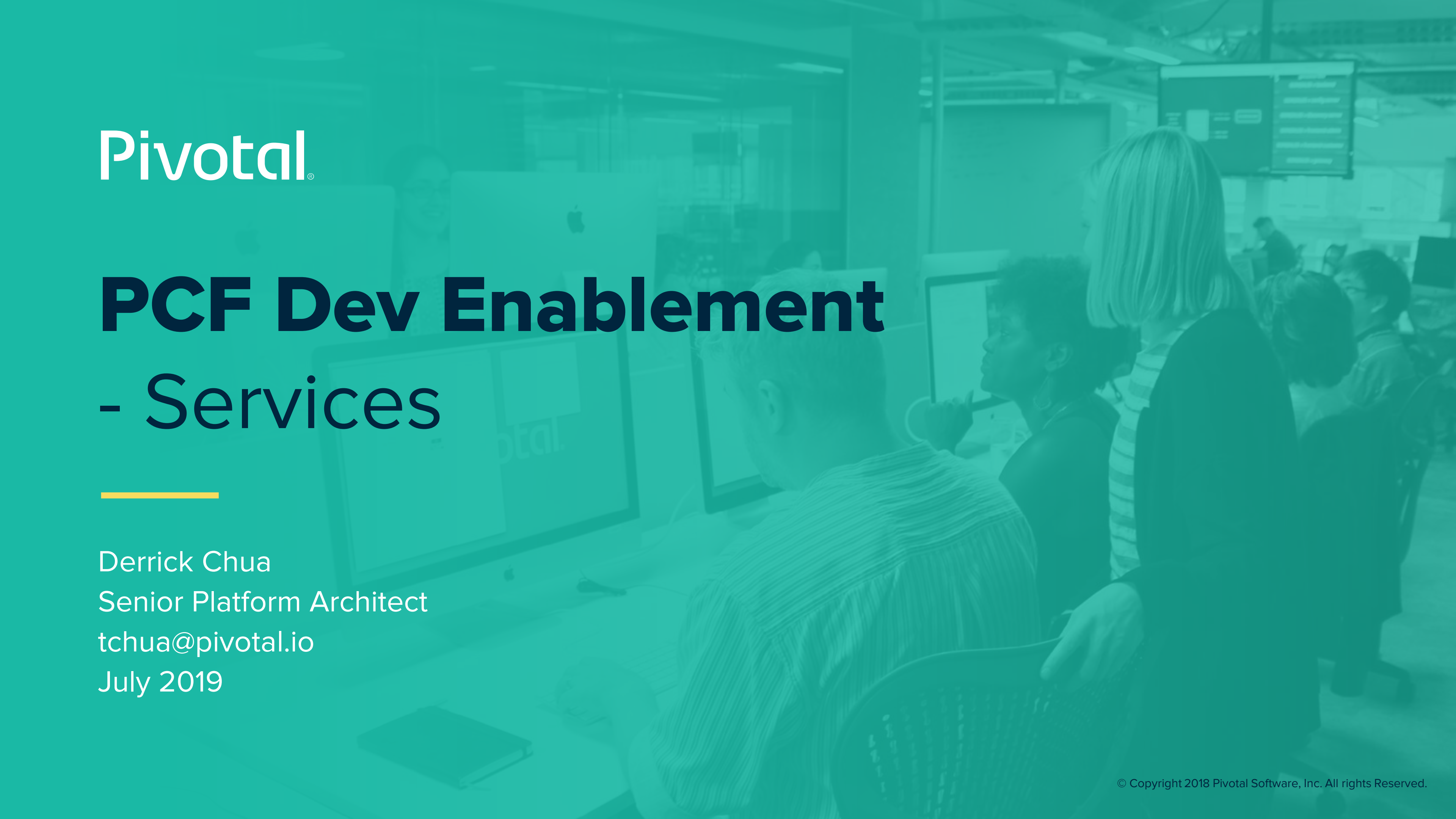


Pivotal

# PCF Dev Enablement - Services

