Hybrid Cloud: Integrating AWS services with OpenShift

Derek Whatley
Software Engineer, Red Hat

Mandus Momberg
Solutions Architect, Amazon

May 8, 2018
Amazon Web Services & OpenShift

Stronger Together
Reliable, scalable cloud computing

Provides hundreds of web services

Offerings in 18 geographic regions

Pay only for what you use

Container Platform by Red Hat
  - Kubernetes orchestration
  - Powerful web UI
  - Developer centric tools

Runs in ANY environment

OpenShift

AWS
Combining the strengths of OpenShift and AWS:
- Run containerized apps with OpenShift
- Attach AWS services to your apps as needed
- Instant scalability and reliability
- Made possible with the AWS Service Broker
Introducing the **AWS Service Broker**

- Collaboration between AWS and Red Hat
- Enables AWS service management from the OpenShift Service Catalog
- Based on open source projects and open standards
AWS in the OpenShift Service Catalog

- Amazon DynamoDB
- Amazon EMR (APB)
- Amazon RDS (APB)
- Amazon Redshift
- Amazon Route 53 (APB)
- Amazon S3
- Amazon SNS (APB)
- Amazon SQS Queue (APB)
- Apache HTTP Server (httpd)
- CakePHP + MySQL (Persistent)
18 Supported AWS Services

- Amazon Athena (APB)
- Amazon DynamoDB (APB)
- Amazon ElastiCache (APB)
- Amazon EMR (APB)
- Amazon Kinesis
- Amazon KMS
- Amazon Lex
- Amazon Polly
- Amazon RDS (APB)
- Amazon RDS for MariaDB
- Amazon RDS for PostgreSQL
- Amazon Redshift (APB)
- Amazon Rekognition
- Amazon Route 53 (APB)
- Amazon S3 (APB)
- Amazon SNS (APB)
- Amazon SQS Queue (APB)
- Amazon Translate
Case Study: A *minimal hybrid cloud app*
Case Study: A minimal hybrid cloud app

OpenShift App

E.g. WordPress HA
Case Study: A minimal hybrid cloud app

OpenShift App

E.g. WordPress HA

AWS Service

E.g. RDS MySQL Database
Case Study: A minimal hybrid cloud app

- OpenShift App: E.g. WordPress HA
- AWS Bind Credentials: E.g. RDS MySQL username and pass
- AWS Service: E.g. RDS MySQL Database
Case Study: A minimal hybrid cloud app

*How you might be doing this today*

1. Go to OpenShift Service Catalog
   a. Create OpenShift app
2. Open AWS Web Console
   a. Open appropriate service dashboard
      i. Create AWS service
   b. Open IAM service dashboard
      i. Create IAM user
      ii. Find/write a policy giving user access to AWS service
      iii. Create AWS access keys
3. Copy AWS access keys into an OpenShift secret
4. Associate newly created secret with OpenShift app

~ 11 steps, 2 interfaces

not self-service

who provides support?
Case Study: A minimal hybrid cloud app

How you **should** be doing this today

1. Go to OpenShift Service Catalog
   a. Create OpenShift app
   b. Create AWS service

2. Attach AWS service binding to OpenShift app

~ 4 steps, 1 interface

self-service

unified support path
OpenShift App

Wordpress-HA (APB)
Amazon RDS for MySQL

OpenShift App

AWS Service
Create a binding for Amazon RDS for MySQL

Bindings create a secret containing the necessary information for an application to use this service.

- Create a secret in `wordpress-high-availability` to be used later
  - Secrets can be referenced later from an application.
- Do not bind at this time
  - Bindings can be created later from within a project.

[Create button]
What is a Service Broker?
Service Catalog ([kubernetes-incubator/service-catalog](https://kubernetes-incubator/service-catalog))

Where Services are Published

- Provision and manage services from a central interface
- Guides users through service creation flow
- OpenShift has a **Service Catalog** component (shown above)

What is a Service Broker?

Amazon S3 (APB)

Amazon Simple Storage Service (Amazon S3) is storage for the Internet. You can use Amazon S3 to store and retrieve any amount of data at any time, from anywhere on the web. You can accomplish these tasks using the simple and intuitive web interface of the AWS Management Console.
What is a Service Broker?

