 5:32:35 PM com.amazonaws.services.kinesis.clientlibrary.lib.worker.Worker initialize
INFO: Syncing Kinesis shard info
Mar 05, 2017 5:32:36 PM com.amazonaws.services.kinesis.clientlibrary.lib.worker.Worker initialize
INFO: Starting LeaseCoordinator
Mar 05, 2017 5:32:36 PM com.amazonaws.services.kinesis.leases.impl.LeaseTaker computeLeasesToTake
INFO: Worker [redacted] needed 2 leases but none were expired, so it will steal lease shardId-000000000002 from bef5
4447-3adb-444f-8dc6-67504e5c86ef
Mar 05, 2017 5:32:36 PM com.amazonaws.services.kinesis.leases.impl.LeaseTaker computeLeasesToTake
INFO: Worker [redacted] saw 3 total leases, 0 available leases, 2 workers. Target is 2 leases, I have 0 leases, I wi
ll take 1 leases
Mar 05, 2017 5:32:36 PM com.amazonaws.services.kinesis.leases.impl.LeaseTaker takeLeases
INFO: Worke[redacted] successfully took 1 leases: shardId-000000000002
Mar 05, 2017 5:32:46 PM com.amazonaws.services.kinesis.clientlibrary.lib.worker.Worker run
INFO: Initialization complete. Starting worker loop.
Mar 05, 2017 5:32:46 PM com.amazonaws.services.kinesis.clientlibrary.lib.worker.Worker infoForce
INFO: Created new shardConsumer for : ShardInfo [shardId=shardId-000000000002, concurrencyToken=[redacted], parentSh
ardIds=[shardId-000000000000], checkpoint={SequenceNumber: TRIM_HORIZON,SubsequenceNumber: 0}]
Mar 05, 2017 5:32:46 PM com.amazonaws.services.kinesis.clientlibrary.lib.worker.BlockOnParentShardTask call
INFO: No need to block on parents [shardId-000000000000] of shard shardId-000000000002
Mar 05, 2017 5:32:47 PM com.amazonaws.services.kinesis.clientlibrary.lib.worker.KinesisDataFetcher initialize
```

This Kinesis app is being run

```
library(futile.logger)
library(AWR.Kinesis)

kinesis_consumer(

    initialize     = function()
        flog.info('Hello'),

    processRecords = function(records)
        flog.info(paste('Received', nrow(records), 'records from Kinesis')),

    shutdown       = function()
        flog.info('Bye'),

    updater        = list(
        list(1, function()
            flog.info('Updating some data every minute')),
        list(1/60*10, function()
            flog.info(paste(
                'This is a high frequency updater call',
                'running every 10 seconds')))),

    checkpointing = 1,
    logfile = '/logs/logger.log')
```

Running the MultiLangDaemon locally

CARD.COM

Terminix: Default

1: ec2-user@[redacted]:~

```
[ec2-user@[redacted] logs]$ head -n 44 logger.log
INFO [2017-03-05 03:35:23] Starting R Kinesis Consumer application
INFO [2017-03-05 03:35:23 UTC] shardId-000000000000 Start of initialize
INFO [2017-03-05 03:35:23 UTC] shardId-000000000000 Hello
INFO [2017-03-05 03:35:23 UTC] shardId-000000000000 End of initialize
INFO [2017-03-05 03:35:23 UTC] shardId-000000000000 Received 2 records from Kinesis
INFO [2017-03-05 03:35:24 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:25 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:26 UTC] shardId-000000000000 Received 2 records from Kinesis
INFO [2017-03-05 03:35:27 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:28 UTC] shardId-000000000000 Received 2 records from Kinesis
INFO [2017-03-05 03:35:29 UTC] shardId-000000000000 Received 2 records from Kinesis
INFO [2017-03-05 03:35:30 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:31 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:32 UTC] shardId-000000000000 Received 2 records from Kinesis
INFO [2017-03-05 03:35:33 UTC] shardId-000000000000 Received 2 records from Kinesis
INFO [2017-03-05 03:35:33 UTC] shardId-000000000000 This is a high frequency updater call running every 10 seconds
INFO [2017-03-05 03:35:34 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:35 UTC] shardId-000000000000 Received 2 records from Kinesis
INFO [2017-03-05 03:35:36 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:37 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:38 UTC] shardId-000000000000 Received 2 records from Kinesis
INFO [2017-03-05 03:35:39 UTC] shardId-000000000000 Received 2 records from Kinesis
INFO [2017-03-05 03:35:40 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:41 UTC] shardId-000000000000 Received 2 records from Kinesis
INFO [2017-03-05 03:35:42 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:43 UTC] shardId-000000000000 Received 2 records from Kinesis
INFO [2017-03-05 03:35:43 UTC] shardId-000000000000 This is a high frequency updater call running every 10 seconds
INFO [2017-03-05 03:35:44 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:45 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:46 UTC] shardId-000000000000 Received 2 records from Kinesis
INFO [2017-03-05 03:35:47 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:48 UTC] shardId-000000000000 Received 2 records from Kinesis
INFO [2017-03-05 03:35:49 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:50 UTC] shardId-000000000000 Received 2 records from Kinesis
INFO [2017-03-05 03:35:51 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:52 UTC] shardId-000000000000 Received 2 records from Kinesis
INFO [2017-03-05 03:35:53 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:54 UTC] shardId-000000000000 Received 2 records from Kinesis
INFO [2017-03-05 03:35:54 UTC] shardId-000000000000 This is a high frequency updater call running every 10 seconds
INFO [2017-03-05 03:35:55 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:56 UTC] shardId-000000000000 Received 2 records from Kinesis
INFO [2017-03-05 03:35:57 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:58 UTC] shardId-000000000000 Received 2 records from Kinesis
INFO [2017-03-05 03:35:59 UTC] shardId-000000000000 Received 3 records from Kinesis
[ec2-user@ip[redacted] logs]$
```

Let's run it in AWS!

CARD.COM

1. Dockerize your Kinesis Consumer:
   - Java
   - R
   - AWR, AWR.Kinesis packages
   - app.R
   - app.properties
   - startup command
2. Put it on Docker Hub
3. Run as a EC2 Container Service Task:
   - Create an ECS cluster
   - Create ECS Task Role
   - Create a Task definition
   - Run it (as a service)

Dockerize your Kinesis Consumer

CARD.COM

card-rocker/Docke

GitHub, Inc. [US] | https://github.com/cardcorp/card-rocker/blob/master/r-kinesis/Dockerfile

Features Explore Pricing This repository Search Sign in or Sign up

cardcorp / card-rocker

Watch 13 Star 6 Fork 2

Code Issues 0 Pull requests 0 Projects 0 Pulse Graphs

Branch: master card-rocker / r-kinesis / Dockerfile Find file Copy path

daroczig break long lines 3bf40ac 26 seconds ago

1 contributor

18 lines (14 sloc) 606 Bytes Raw Blame History