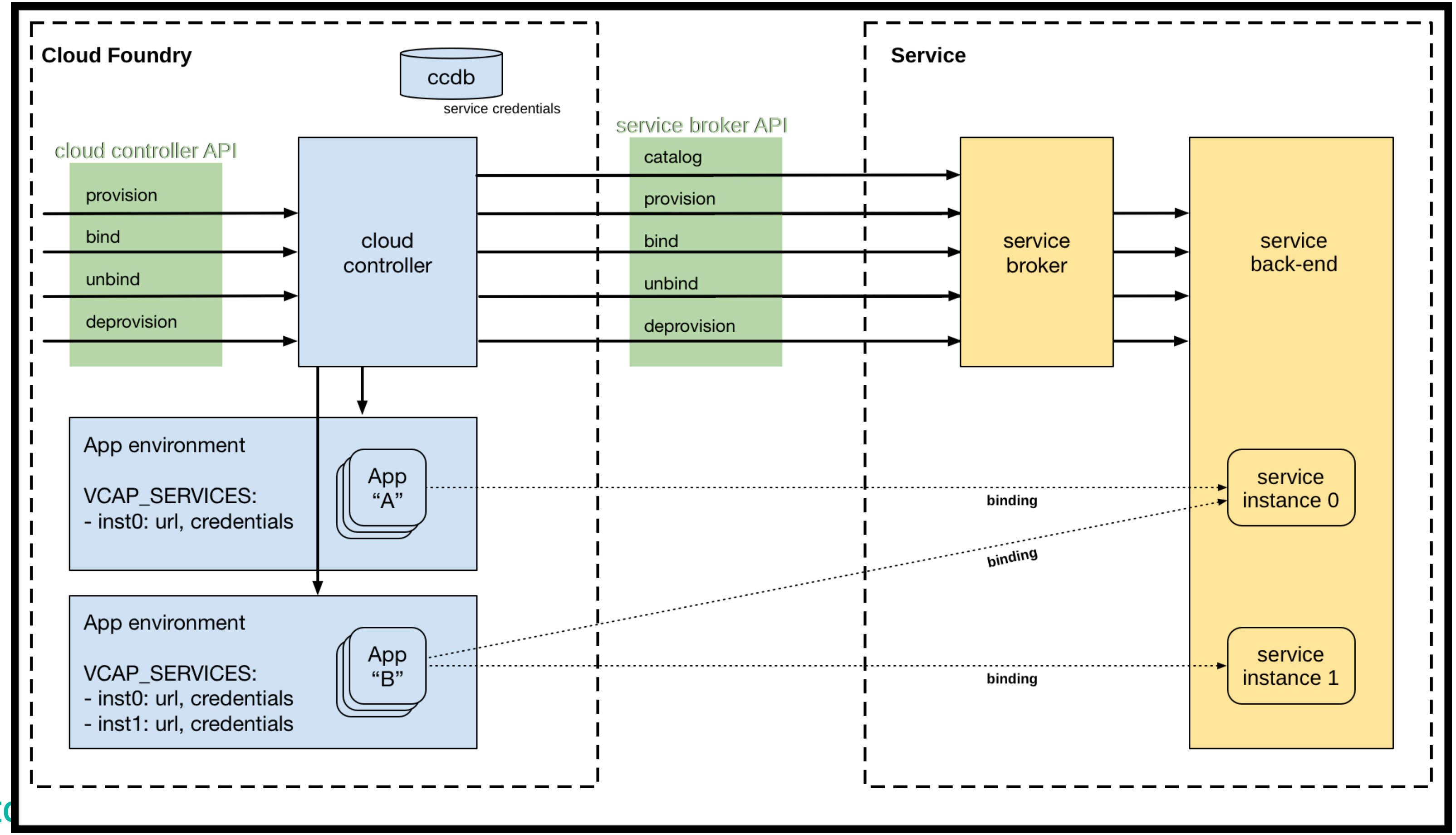
Derrick Chua
Senior Platform Architect
tchua@pivotal.io