Amazon S3 (APB)

Amazon Simple Storage Service (Amazon S3) is storage for the Internet. You can use Amazon S3 to store and retrieve any amount of data at any time, from anywhere on the web. You can accomplish these tasks using the simple and intuitive web interface of the AWS Management Console or using one of the available storage interfaces.

S3 Bucket

Select a Plan
- Amazon S3 - Custom
  S3 Bucket with a custom configuration
- Amazon S3 - Production
  S3 Bucket pre-configured with production best practices

Create a binding for Amazon S3 (APB)

Bindings create a secret containing the necessary information for an application to use this service.
- Create a secret in your service broker to be used later
- Secrets can be referenced later from an application.
- Do not bind this time
- Bindings can be created later from within a project.
Service Catalog and Brokers
Exposé and Provision Services

What is a Service Broker?

OpenShift Service Catalog

SERVICE BROKERS

OPENSHIFT SERVICE CATALOG

AWS Service Broker

ANSIBLE

OPENSHIFT

AMAZON WEB SERVICES

PUBLIC CLOUD SERVICES

Ansible Playbook Bundles

OpenShift Templates

Other Services

What is a Service Broker?
Service Catalog + AWS Broker Demo
The Service Catalog + Broker Workflow

What is a Service Broker?
How we *built* the AWS Broker
Open Service Broker API in action

Interface between Service Marketplace and Brokers
Open Service Broker API

Defines an HTTP interface between service marketplaces and service brokers

Background

- Aims to standardize how services are consumed on cloud platforms
- Product of a multi-vendor working group formed in September 2016

- **Service Brokers** implement the API
  - AWS Service Broker
  - OpenShift Ansible Broker

- **Service Marketplaces** give users access Service Broker offerings.
  - OpenShift Service Catalog
AWS Broker: How It's Made
Automation Broker

- Open source project **powering the AWS Broker**
- Uses Ansible Playbook Bundles (APBs) to provision apps and services

Ansible Playbook Bundle (APB)

- Collection of named Ansible playbooks
- Each playbook handles a service management action
  - provision.yml - *install*
  - deprovision.yml - *uninstall*
- Containerized with an embedded Ansible runtime
Each AWS service has a corresponding Ansible Playbook Bundle on GitHub

These APBs contain Ansible Playbooks for managing AWS services with CloudFormation

**Ansible Playbook Bundles (APBs) for AWS**

- S3 Management Playbooks
  - **aws-servicebroker-s3** *(APB source files)*
    - AWS Service Broker deployment module for Amazon Simple Storage Service
    - Language: Shell
    - Updated on Mar 19

- RDS Management Playbooks
  - **aws-servicebroker-rds** *(More APB source files)*
    - AWS Service Broker deployment module for Amazon Relational Database Service
    - Language: Python
    - Updated 28 days ago

[https://github.com/awslabs?q=aws-servicebroker](https://github.com/awslabs?q=aws-servicebroker)
Ansible Playbook Bundles (APBs) for AWS

- Containers are built from AWS Labs GitHub repos and published on Docker Hub
- AWS Broker gets list of available AWS services from docker.io/awsservicebroker

S3 Management Playbooks

https://github.com/awslabs?q=aws-servicebroker

https://hub.docker.com/r/awsservicebroker/

https://github.com/awslabs?g=aws-servicebroker
When an OpenShift user requests an AWS service, an APB container runs to complete the task.

- User requests new S3 bucket, request sent to AWS Broker
- AWS Broker starts S3 APB container
- S3 APB container runs Ansible Playbook `provision.yml`
- AWS S3 Bucket is created
Simplify the Orchestration & Management of Kubernetes Apps

Let's make your complex services simple.

Many applications consist of multiple services, such as a database, API service, and frontend. Provisioning them as a single application in Kubernetes can be challenging, especially if one or more services run outside your cluster.

The Automation Broker, an implementation of the Open Service Broker API, works in conjunction with the Kubernetes Service Catalog. By leveraging a lightweight, container-based application definition called an Ansible Playbook Bundle (APB), it simplifies the orchestration and management of Kubernetes applications.

APBs are a method of modeling applications as a collection of Ansible Playbooks built into a portable container with an Ansible runtime. They’re designed to guide provisioning.