```
FROM cardcorp/r-aws-java-pandoc:latest
MAINTAINER Gergely Daroczi <gergely.daroczi@card.com>

## Install extra AWR packages
RUN install2.r --error \
    AWR.KMS \
    AWR.Kinesis \
  && rm -rf /tmp/downloaded_packages/ /tmp/*.rds

## Run MultiLangDaemon on /app
ENTRYPOINT ["/usr/bin/java", \
            "-cp", \
            "/usr/local/lib/R/site-library/AWR/java/*:/usr/local/lib/R/site-library/AWR.Kinesis/java/*", \
            "com.amazonaws.services.kinesis.multilang.MultiLangDaemon"]

## Override this if the consumer app is mounted elsewhere or the config file has a different name
CMD ["/app/app.properties"]
```

© 2017 GitHub, Inc. Terms Privacy Security Status Help Contact GitHub API Training Shop Blog About

Dockerize your Kinesis Consumer

CARD.COM

card-rocker/Docke

GitHub, Inc. [US] | https://github.com/cardcorp/card-rocker/blob/master/r-kinesis-example/Dockerfile

Features Explore Pricing

This repository Search

Sign in or Sign up

cardcorp / card-rocker

Watch 13 Star 6 Fork 2

Code Issues 0 Pull requests 0 Projects 0 Pulse Graphs

Branch: master card-rocker / r-kinesis-example / Dockerfile

Find file Copy path

daroczig add example Kinesis consumer using R 0fe8aac a day ago

1 contributor

6 lines (4 sloc) | 118 Bytes

Raw Blame History

```
FROM cardcorp/r-kinesis:latest
MAINTAINER Gergely Daroczi <gergely.daroczi@card.com>

## Add consumer
COPY files /app
```

© 2017 GitHub, Inc. Terms Privacy Security Status Help

Contact GitHub API Training Shop Blog About

Dockerize your Kinesis Consumer

CARD.COM

card-rocker/app.R

GitHub, Inc. [US] | https://github.com/cardcorp/card-rocker/blob/master/r-kinesis-example/files/app.R

Features Explore Pricing This repository Search Sign in or Sign up

cardcorp / card-rocker

Watch 13 Star 6 Fork 2

Code Issues 0 Pull requests 0 Projects 0 Pulse Graphs

Branch: master card-rocker / r-kinesis-example / files / app.R Find file Copy path

daroczig run demo in east region + log in central folder 2b9e5e8 22 hours ago

1 contributor

Executable File 24 lines (17 sloc) 619 Bytes Raw Blame History

```
#!/usr/bin/r --vanilla
library(futile.logger)
library(AWR.Kinesis)

kinesis_consumer(

    initialize     = function()
        flog.info('Hello'),

    processRecords = function(records)
        flog.info(paste('Received', nrow(records), 'records from Kinesis')),

    shutdown       = function()
        flog.info('Bye'),

    updater        = list(
        list(1, function()
            flog.info('Updating some data every minute')),
        list(1/60*10, function()
            flog.info('This is a high frequency updater call running every 10 seconds'))),

    checkpointing = 1,
    logfile = '/logs/logger.log')
```

Dockerize your Kinesis Consumer

CARD.COM

card-rocker/app.pr

GitHub, Inc. [US] | https://github.com/cardcorp/card-rocker/blob/master/r-kinesis-example/files/app.properties

Features Explore Pricing This repository Search Sign in or Sign up

cardcorp / card-rocker

Watch 13 Star 6 Fork 2

Code Issues 0 Pull requests 0 Projects 0 Pulse Graphs

Branch: master card-rocker / r-kinesis-example / files / app.properties Find file Copy path

daroczig run demo in east region + log in central folder 2b9e5e8 23 hours ago

1 contributor

7 lines (6 sloc) | 187 Bytes Raw Blame History