July 2019

## Services Overview

Pivotal

# Services Architecture

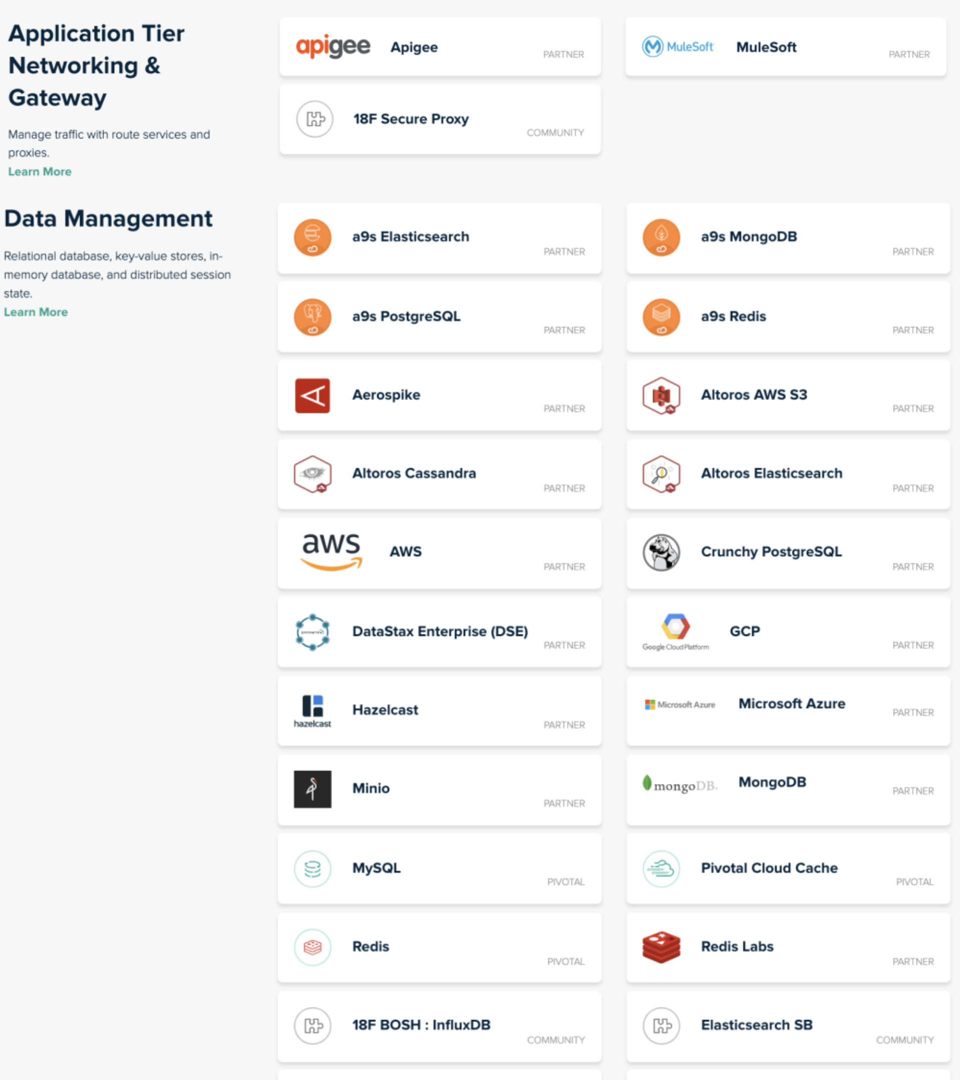


## Services types

Marketplace Services

User Provided Service Instances (UPSI)

