**PCF (Pivotal Cloud Foundry):**

**spaces** - every application and service is scoped to a space. Each org contains at least one space. A space provides users with access to a shared location for application development, deployment, and maintenance. Each space role applies only to a particular space.

**organizations** - an org is a development account that an individual or multiple collaborators can own and use. All collaborators access an org with user accounts. Collaborators in an org share a resource quota plan, applications, services availability, and custom domains.

**routes** - define how to get to an application

* a unique route exists to each application in every space
* domain can be mapped to multiple spaces
* route can only be mapped to one space
* same application can be deployed in multiple spaces
* each must have a different, unique URL
* development space route: [http://myapp-test.cfapps.io](http://myapp-test.cfapps.io/)
* production space route: [http://myapp.cfapps.io](http://myapp.cfapps.io/)

**services** - any type of add-on that can be provisioned along side your apps

* database, messaging, mail, third-party SaaS provider
* services are usually bound to 1 or more applications
* connection info and credentials are put in an environment variable: VCAP\_SERVICES

**domains** - define routes to apps

**users** - a user account represents an individual person within the context of a PCF application. A user can have different roles in different spaces within an org, governing what level and type of access they have within that space.

**quotas** - restrict resources for orgs and spaces

* total memory
* total number of routes
* max application instance size
* total number of services
* How do you login to Cloud Foundry?

cf login -a api.run.pivotal.io

Enter username :

Enter password :

* After entering credentials you would be logged in to your Org and space.
* How do you deploy an application? What are three activities involved?

cf push APP\_NAME

example : cf push atendee-service -p D:/Users/shupurwa/Downloads/pcf-attendee-service-code-master/target/attendee-1.0.jar -m 650M --random-route

**attendee-service** : APP\_NAME

**-p** denotes path of JAR/WAR/EAR

**-m** denotes memory allocation to you application

**--random-route** denotes that application route will be taken automatically by PCF

1. Upload
2. Stage
3. Start

* What infrastructure does Cloud Foundry runs on?
  + AWS
  + VMware
  + Openstack
  + Azure
* What is BOSH? Why is it useful?

BOSH can provision and deploy software over hundreds of VMs. It also performs monitoring, failure recovery, and software updates with zero-to-minimal downtime.

* What is staging? What does it do?

1. Uploads application packages to staging droplet.
2. Applies Buildpack
3. Builds the application and make it ready to be run.

* Do you know the difference between restarting, restaging and redeploying and application? How does each of these affect the services, environment-variables available on an application?

**restart** will simply stop your application and start it with the existing droplet.

**restage** will stop your application, run the application bits through the staging process to create a new droplet, and then start the new droplet. It's a lot like push but without actually pushing new application bits.

You typically restart when you need your applicaiton's environment refreshed and you typically restage when you need/want the buildpack to run without updating the application source.

* What is meant by ephemeral? What are the design implications for an application?

ephemeral -> temporary virtual machines and containers are temporary immutable infrastructure - updates to systems and applications are not done in-place new, updated instances are created instead

* What are the 12 Factor Design patterns? Could you list each one from memory?

1. **Processes** - Execute the app as one or more stateless processes
2. **Concurrency** - Scale out via the process model
3. **Disposability** - Maximize robustness with fast startup and graceful shutdown
4. **Logs** - Treat logs as event streams
5. **Build, release, run** - Strictly separate build and run stages
6. **Processes** - Execute the app as one or more stateless processes
7. **Port binding** - Export services via port binding
8. **Concurrency** - Scale out via the process model
9. **Disposability** - Maximize robustness with fast startup and graceful shutdown
10. **Dev/prod parity** - Keep development, staging, and production as similar as possible
11. **Logs** - Treat logs as event streams
12. **Admin processes** - Run admin/management tasks as one-off processes

* Why does Cloud Foundry rely on environment-variables?

Environment variables are the means by which the Cloud Foundry runtime communicates with a deployed application about its environment.