```
executableName = /app/app.R
streamName = test-AWR
applicationName = AWR-demo-app
AWSCredentialsProvider = DefaultAWSCredentialsProviderChain
processingLanguage = R
regionName = us-east-1
```

© 2017 GitHub, Inc. Terms Privacy Security Status Help Contact GitHub API Training Shop Blog About

Put it on Docker Hub

CARD.COM

cardcorp/r-kinesis-

https://hub.docker.com/r/cardcorp/r-kinesis-example/builds/

Search

Explore Help Sign up Sign in

PUBLIC | AUTOMATED BUILD

cardcorp/r-kinesis-example

Last pushed: a day ago

Repo Info Tags Dockerfile Build Details

Status	Actions	Tag	Created	Last Updated
Building	Cancel	latest	3 minutes ago	a minute ago
Success		latest	a day ago	a day ago
Success		latest	a day ago	a day ago
Success		latest	a day ago	a day ago
Success		latest	a day ago	a day ago
Success		latest	a day ago	a day ago
Success		latest	a day ago	a day ago
Success		latest	a day ago	a day ago

Source Repository

cardcorp/card-rocker

Create an ECS cluster

CARD.COM

Amazon EC2 Conta

https://console.aws.amazon.com/ecs/home?region=us-east-1#/clusters/create/new

Services | Resource Groups | Gergely Daróczi | N. Virginia | Support

Amazon ECS
- Clusters
- Task Definitions
- Repositories

Create Cluster

When you run tasks using Amazon ECS, you place them on a cluster, which is a logical grouping of EC2 instances. This wizard will guide you through the process to create a cluster . You will name your cluster, and then configure the container instances that your tasks can be placed on, the security group for your container instances to use, and the IAM role to associate with your container instances so that they can make calls to the AWS APIs on your behalf.

Cluster name* AWR-test

Create an empty cluster

EC2 instance type* t2.medium

Number of instances* 1

EC2 Ami Id* amzn-ami-2016.09.f-amazon-ecs-optimized [ami-b2df2ca4]

EBS storage (GiB)* 22

Key pair

You will not be able to SSH into your EC2 instances without a key pair. You can create a new key pair in the EC2 console .

Networking

Configure the VPC for your container instances to use. A VPC is an isolated portion of the AWS cloud populated by AWS objects, such as Amazon EC2 instances. You can choose an existing VPC, or create a new one with this wizard.

VPC Create a new vpc

CIDR Block 10.0.0.0/16

Create ECS Task Role

CARD.COM

IAM Management

https://console.aws.amazon.com/iam/home?region=us-east-1#/roles

Services | Resource Groups | Gergely Daróczi | Global | Support

Create Role

Step 1 : Set Role Name

Step 2 : Select Role Type

Step 3 : Establish Trust

Step 4 : Attach Policy

Step 5 : Review

Set Role Name

Enter a role name. You cannot edit the role name after the role is created.

Role Name AWR-ECS

Maximum 64 characters. Use alphanumeric and '+=,.@-_' characters

Cancel | Next Step

Create ECS Task Role

CARD.COM

IAM Management

https://console.aws.amazon.com/iam/home?region=us-east-1#/roles

Services | Resource Groups

Create Role

Step 1 : Set Role Name

Step 2 : Select Role Type

Step 3 : Establish Trust

Step 4 : Attach Policy

Step 5 : Review

Select Role Type

AWS Service Roles

Amazon EC2 Role for EC2 Container Service
Role to allow EC2 instances in an Amazon ECS cluster to access Amazon ECS. — Select

Amazon EC2 Container Service Role
Allows ECS to create and manage AWS resources on your behalf. — Select

Amazon EC2 Container Service Task Role
Allows ECS tasks to call AWS services on your behalf. — Select

Amazon EC2 Spot Fleet Role
Role to Allow EC2 Spot Fleet to request and terminate Spot Instances on your behalf. — Select

Amazon Elastic MapReduce
Role to allow EMR to access other AWS services such as EC2 on your behalf. — Select

Role for Cross-Account Access

Role for Identity Provider Access

Cancel | Previous | Next Step

Create ECS Task Role

CARD.COM

IAM Management

https://console.aws.amazon.com/iam/home?region=us-east-1#/roles

Services | Resource Groups

Create Role

- Step 1 : Set Role Name
- Step 2 : Select Role Type
- Step 3 : Establish Trust
- **Step 4 :** Attach Policy
- Step 5 : Review

Attach Policy

Select one or more policies to attach. Each role can have up to 10 policies attached.

Filter: Policy Type kinesis — Showing 7 results

	Policy Name	Attached Entities	Creation Time	Edited Time
☐	AmazonKinesisFullAccess	1	2015-02-06 10:40 PST	2015-02-06 10:40 PST
☑	AmazonKinesisReadOnlyA..	1	2015-02-06 10:40 PST	2015-02-06 10:40 PST
☐	AmazonKinesisAnalyticsF...	0	2016-09-21 12:01 PST	2016-09-21 12:01 PST
☐	AmazonKinesisAnalyticsR...	0	2016-09-21 11:16 PST	2016-09-21 11:16 PST
☐	AmazonKinesisFirehoseFu..	0	2015-10-07 11:45 PST	2015-10-07 11:45 PST
☐	AmazonKinesisFirehoseR...	0	2015-10-07 11:43 PST	2015-10-07 11:43 PST
☐	AWSLambdaKinesisExecu...	0	2015-04-09 08:14 PST	2015-04-09 08:14 PST

Cancel | Previous | Next Step

Create a Task definition

CARD.COM

Amazon EC2 Conta

https://console.aws.amazon.com/ecs/home?region=us-east-1#/taskDefinitions/create

Services | Resource Groups | Gergely Daróczi | N. Virginia | Support

Amazon ECS
- Clusters
- Task Definitions
- Repositories

Create a Task Definition

A task definition specifies which containers are included in your task and how they interact with each other. You can also specify data volumes for your containers to use. Learn more

Task Definition Name* AWR-logger

Task Role AWR-ECS