Custom Services



## Application Tier Networking & Gateway

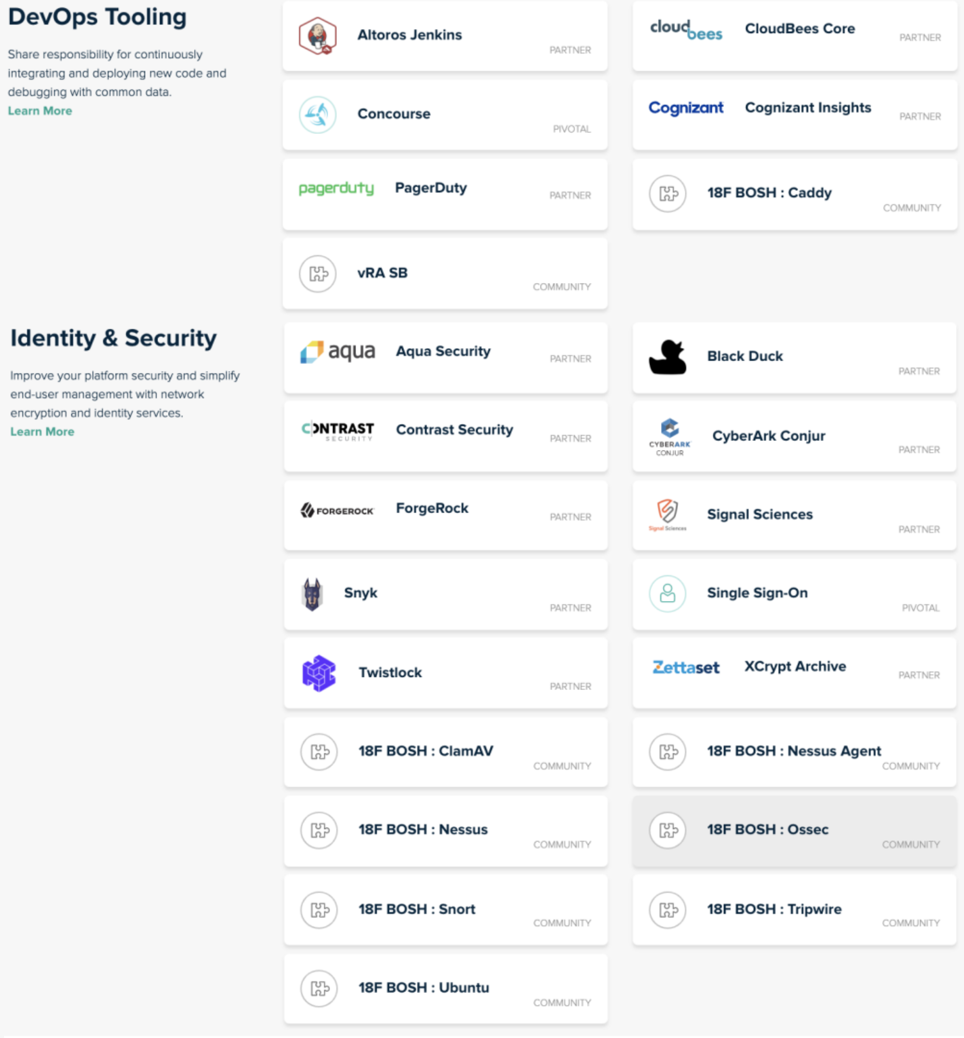
Manage traffic with route services and proxies.
Learn More

| | |
|---|---|
| Apigee | PARTNER |
| MuleSoft | PARTNER |
| 18F Secure Proxy | COMMUNITY |

## Data Management

Relational database, key-value stores, in-memory database, and distributed session state.
Learn More

| | |
|---|---|
| a9s Elasticsearch | PARTNER |
| a9s MongoDB | PARTNER |
| a9s PostgreSQL | PARTNER |
| a9s Redis | PARTNER |
| Aerospike | PARTNER |
| Altoros AWS S3 | PARTNER |
| Altoros Cassandra | PARTNER |
| Altoros Elasticsearch | PARTNER |
| AWS | PARTNER |
| Crunchy PostgreSQL | PARTNER |
| DataStax Enterprise (DSE) | PARTNER |
| GCP | PARTNER |
| Hazelcast | PARTNER |
| Microsoft Azure | PARTNER |
| Minio | PARTNER |
| MongoDB | PARTNER |
| MySQL | PIVOTAL |
| Pivotal Cloud Cache | PIVOTAL |
| Redis | PIVOTAL |
| Redis Labs | PARTNER |
| 18F BOSH : InfluxDB | COMMUNITY |
| Elasticsearch SB | COMMUNITY |



## DevOps Tooling

Share responsibility for continuously integrating and deploying new code and debugging with common data.
Learn More

| | |
|---|---|
| Altoros Jenkins | PARTNER |
| CloudBees Core | PARTNER |
| Concourse | PIVOTAL |
| Cognizant Insights | PARTNER |
| PagerDuty | PARTNER |
| 18F BOSH : Caddy | COMMUNITY |
| vRA SB | COMMUNITY |

## Identity & Security

Improve your platform security and simplify end-user management with network encryption and identity services.
Learn More

| | |
|---|---|
| Aqua Security | PARTNER |
| Black Duck | PARTNER |
| Contrast Security | PARTNER |
| CyberArk Conjur | PARTNER |
| ForgeRock | PARTNER |
| Signal Sciences | PARTNER |
| Snyk | PARTNER |
| Single Sign-On | PIVOTAL |
| Twistlock | PARTNER |
| XCrypt Archive | PARTNER |
| 18F BOSH : ClamAV | COMMUNITY |
| 18F BOSH : Nessus Agent | COMMUNITY |
| 18F BOSH : Nessus | COMMUNITY |
| 18F BOSH : Ossec | COMMUNITY |
| 18F BOSH : Snort | COMMUNITY |
| 18F BOSH : Tripwire | COMMUNITY |
| 18F BOSH : Ubuntu | COMMUNITY |