* Can you manage environment-variables manually? If so how?

Yes, environment variables can be managed manually!

* + cf set-env my-app my-variable\_name my-variable\_value
  + cf unset-env my-app my-variable\_name my-variable\_value
* Can you name two predefined environment-variables available to any application?
  + VCAP\_APPLICATION
  + VCAP\_SERVICES

Commands Used in CLI:

-- View organizations?

cf orgs

-- View routes?

cf routes

-- View spaces?

cf spaces

-- View a single organization, route or space?

cf org ORG cf route ROUTE cf space SPACE

-- Deploy an application?

cf push

-- Select a space and/or organization to deploy to?

cf target [-o ORG] [-s SPACE]

-- View logs?

cf logs APP\_NAME

-- Connect (login) to the Cloud Controller?

cf login [-a API\_URL] [-u USERNAME] [-p PASSWORD] [-o ORG] [-s SPACE]

-- Start, stop or restart an application?

cf start APP\_NAME cf stop APP\_NAME cf restart APP\_NAME

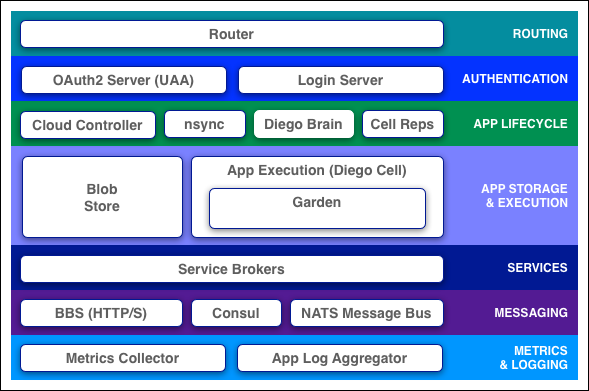
-- What does cf target do? What information does it give you?

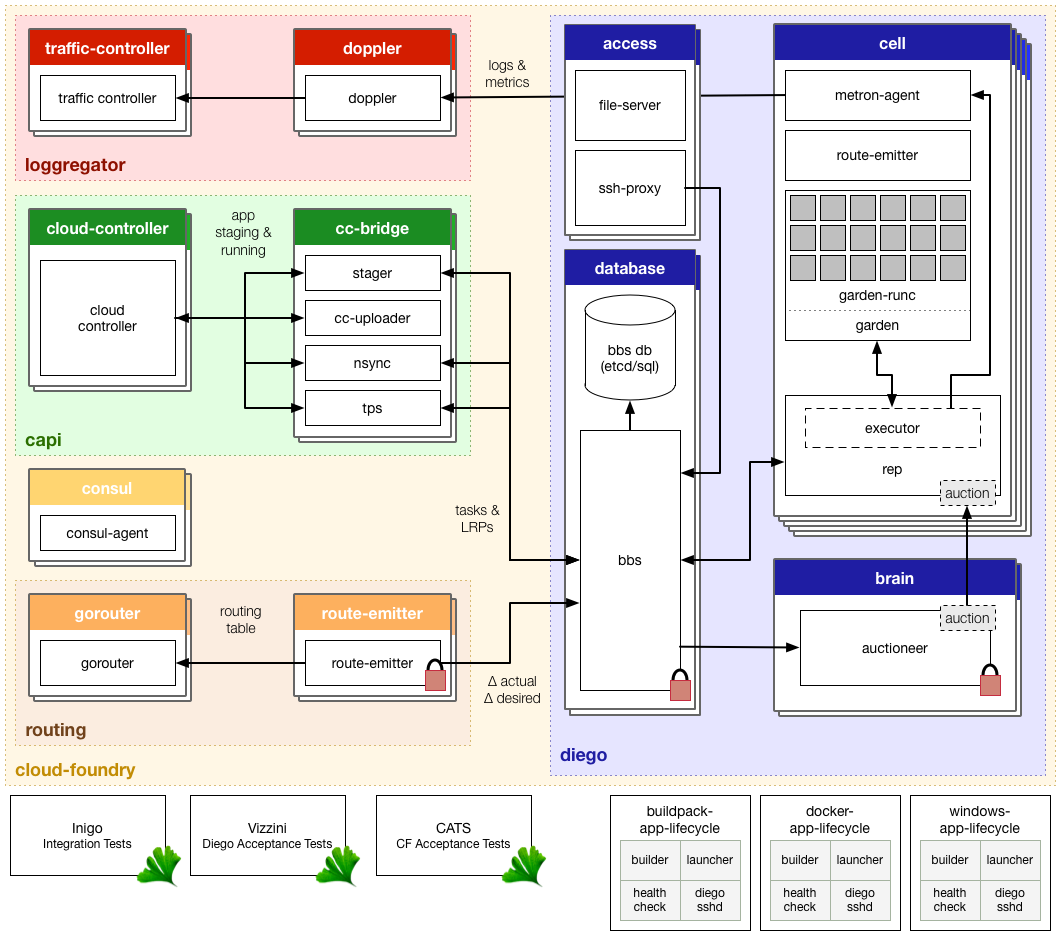
it set or view the targeted org or space