Ansible Playbook Bundle (APB)

Install
Uninstall
Grant
Revoke
Metadata
AWS Broker: How It's Made

OPEN SERVICE BROKER API™

AUTOMATION BROKER

AWS CloudFormation
AWS CloudFormation

- *Infrastructure as code* for AWS, written in JSON or YAML
- Used by the AWS Broker to create best-practice adherent AWS services

```yaml
AWSTemplateFormatVersion: 2010-09-09
Description: 'AWS Service Broker S3 . qs-int0fs937'
Parameters:
  ApplicationName:
    Description: =>
    Type: String
  BucketName:
    Description: =>
    Type: String
    Default: apps3bucket
```

AWS CloudFormation

**Step 1:** AWS experts write CloudFormation templates
**Step 2:** CloudFormation templates are packaged into APBs
**Step 3:** User requests an AWS Service
**Step 4:** APB container runs, uses Ansible Playbooks + CloudFormation to create service
AWS CloudFormation

Step 1: AWS experts write CloudFormation templates
Step 2: CloudFormation templates are packaged into APBs
Step 3: User requests an AWS Service
Step 4: APB container runs, uses Ansible Playbooks + CloudFormation to create service
AWS CloudFormation

Step 1: AWS experts write CloudFormation templates
Step 2: CloudFormation templates are packaged into APBs
Step 3: **User requests an AWS Service**
Step 4: APB container runs, uses Ansible Playbooks + CloudFormation to create service
AWS CloudFormation

Step 1: AWS experts write CloudFormation templates
Step 2: CloudFormation templates are packaged into APBs
Step 3: User requests an AWS Service

Step 4: APB container runs, uses Ansible Playbooks + CloudFormation to create service
AWS Broker: How It's Made

- Open Service Broker API™
- Automation Broker
- AWS CloudFormation
- AWS IAM
AWS IAM (Identity and Access Management)

- Enables *fine-grained access control* to AWS resources
- Used by AWS Broker to keep your master AWS access keys safe
IAM - Automatic Access Key Management

- OpenShift **admin** sets master AWS keys and IAM role to use for service management
  - Stored as a secret in the `aws-service-broker` namespace (see image below)

- OpenShift **users** can create AWS services without knowing master AWS keys
  - Users only have access to **scoped** AWS keys
IAM - Scoped AWS Keys

- What are scoped AWS keys?
  - Gives app permission to talk to an AWS service
  - Limited in scope of permissions granted

- Benefits of scoped AWS access keys
  - If bad actor gets keys, damage is limited
  - Enables self-service for regular users