More at https://pivotal.io/platform/services-marketplace

## User Provided Service Instances (USPI)

- Deliver service credentials to app

```
cf cups SERVICE_INSTANCE -p '{"username":"admin","password":"pa55woRD"}'
```

- Stream app logs to a service

```
cf cups SERVICE_INSTANCE -l syslog://example.log-aggregator.com
```

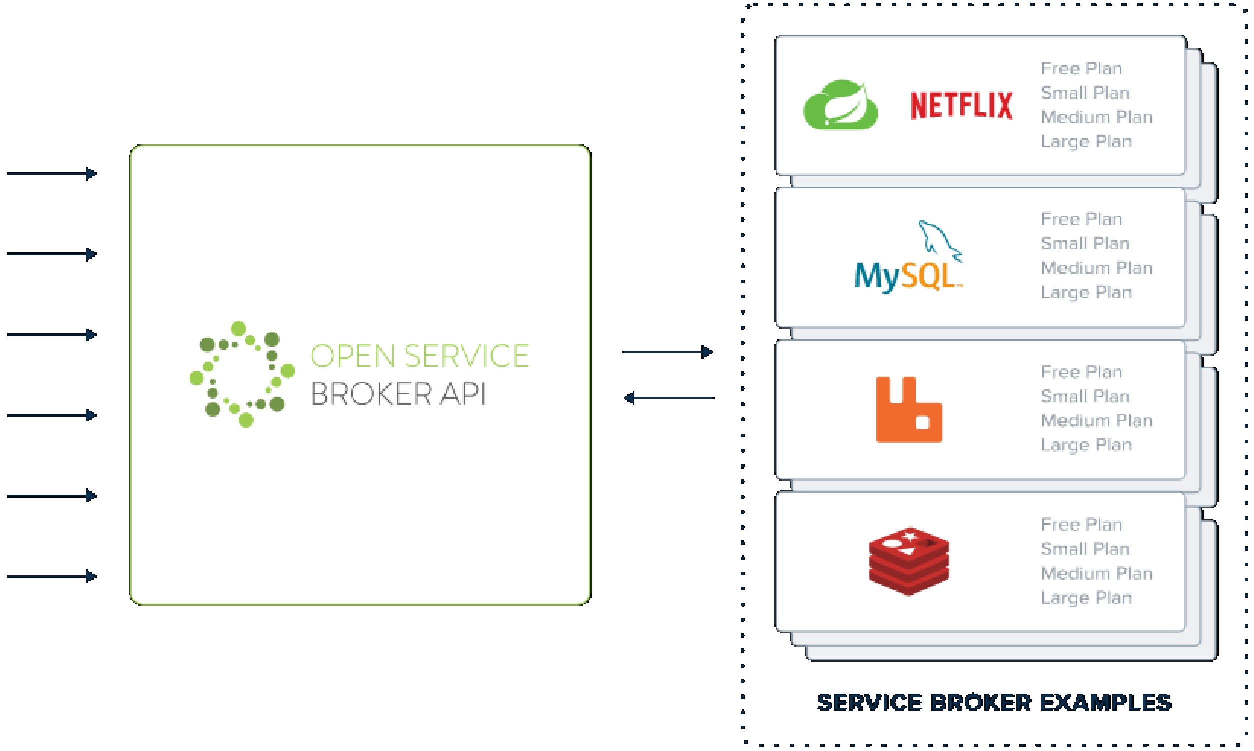
- Proxy app requests to a route service

```
$ cf create-user-provided-service my-user-provided-route-service -r
https://my-route-service.example.com
```

# Custom Services

## Open Service Broker API + Bring your own service

1. Create a service broker
2. Add it your CF marketplace, or K8s service catalog
3. Create a service instance; choose your plan size
4. Bind the service to your app or K8s pod
5. Unbind the service
6. Delete the service when it's no longer needed



## Custom Services

### Implementing Open Service Broker API

Fetching the catalogue of backing services that a service broker offers

Provisioning new service instances

Connecting and disconnecting applications and containers from those service instances

Deprovisioning service instances

Pivotal

Full spec at https://github.com/openservicebrokerapi/servicebroker/blob/v2.13/spec.md

# Managing (Marketplace) Service Instances

Pivotal

## Hands-On: Step 1

- Find out through the command line the list of available services for your PAS account, using the following command
    - cf marketplace