Optional IAM role that tasks can use to make API requests to authorized AWS services. Create an Amazon EC2 Container Service Task Role in the IAM Console

Network Mode Bridge

Constraint

Constraints allow you to filter the instances used for your placement strategies using built-in or custom attributes. The scheduler first filters the instances that match the constraints and then applies the placement strategy to place the task.

Type	Expression

Add constraint

Container Definitions

Add container

Container Name	Image	Hard/Soft memory limits (MB)	Essential
No results			

Volumes

Name	Source Path
No results	

Add volume

Configure via JSON

Create a Task definition

CARD.COM

Amazon EC2 Conta

https://console.aws.amazon.com/ecs/home?region=us-east-1#/taskDefinitions/create

Services | Resource Groups | Gergely Daróczi | N. Virginia | Support

Amazon ECS
- Clusters
- Task Definitions
- Repositories

Create a Task Definition

A task de ... h other. You can also specify data volumes for your ...

Add volume

Name* logs

Source path /logs

*Required | Cancel | Add

Constraint

Constraints allow you to filter the instances used for your placement strategies using built-in or custom attributes. The scheduler first filters the instances that match the constraints and then applies the placement strategy to place the task.

Type	Expression

Add constraint

Container Definitions

Add container

Container Name	Image	Hard/Soft memory limits (MB)	Essential
No results			

Volumes

Name	Source Path
No results	

Add volume

Configure via JSON

Create a Task definition

CARD.COM

Amazon EC2 Conta

https://console.aws.amazon.com/ecs/home?region=us-east-1#/taskDefinitions/create

Services | Resource Groups

Amazon ECS
- Clusters
- Task Definitions
- Repositories

Create a Ta

A task definition specifi
for your containers to u

Constraint

Constraints allow you t
instances that match th

Type

Add constraint

Container Definitio

Add container

Container Name

Volumes

Name

logs

Add volume

Configure via JSON

Add container

Standard

Container name* logger

Image* cardcorp/r-kinesis-example:latest

Custom image format: [registry-url]/[namespace]/[image]:[tag]

Memory Limits (MB)* Hard limit 512

Add Soft limit

Define hard and/or soft memory limits in MiB for your container. Hard and soft limits correspond to the `memory` and `memoryReservation` parameters, respectively, in task definitions.
ECS recommends 300-500 MB as a starting point for web applications.

Port mappings Host port Container port Protocol tcp

Add port mapping

Advanced container configuration

ENVIRONMENT

CPU units

The number of cpu units to reserve for the container. A container instance has 1,024 cpu units for every CPU core.

Essential

* Required

Cancel Add

Create a Task definition

CARD.COM

Amazon EC2 Conta

https://console.aws.amazon.com/ecs/home?region=us-east-1#/taskDefinitions/create

Services | Resource Groups

Amazon ECS
- Clusters
- Task Definitions
- Repositories

Create a Ta

A task definition specifi
for your containers to u

Constraint

Constraints allow you t
instances that match th

Type

Add constraint

Container Definitio

Add container

Container Name

Volumes

Name

logs

Add volume

Configure via JSON

Add container

IP address

Add extra host

STORAGE AND LOGGING

When this parameter is true, the container is given read-only access to its root file system.

Read only root file system

Mount points

Source volume: logs

Container path: /logs

Read only

Add mount point

Volumes from

Source container

Read only

Add volumes

Log configuration

Log driver: <none>

Log options

Key: Add key

Value: Add value

SECURITY

* Required

Cancel | Add

Run the ECS Task

CARD.COM

Amazon EC2 Conta

https://console.aws.amazon.com/ecs/home?region=us-east-1#/taskDefinitions/AWR-logger/status/ACTIVE

Services | Resource Groups | Gergely Daróczi | N. Virginia | Support

Amazon ECS
- Clusters
- Task Definitions
- Repositories

Task Definitions > AWR-logger > status > ACTIVE

Task Definition Name : AWR-logger

Select a revision for more details

Create new revision | Actions | Last updated on March 5, 2017 6:00:27 PM (0m ago)

- Run Task
- Create Service
- Update Service
- Deregister

Status: Active Inactive

Filter in this page | 1-1 | Page size 50

Task Definition Nar	Status
AWR-logger:1	Active

Feedback | English | © 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved. | Privacy Policy | Terms of Use

Run the ECS Task

CARD.COM

Amazon EC2 Conta

https://console.aws.amazon.com/ecs/home?region=us-east-1#/clusters/AWR/tasks

Services | Resource Groups | Gergely Daróczi | N. Virginia | Support

Amazon ECS
- Clusters
- Task Definitions
- Repositories

Clusters > AWR

Cluster : AWR

Delete Cluster

Get a detailed view of the resources on your cluster.

Status	ACTIVE
Registered container instances	1
Pending tasks count	0
Running tasks count	1

Services | Tasks | ECS Instances | Metrics

Run new Task | Stop | Stop All

Last updated on March 5, 2017 6:05:01 PM (0m ago)

Desired task status: Running Stopped

Filter in this page

1-1 Page size 50

Task	Task Definition	Group	Container Inst...	Last status	Desired status	Started By
0c9f224a-7808...	AWR-logger:1	family:AWR-log...	278b-4935-70e2...	RUNNING	RUNNING	

https://console.aws.amazon.com/ecs/home?region=...

Privacy Policy | Terms of Use

Run the ECS Task

CARD.COM

Terminix: Default

1: ec2-user@[redacted]:~