Sample IAM policy granting access to an SNS topic

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Resource": "*",
      "Action": [
        "sns:Unsubscribe",
        "sns:ListSubscriptionsByTopic",
        "sns:GetSubscriptionAttributes",
        "sns:SetSubscriptionAttributes"
      ]
    },
    {
      "Effect": "Allow",
      "Resource": "{{ sns.stack_outputs.TopicARN }}",
      "Action": [
        "sns:Publish",
        "sns:Subscribe"
      ]
    }
  ]
}
```
IAM - Creating Scoped AWS Keys

OpenShift App

Waiting for AWS keys...
IAM - Creating Scoped AWS Keys

OpenShift App
- Waiting for AWS keys...

AWS Broker
- Privileged AWS keys
- Waiting for scoped AWS keys...
IAM - Creating Scoped AWS Keys

OpenShift App
Waiting for AWS keys...

AWS Broker
Privileged AWS keys
Waiting for scoped AWS keys...
IAM - Creating Scoped AWS Keys

OpenShift App
Waiting for AWS keys...

AWS Broker
Privileged AWS keys
Waiting for scoped AWS keys...

1. Request SQS queue

AWS CloudFormation
IAM - Creating Scoped AWS Keys

1. Request SQS queue
2. Create SQS queue

- AWS Broker
  - Privileged AWS keys
  - Waiting for scoped AWS keys...

- AWS CloudFormation
- AWS Simple Queue Service (my-sqs-1)

- OpenShift App
  - Waiting for AWS keys...
IAM - Creating Scoped AWS Keys

OpenShift App
Waiting for AWS keys...

AWS Broker
Privileged AWS keys
Waiting for scoped AWS keys...

AWS Simple Queue Service (my-sqs-1)

AWS CloudFormation

1. Request SQS queue

2. Create SQS queue

3. Return SQS queue ID (my-sqs-1)
IAM - Creating Scoped AWS Keys

1. Request SQS queue
2. Create SQS queue
3. Return SQS queue ID (my-sqs-1)
4. Create IAM user

AWS Simple Queue Service (my-sqs-1)

AWS CloudFormation

AWS IAM

OpenShift App

Waiting for AWS keys...

AWS Broker

Privileged AWS keys

Waiting for scoped AWS keys...
IAM - Creating Scoped AWS Keys

1. Request SQS queue
2. Create SQS queue
3. Return SQS queue ID
4. Create IAM user
5. Return scoped keys

AWS Simple Queue Service *(my-sqs-1)*

AWS CloudFormation

AWS IAM

Scoped AWS Keys *(my-sqs-1)*

Privileged AWS keys

Waiting for AWS keys...
IAM - Creating Scoped AWS Keys

1. Request SQS queue
2. Create SQS queue
3. Return SQS queue ID
4. Create IAM user
5. Return scoped keys
6. Bind

AWS Simple Queue Service (my-sqs-1)
AWS CloudFormation
AWS IAM
AWS Broker
Privileged AWS keys
Scoped AWS Keys (my-sqs-1)

OpenShift App
Scoped AWS Keys (my-sqs-1)
IAM - Creating Scoped AWS Keys

1. Request SQS queue
2. Create SQS queue
3. Return SQS queue ID
4. Create IAM user
5. Return scoped keys
6. Bind
7. Authenticate using Scoped AWS Keys

AWS Broker
- Privileged AWS keys
- Scoped AWS Keys (my-sqs-1)

AWS CloudFormation

AWS Simple Queue Service (my-sqs-1)

AWS IAM
- Creating Scoped AWS Keys
Installing the AWS Broker on OpenShift: 3 methods
Installing on an **existing OpenShift cluster**

- AWS provides an **OpenShift template** for installing the AWS Broker
  - Helper script creates required TLS certs and installs AWS Broker

- Install in just a few commands.

```
$ wget https://s3.amazonaws.com/[..]/deploy-awsservicebroker.template.yaml
$ wget https://s3.amazonaws.com/[..]/deploy_aws_broker.sh
$ bash deploy_aws_broker.sh
```

Installing in a development environment

- **CatASB** deploys OpenShift + Service Catalog + Brokers
  - Developer tool built from Ansible Playbooks
  - Uses `oc cluster up` to start OpenShift

- Deploy an OpenShift cluster + the AWS Broker in a few commands

```bash
$ git clone https://github.com/fusor/catasb.git
$ echo "deploy_awsservicebroker: True" >> catasb/config/my_vars.yml
$ cd catasb/local/linux
$ ./run_local_setup.sh
```

- [https://github.com/fusor/catasb](https://github.com/fusor/catasb)
Installing with an AWS QuickStart

- AWS offers an OpenShift on AWS quickstart
  - Creates an OpenShift cluster on AWS
  - Has a parameter to enable the AWS Broker on the created cluster
- Free AWS credits available for evaluation purposes

https://aws.amazon.com/quickstart/architecture/openshift/
AWS Service Broker - Roadmap
AWS Broker - Roadmap

- Asynchronous Bind Support
- Advanced Parameter Validations
- More AWS Services

- We want your feedback!
Let's build a hybrid cloud app!
Purpose

- Pull share prices from New York Stock Exchange
- Use affinity propagation to cluster related stocks
- Render graphic grouping related stocks
## Architecture

- **Web UI (flask)**
  - Web UI accepts new jobs
  - Add jobs to **AWS SQS** queue
- **Worker (scikit learn + python)**
  - Pick up jobs from **AWS SQS** queue
  - Build graph showing stock clusters
  - Store results in **AWS S3**
  - Send results notification w/ **AWS SNS**
1. AWS Broker creates services
2. AWS bind credentials are attached.
3. App uses bind credentials to communicate with AWS services.
Demo - Machine Learning on the NYSE with AWS
Where to go from here
Resources

- **AWS Broker**
  - Docs - [https://bit.ly/2jlm0zO](https://bit.ly/2jlm0zO)
  - AWS QuickStart (free credits!) - [https://amzn.to/2x6m1ph](https://amzn.to/2x6m1ph)

- **Automation Broker (base project)**
  - Freenode IRC: #asbroker
Your feedback is appreciated!