* API endpoint: https://api.run.pivotal.io (API version: 2.6.0)
* User: xyz@gmail.com
* Org: pivotaledu
* Space: development

**Cloud Foundry Architecture**

* Can you name the main components running inside Cloud Foundry? Do you know what each of them does?





**Elastic Runtime Subsystems**

* Diego
* Loggregator
* Cloud Controller API
* Routing

**Key Sequence Flows through the ER**

**Diego**

* Schedules tasks and Long-Running Processes (LRPs)

**Task**

* Is guaranteed to run at most once
  + E.g. stage an application

**LRP**

* Typically represented as a web app. LRPs can have multiple instances

**Container**

* An application instance is run within an immutable container

**Cell**

* Containers are run within a cell

**Garden**

* Containers are managed by Garden
* Garden is an interface
* Garden-Linux is a backend implementation

**Rep**

* Represents the Cell in the BBS/Auctions

**Auction**

* An auction is held to bid on executing a task or an LRP

**Executor**

* Manages container allocations on the cell
* Subprocess of a Rep
* Steams stdout and stderr to metron

**Metron**

* Forwards logs to the Loggregator subsystem

**BBS**

* Bulletin board system
* API to access the Diego database for tasks and LRPs
* Data is persisted to etcd

**Brain**

* Composed of two components
  + Auctioneer
    - Hold auctions for tasks and LRPs
  + Converger
    - Reconciles desired LRPs vs Actual LRPs through auctions

**Loggregator**

**Doppler**

* Gathers logs from Metron
* Create app syslog drains Splunk Papertrail

**Traffic Controller**

* Handles client requests for logs
* Exposes a web socket endpoint called the firehose

**Firehose**

* A websocket endpoint that exposes app logs, container metrics and ER component metrics

**Nozzles**

* Consume the firehouse output
  + E.g. Datadog Nozzle

**Cloud Controller API**

**Cloud Controller**

* Exposes an API for using and managing the Elastic Runtime

**Cloud Controller Database (CC\_DB)**

* The Cloud Controller persists Org/Space/App data in the Cloud Controller Database

**Blob Store**

* The Cloud Controller persists app packages and droplets to the Blob Store

**CC-Bridge**

* CC-Bridge translates app specific messages into the generic language of task and LRPs

**Routing**

**Router**

* Routes traffic to appropriate component

What does Diego refer to?

PCF container management system

* What is Garden?

it's a component that CF uses to create and manage isolated environments called containers.