```
[redacted] ssh -i ~/.ssh/[redacted] ec2-user@[redacted]
Last login: Mon Mar  6 02:05:29 2017 from [redacted]

   __|  __|  __|
   _|  (   \__ \   Amazon ECS-Optimized Amazon Linux AMI 2016.09.f
 ____|\___|____/

For documentation visit, http://aws.amazon.com/documentation/ecs
4 package(s) needed for security, out of 6 available
Run "sudo yum update" to apply all updates.
[ec2-user@[redacted] ~]$ docker ps
CONTAINER ID        IMAGE                               COMMAND                  CREATED             STATUS              PORTS               N
AMES
b59981414dda        cardcorp/r-kinesis-example:latest   "/usr/bin/java -cp /u"   22 hours ago        Up 22 hours                             e
cs-AWR-logger-1-logger-92ca888bd5d681e59d01
2ed65c5a3752        amazon/amazon-ecs-agent:latest      "/agent"                 23 hours ago        Up 23 hours                             e
cs-agent
[ec2-user@[redacted] ~]$ pgrep app.R
29435
[ec2-user@[redacted] ~]$ head -n 10 /logs/logger.log
INFO [2017-03-05 03:35:23] Starting R Kinesis Consumer application
INFO [2017-03-05 03:35:23 UTC] shardId-000000000000 Start of initialize
INFO [2017-03-05 03:35:23 UTC] shardId-000000000000 Hello
INFO [2017-03-05 03:35:23 UTC] shardId-000000000000 End of initialize
INFO [2017-03-05 03:35:23 UTC] shardId-000000000000 Received 2 records from Kinesis
INFO [2017-03-05 03:35:24 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:25 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:26 UTC] shardId-000000000000 Received 2 records from Kinesis
INFO [2017-03-05 03:35:27 UTC] shardId-000000000000 Received 3 records from Kinesis
INFO [2017-03-05 03:35:28 UTC] shardId-000000000000 Received 2 records from Kinesis
[ec2-user@[redacted] ~]$
```

Scaling the Kinesis Consumer up

CARD.COM

Terminix: Default

1: ec2-user@[redacted]:~