- Find out through the Apps Manager UI the list of available services for your PAS account

## Hands-On: Step 2

- Create a new service instance of MySQL database for your app, using the following command
    - cf create-service [service-name] [plan-name] emp-db-svc

## Hands-On: Step 3

- There are multiple ways to bind the database service instance to the employees-api microservice – through the command line, the Apps Manager UI and as part of the manifest. We will be using the manifest method here.
- Add the following mysql dependency to pom.xml, to the point just above the h2 dependency

```
<dependency>
      <groupId>mysql</groupId>
      <artifactId>mysql-connector-java</artifactId>
</dependency>
```

- At the root directory of the application, run the following commands
  - Clean the target directory – mvnw.cmd clean
  - Rebuild the target jar – mvnw.cmd –DskipTests package
- Push the jar to PAS using another manifest named manifest-service.yml, using the following command
  - cf push -f manifest-service.yml

## Hands-On: Step 4

- Once cf push has completed successfully, query the API GET methods by accessing the /employees endpoint
- Using the Apps Manager UI, visit the app page and expand the health check section under each app instance. Verify that mysql is now being used as the backing database

## Think about...

- Are there any services in the marketplace that you might find useful for your current or next application?
- What are the other cf commands related to services?
- Does every app binding deliver service credentials?
- How can you use UPSI to deliver credentials for your current RDBMS to apps in PAS?
- How can you get service keys through the Apps Manager UI?

## Service Instance Credentials

Pivotal

## Credentials are delivered via app binding
### using VCAP_SERVICES environment variable

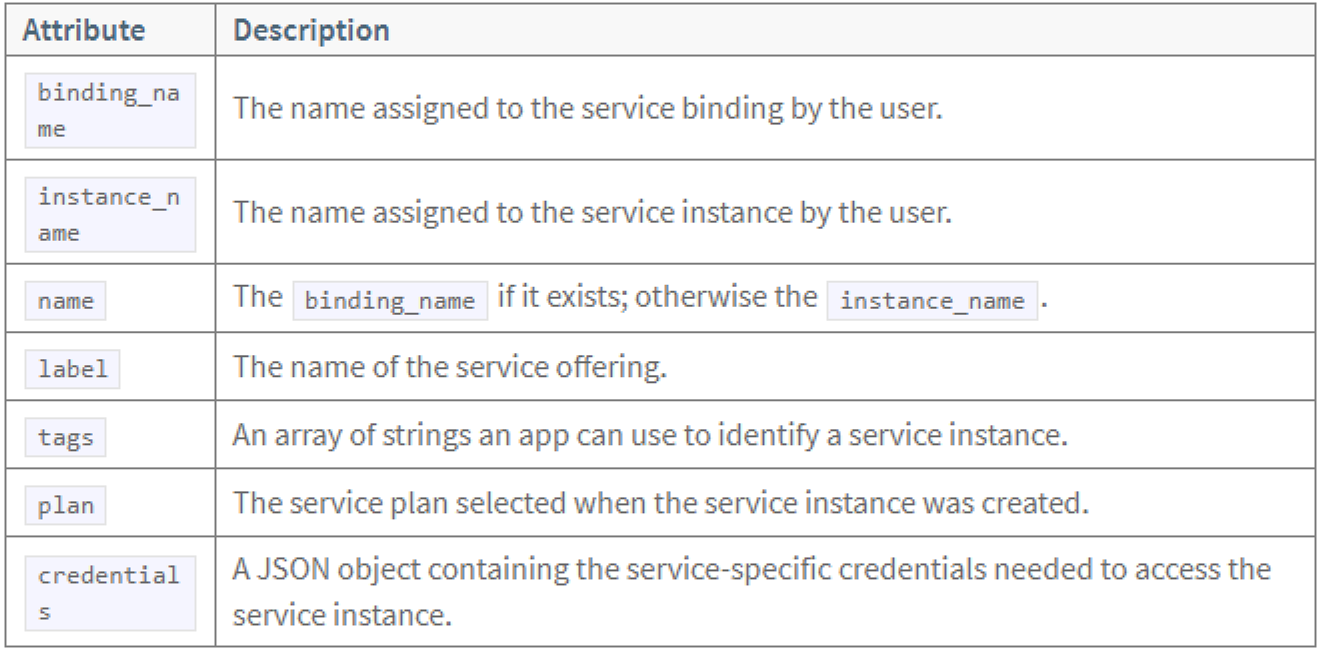
```
VCAP_SERVICES=
{
  "elephantsql": [
    {
      "name": "elephantsql-binding-c6c60",
      "binding_name": "elephantsql-binding-c6c60",
      "instance_name": "elephantsql-c6c60",
      "label": "elephantsql",
      "tags": [
        "postgres",
        "postgresql",
        "relational"
      ],
      "plan": "turtle",
      "credentials": {
        "uri": "postgres://exampleuser:examplepass@babar.elephantsql.com:5432/exampleuser"
      }
    }
  ],
  "sendgrid": [
    {
      "name": "mysendgrid",
      "binding_name": null,
      "instance_name": "mysendgrid",
      "label": "sendgrid",
      "tags": [
        "smtp"
      ],
      "plan": "free",
      "credentials": {
        "hostname": "smtp.sendgrid.net",
        "username": "QvsXMbJ3rK",
        "password": "HCHMOYluTv"
      }
    }
  ]
}
```