* What components run in a Diego Cell?
* Rep
* Executor
* Garden
* Metron Agent
* Route-emitter
* What is the System domain? And the application domain?
  + The system domain is the domain used for things like the API, Apps Manager and other system provided services.
  + The apps domain is the default domain that will be setup for developers to use when pushing applications to the environment.
* What is Cloud Foundry's API endpoint for?
* it provides access to the system
* for managing spaces and orgs
* What is a container? What is it used for?

An application instance is run within an immutable container

* What is a droplet? How is it created? Where is it stored?
* Droplets is a staged apps packaged with everything needed to run in a container.
* It is created during application staging process by Diego cell
* It is stored in Blobstore
* What are the purposes of the two data stores used by the Cloud Controller?
* Blobstore
  + Store app packages, buildpacks, droplets
* Database (CC\_DB)
  + The Cloud Controller persists Org/Space/App data in the Cloud Controller Database

## Logging, Scaling, and High Availability

* How do you access application logs?

cf logs APP\_NAME

* What are the components of the Loggregator system?

1. Metron - Forward logs to the Loggregator subsytem
2. Doppler - gather logs from Metron
3. Traffic Controller - handles client requests for logs
4. Firehose - a websocket endpoint that exposes app logs, container metrics and ER component metrics
5. Nozzles - consume the firehose output

* How do you scale an application manually?

cf scale <APP\_NAME> -m -i <Instance\_COUNT>

* What are the four levels of high-availability provided by PCF?

1. BOSH detects and recreates failed VMs
2. BOSH also detects and restarts failed processes
3. Application instances are automatically spread -- Across different Cells -- Across Cells in different Availability Zones
4. CF Converger detects if an application instance has failed and restarts it

* What is the difference between scaling up and scaling out?
* scale up (vertically) increases the disk space limit or memory limit of all instances of an application. It requires downtime as the container is recreated.
* scale out (horizontally) increases the number of instances of an application by creating or destroying them. It requires no downtime.
* What are some fundamental changes impacting application design and delivery?
* Distributed computing
* Ephemeral infrastructure
* Immutable infrastructure
* What is a modern methodology for delivering cloud based applications?

Twelve-factor app

* How are logs handled in a Twelve-factor app?

As event streams

* True or false, Garden is an interface with many implementations.

True, there's Linux and Windows and more on the way

* Which component of the Elastic runtime exposes an API for users to manage applications?

Cloud controller

* What are the four levels of high availability that keep your applications running?
* Availability zones
* BOSH managed processes
* Failed VMs - any VM that BOSH deploys will be recreated if you enable resurrector
* Self healing application instances handled by the brain within Diego
* Where should you write your application logs?
* To Standard Out and Standard Error
* What are some of the different origin codes that were seen in the log?
* cf logs
* API for cloud controller
* stage for stating
* cell for running applications
* What is the difference between scaling out versus scaling up?

Scaling up requires down time to recreate the container Scaling horizontally requires no downtime

* How could you tell if your application has been crashing?

Look at the logs, but probably the best place to go is cf events

**Managed and User-Provided Services**

* What is a service? Can you name some examples?

service is a resource reserved on demand examples: database on a shared or dedicated server, access to a SaaS application

* What is the "marketplace"? Does it show all services?

A marketplace is a listing of all service brokers and their corresponding catalogs which can be used by the apps deployed into cloud foundry by creating respective service instances.

All market place services can be seen using: cf marketplace

All services ( Managed & User-Provided) inside a particular space can be seen using: cf services

* What is the difference between a managed and user-provided service?

Managed service is a service that is available in marketplace. User-provided service instances enable developers to use services that are not available in the marketplace with their applications running on Cloud Foundry.

* How would you create a managed service?

cf create-service SERVICE PLAN SERVICE\_INSTANCE

* How would you use a managed service?

cf bind-service APP\_NAME SERVICE\_INSTANCE [-c PARAMETERS\_AS\_JSON]

* How would you create a user-provided service?

cf cups SERVICE\_INSTANCE [-p CREDENTIALS] [-l SYSLOG\_DRAIN\_URL] [-r ROUTE\_SERVICE\_URL]

* How would you use a user-provided service?

cf bind-service APP\_NAME SERVICE\_INSTANCE [-c PARAMETERS\_AS\_JSON]

* What is VCAPS\_SERVICES? Why is it important?

it is an environment variable containing connection details of bindable services that Cloud Foundry added after binding the services instance to the application.