```
INFO [2017-03-05 03:43:01 UTC] shardId-000000000000 Received 1 records from Kinesis
INFO [2017-03-05 03:43:01 UTC] shardId-000000000000 This is a high frequency updater call running every 10 seconds
INFO [2017-03-05 03:43:09 UTC] shardId-000000000000 Shutting down
INFO [2017-03-05 03:43:09 UTC] shardId-000000000000 Bye
INFO [2017-03-05 03:43:15] Starting R Kinesis Consumer application
INFO [2017-03-05 03:43:15 UTC] shardId-000000000002 Start of initialize
INFO [2017-03-05 03:43:15 UTC] shardId-000000000002 Hello
INFO [2017-03-05 03:43:15 UTC] shardId-000000000002 End of initialize
INFO [2017-03-05 03:43:15] Starting R Kinesis Consumer application
INFO [2017-03-05 03:43:16 UTC] shardId-000000000001 Start of initialize
INFO [2017-03-05 03:43:16 UTC] shardId-000000000001 Hello
INFO [2017-03-05 03:43:16 UTC] shardId-000000000001 End of initialize
INFO [2017-03-05 03:44:32 UTC] shardId-000000000002 Received 1 records from Kinesis
INFO [2017-03-05 03:44:32 UTC] shardId-000000000002 Updating some data every minute
INFO [2017-03-05 03:44:32 UTC] shardId-000000000002 This is a high frequency updater call running every 10 seconds
INFO [2017-03-05 03:44:33 UTC] shardId-000000000002 Received 3 records from Kinesis
INFO [2017-03-05 03:44:34 UTC] shardId-000000000002 Received 1 records from Kinesis
INFO [2017-03-05 03:44:35 UTC] shardId-000000000002 Received 1 records from Kinesis
INFO [2017-03-05 03:44:36 UTC] shardId-000000000001 Received 2 records from Kinesis
INFO [2017-03-05 03:44:36 UTC] shardId-000000000002 Received 1 records from Kinesis
INFO [2017-03-05 03:44:36 UTC] shardId-000000000001 Updating some data every minute
INFO [2017-03-05 03:44:36 UTC] shardId-000000000001 This is a high frequency updater call running every 10 seconds
INFO [2017-03-05 03:44:37 UTC] shardId-000000000001 Received 1 records from Kinesis
INFO [2017-03-05 03:44:38 UTC] shardId-000000000001 Received 1 records from Kinesis
INFO [2017-03-05 03:44:39 UTC] shardId-000000000002 Received 1 records from Kinesis
INFO [2017-03-05 03:44:39 UTC] shardId-000000000001 Received 2 records from Kinesis
INFO [2017-03-05 03:44:40 UTC] shardId-000000000001 Received 1 records from Kinesis
INFO [2017-03-05 03:44:40 UTC] shardId-000000000002 Received 1 records from Kinesis
INFO [2017-03-05 03:44:41 UTC] shardId-000000000001 Received 2 records from Kinesis
INFO [2017-03-05 03:44:42 UTC] shardId-000000000001 Received 2 records from Kinesis
INFO [2017-03-05 03:44:43 UTC] shardId-000000000002 Received 2 records from Kinesis
INFO [2017-03-05 03:44:43 UTC] shardId-000000000002 This is a high frequency updater call running every 10 seconds
INFO [2017-03-05 03:44:44 UTC] shardId-000000000002 Received 1 records from Kinesis
INFO [2017-03-05 03:44:45 UTC] shardId-000000000002 Received 2 records from Kinesis
INFO [2017-03-05 03:44:45 UTC] shardId-000000000001 Received 1 records from Kinesis
INFO [2017-03-05 03:44:46 UTC] shardId-000000000002 Received 1 records from Kinesis
INFO [2017-03-05 03:44:46 UTC] shardId-000000000001 Received 1 records from Kinesis
INFO [2017-03-05 03:44:47 UTC] shardId-000000000001 Received 1 records from Kinesis
INFO [2017-03-05 03:44:47 UTC] shardId-000000000001 This is a high frequency updater call running every 10 seconds
INFO [2017-03-05 03:44:47 UTC] shardId-000000000002 Received 1 records from Kinesis
INFO [2017-03-05 03:44:48 UTC] shardId-000000000001 Received 1 records from Kinesis
INFO [2017-03-05 03:44:48 UTC] shardId-000000000002 Received 1 records from Kinesis
INFO [2017-03-05 03:44:49 UTC] shardId-000000000001 Received 1 records from Kinesis
INFO [2017-03-05 03:44:49 UTC] shardId-000000000002 Received 1 records from Kinesis
INFO [2017-03-05 03:44:50 UTC] shardId-000000000001 Received 1 records from Kinesis
:
```

Kinesis Consumers in Production

CARD.COM

Nice example project, but . . .

- I might want to avoid publishing my Consumer on Docker Hub
- I might want to avoid publishing my code on GitHub
- I might want to avoid commiting credentials etc to the repo

Problems:

- How to store credentials in the Docker images?
- Where to store the Docker images?

Kinesis Consumers in Production

CARD.COM

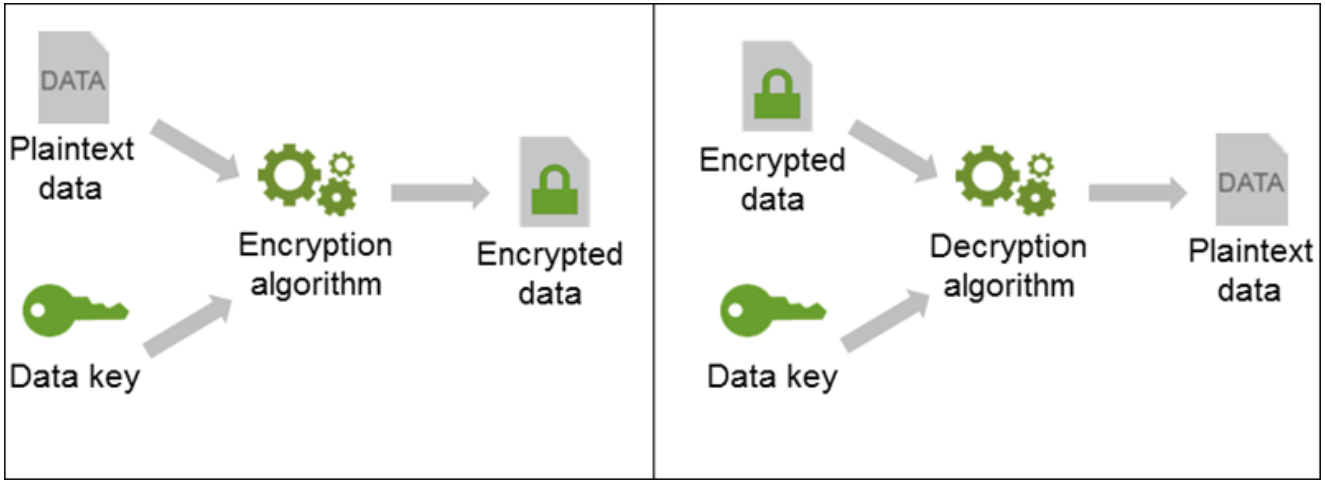
Nice example project, but . . .

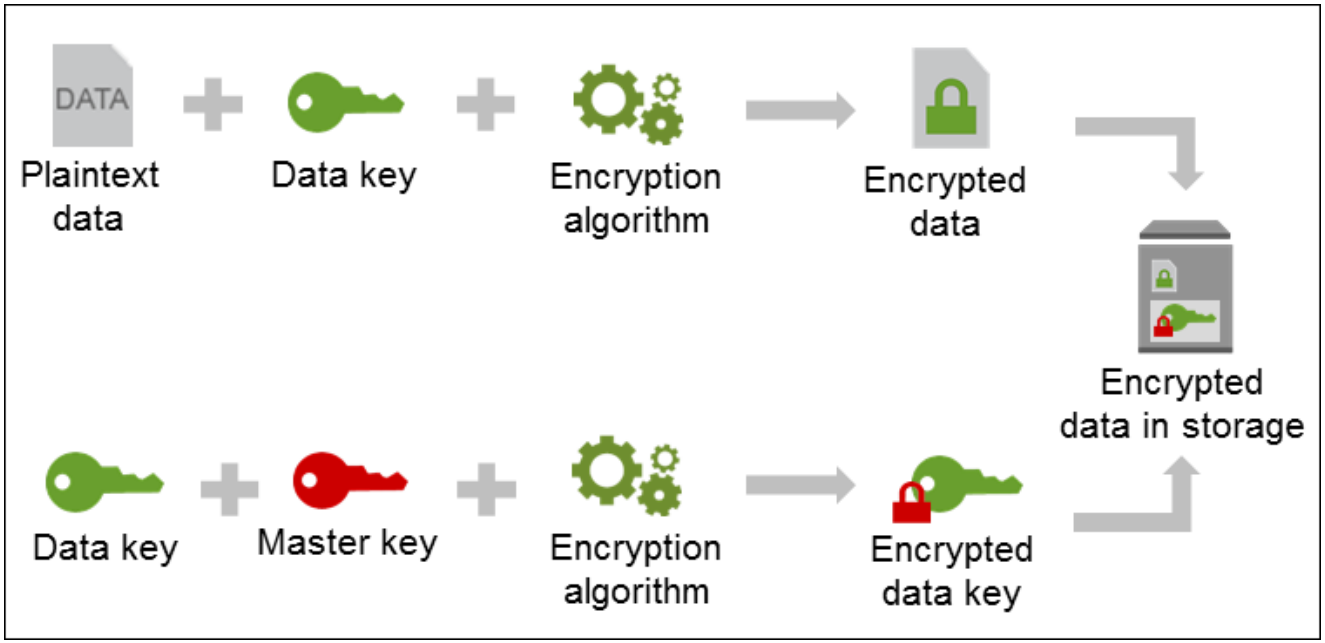
- I might want to avoid publishing my Consumer on Docker Hub
- I might want to avoid publishing my code on GitHub
- I might want to avoid commiting credentials etc to the repo

Problems:

- How to store credentials in the Docker images? **KMS**
- Where to store the Docker images? **ECR**

KMS





Source: AWS Encryption SDK

Current AWR.KMS Features

CARD.COM

- encrypt up to 4 KB of arbitrary data:

```
> library(AWR.KMS)
> kms_encrypt('alias/mykey', 'foobar')
[1] "Base-64 encoded ciphertext"
```

- decrypt such Base-64 encoded ciphertext back to plaintext:

```
> kms_encrypt('Base-64 encoded ciphertext')
[1] "foobar"
```

- generate a data encryption key:

```
> kms_generate_data_key('alias/mykey')
$cipher
[1] "Base-64 encoded, encrypted data encryption key"
$key
[1] "alias/mykey"
$text
[1] 00 01 10 11 00 01 10 11 ...
```

Encrypting Data Larger Than 4 KB?

CARD.COM

```
## let's say we want to encrypt the mtcars dataset stored in JSON
library(jsonlite)
data <- toJSON(mtcars)

## generate a 256-bit data encryption key (that's supported by digest::AES)
library(AWR.KMS)
key <- kms_generate_data_key('alias/mykey', byte = 32L)

## convert the JSON to raw so that we can use that with digest::AES
raw <- charToRaw(data)
## the text length must be a multiple of 16 bytes
## https://github.com/sdoyen/r_password_crypt/blob/master/crypt.R
raw <- c(raw, as.raw(rep(0, 16 - length(raw) %% 16)))

## encrypt the raw object with the new key + digest::AES
## the resulting text and the encrypted key can be stored on disk
library(digest)
aes <- AES(key$text)
base64_enc(aes$encrypt(raw))

## decrypt the above returned ciphertext using the decrypted key
rawToChar(aes$decrypt(base64_dec(...), raw = TRUE))
```

Example "Production" Consumer App

CARD.COM

```
library(AWR.Kinesis); library(jsonlite); library(AWR.KMS); library(futile.logger); flog.threshold(DEBUG)

kinesis_consumer(
    initialize     = function() {
        flog.info('Decrypting Redis hostname via KMS')
        host <- kms_decrypt('AQECAHiiz4GEPFQLL9AAON5TY/lDR5euQQScpXQU9iYTn+u...')
        flog.info('Connecting to Redis')
        library(rredis); redisConnect(host = host)
        flog.info('Connected to Redis')
    },
    processRecords = function(records) {
        flog.info(paste('Received', nrow(records), 'records from Kinesis'))
        for (record in records$data) {
            flight <- fromJSON(record)$dest
            if (!is.null(flight)) {
                flog.debug(paste('Adding +1 to', flight))
                redisIncr(sprintf('flight:%s', flight))
            } else {
                flog.error('Flight destination not found')
            }
        }
    },
    updater = list(
        list(1/6, function() {
            flog.info('Checking overall counters')
            flights <- redisKeys('flight:*')
            for (flight in flights) {
                flog.debug(paste('Found', redisGet(flight), sub('^flight:', '', flight)))
            }
        })),
    logfile = '/logs/redis.log')
```

Private Docker Image

CARD.COM

Dockerfile:

```
FROM cardcorp/r-kinesis:latest
MAINTAINER Gergely Daroczi <gergely.daroczi@card.com>

## Install R package to interact with Redis
RUN install2.r --error rredis  && rm -rf /tmp/downloaded_packages/ /tmp/*.rds