Manual parsing is required

Pivotal

# Credentials are delivered via app binding

## using VCAP_SERVICES environment variable



| Attribute | Description |
|---|---|
| `binding_name` | The name assigned to the service binding by the user. |
| `instance_name` | The name assigned to the service instance by the user. |
| `name` | The `binding_name` if it exists; otherwise the `instance_name` . |
| `label` | The name of the service offering. |
| `tags` | An array of strings an app can use to identify a service instance. |
| `plan` | The service plan selected when the service instance was created. |
| `credentials` | A JSON object containing the service-specific credentials needed to access the service instance. |

## Credentials are delivered via app binding

### using auto-configuration (Spring example)

- **Criteria**
  - Only one service instance of a given service type is bound to the application
  - Only one bean of a matching type is in the Spring application context. For example, you can have only one bean of type javax.sql.DataSource
- Cloud Foundry automatically creates DataSource object and binds it to database service using internal credentials

No manual parsing, configuration nor coding required

Details at https://docs.pivotal.io/pivotalcf/2-6/buildpacks/java/configuring-service-connections/spring-service-bindings.html

## Service Keys

- Credentials managed manually
  - Used by apps in a different space
  - Used by apps not within Cloud Foundry

```
$ cf create-service-key MY-SERVICE MY-KEY
Creating service key MY-KEY for service instance MY-SERVICE as me@example.com...
OK
```

```
$ cf service-key MY-SERVICE MY-KEY
Getting key MY-KEY for service instance MY-SERVICE as me@example.com...

{
  uri: foo://user2:pass2@example.com/mydb,
  servicename: mydb
}
```

Pivotal

## Sharing Service Instances

## Use case

- Allow apps in different spaces to share databases
- Without using service keys

## Criteria

- Service instances can be shared into multiple spaces and across orgs.
- Developers and administrators can share service instances between spaces in which they have the Space Developer role.
- Developers who have a service instance shared with them can only bind and unbind apps to that service instance. They cannot update, rename, or delete it.
- Developers who have a service instance shared with them can view the values of any configuration parameters that were used to provision or update the service instance.

## How to share

- Enabling Service Instance Sharing in Cloud Foundry

```
$ cf enable-feature-flag service_instance_sharing
```

- Sharing a Service Instance

```
$ cf share-service SERVICE-INSTANCE -s OTHER-SPACE [-o OTHER-ORG]
```

*To share a service instance into a space, the space must have access to the service and service plan of the service instance that you are sharing.
Run the `cf enable-service-access` command to set this access.

- Unsharing a Service Instance

```
$ cf unshare-service SERVICE-INSTANCE -s OTHER-SPACE [-o OTHER-ORG] [-f]
```

## Security Considerations

- Service keys cannot be created from a space that a service instance has been shared into.
- This ensures that developers in the space where a service instance has been shared from have visibility into where and how many times the service instance is used.
- Sharing service instances does not automatically update app security groups (ASGs). The network policies defined in your ASGs may need to be updated to ensure that apps using shared service instances can access the underlying service.
- Access to a service must be enabled using the cf enable-service-access command for a service instance to be shared into a space.
- Not all services are enabled for sharing instances functionality. Contact the service vendor directly if you are unable to share instances of their service. If you are a service author, see Enabling Service Instance Sharing.

Pivotal®

## Transforming How The World Builds Software