## Add consumer
COPY files /app
```

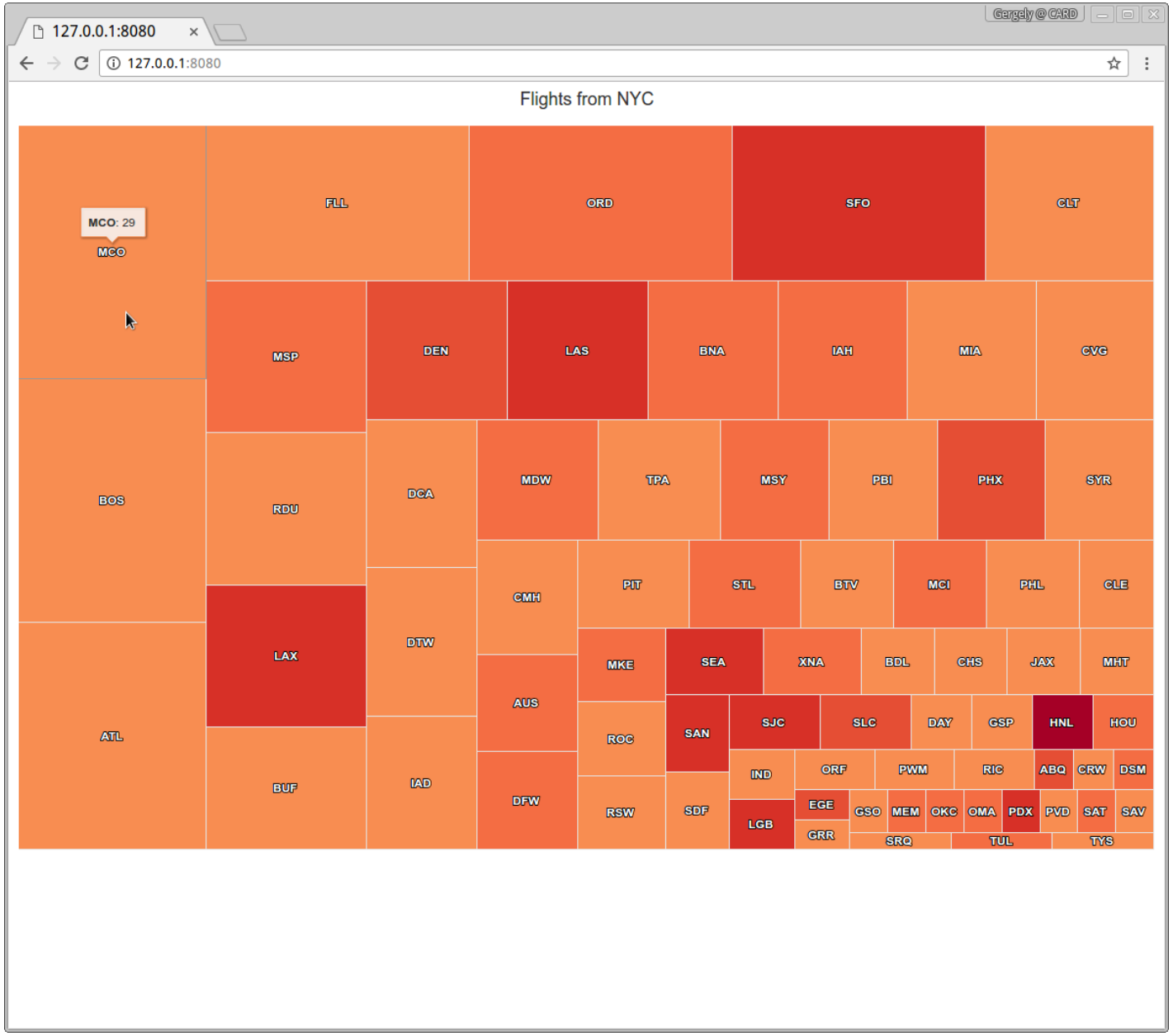
Build and push to ECR:

```
docker build -t cardcorp/r-kinesis-secret .
`aws ecr get-login --region us-east-1`
docker tag -f cardcorp/r-kinesis-secret:latest \
            ***.dkr.ecr.us-east-1.amazonaws.com/cardcorp/r-kinesis-secret:latest
docker push ***.dkr.ecr.us-east-1.amazonaws.com/cardcorp/r-kinesis-secret:latest
```

Shiny Dashboard

```
library(treemap);library(highcharter);library(nycflights13)
library(rredis);redisConnect(host = '***', port = '***')

ui     <- shinyUI(highchartOutput('treemap', height = '800px'))
server <- shinyServer(function(input, output, session) {

    destinations <- reactive({
        reactiveTimer(2000)()
        flights <- redisMGet(redisKeys('flight:*'))
        flights <- data.frame(faa = sub('^flight:', '', names(flights)),
                              N   = as.numeric(flights))
        merge(flights, airports, by = 'faa')
    })

    output$treemap <- renderHighchart({
        tm <- treemap(destinations(), index = c('faa'),
                      vSize = 'N', vColor = 'tz',
                      type = 'value', draw = FALSE)
        hc_title(hctreemap(tm, animation = FALSE), text = 'Flights from NYC')
    })

})
shinyApp(ui = ui, server = server)
```

Shiny Dashboard



Technical Details

- AWR repo:
  - 6.3 GB
  - 273 tags/versions
  - GitLab + CI + drat

```
install.packages('AWR', repos = 'https://cardcorp.gitlab.io/AWR')
```

Technical Details

- AWR repo:
  - 6.3 GB
  - 273 tags/versions
  - GitLab + CI + drat

```
install.packages('AWR', repos = 'https://cardcorp.gitlab.io/AWR')
```

- Submitted to CRAN on
  - 2016-12-05

Technical Details

- AWR repo:
  - 6.3 GB
  - 273 tags/versions
  - GitLab + CI + drat

```
install.packages('AWR', repos = 'https://cardcorp.gitlab.io/AWR')
```

- Submitted to CRAN on
  - 2016-12-05
  - 2017-01-09
  - 2017-01-10
  - 2017-01-11
  - 2017-01-11
  - 2017-01-13

Technical Details

CARD.COM

- AWR repo:
  - 6.3 GB
  - 273 tags/versions
  - GitLab + CI + drat

```
install.packages('AWR', repos = 'https://cardcorp.gitlab.io/AWR')
```

- Submitted to CRAN on
  - 2016-12-05
  - 2017-01-09
  - 2017-01-10
  - 2017-01-11
  - 2017-01-11
  - 2017-01-13
- Release cycle: 2 minor, ~125 patch versions in the past 12 months
- CI

What's Next?

CARD.COM


```
> library(rJava)
> kc <- .jnew('com.amazonaws.services.s3.AmazonS3Client')
> kc$getS3AccountOwner()$getDisplayName()
[1] "foobar"
```


AWR
Because "S"
is so 1992.