* How do services interact with spaces?

Every application and service is scoped to a space. Use service keys when you want a set of credentials for use by clients other than the application in the same space. Not all services support service keys. Some services support credentials through application binding only.

* What is the difference between a managed service and a user-provided service instance?

A managed service is advertised in the marketplace

User provided service are provisioned outside of the platform

* Would it be advisable for a 12 factor app to store a configuration in a local file?

No, because the application would have to be rebuild to change that configuration.

* Does a 12 factor app make a distinction between local and third party services?

No, it doesn't. Both should provide location and credential information through the config, through the operating system environment variables

* After binding a service to an application, why is the application restarted, or restaged?

So the application can have access to the new environment variables. The container is immutable, so once you bind a service to an application, we'll need to throw that one away by either restarting or restaging, or create a new one.

* Why could we restart versus restage?

Restage rebuilt the application droplet.

* Whe does it need to be rebuilt?

When there's new dependencies to pull in

**Manifests**

* What is a manifest? What type of file is it? What is its purpose?
  + describes the application deployment options
  + YAML file
  + automates subsequent deployments
  + same options as cf push command
* How do you create a manifest?
  + using text editor, or
  + cf create-app-manifest APP\_NAME
* If I specify a command in a manifest and on the command line what happens?

the cf push CLI options always take precedence over manifest file

* What happens if I don't specify a deployment option at all?

cf push will default all deployment options

**Application Security Groups**

* What is an application security group (ASG)? What does it do?

An application security group is a virtual firewall that control outbound traffic for applications. It will allow/disallow what the app can access.

* How do you define one?

cf create-security-group <SECURITY\_GROUP\_NAME> <ASG.json> ASG.json is the security group rules file.

* What can an ASG apply to?

An ASG(Application Security group) can be applied to all the applications and ASG will be applied to running applications when they are restarted.

* What is the difference between white and black listing? Which do you use when defining an ASG?

White listing - limiting the application to limit the exposure of specific data.

Black listing - limiting the application to limit the application access to internet etc.,

White listing is used when defining ASG.

* As a developer what cf commands can be used to investigate and understand what's going on with application security groups?
* cf security-groups
* cf security-group APP\_NAME
* Do application security groups use a white list or black list approach to firewall rules?

White list

* What is the relationship between running security groups and space security groups?

It's a union of both of those put together. So if you've allowed any outbound connection from either running security group or space security group your application won't be able to take advantage of that.

* What are some reasons why security groups could be used?
* Protection against typically trusted resources like employees and preventing them from doing bad things
* Limit your exposure and only allowing access to what application should have access to
* Force applications to really use the applications to use the appropriate APIs. In a microservie based architecture you really want to force all of the data access through the appropriate application rather than side stepping that application and accessing the data directly

**Log Drains**

* What is a log drain?

Inorder to persist more than limited amount of logging information that cloud foundry can buffer, logs can be collected into an external Log Management Service.

* How do you create one?

External Log Management Services are readily available in the marketplace. $ cf create-service SERVICE PLAN SERVICE-INSTANCE & $ cf bind-service YOUR-APP YOUR-LOG-STORE

* Why would you use it?

Logs are important to analyze the health & performance of an application. Constant monitoring is required on the application when it is in production. Inorder to address this, we should persist the logs.

* What is "Syslog"?

de facto standard for logging on Unix/Linux

* Is cf restart or restage required for log draining to start working? Why or why not?

It's not required, because from application perspective it's just writing to standard out or standard error. Doppler is a component of the Loggregator system that's responsible for forwarding those logs to third party log management services like Paper Trail or Splunk.

**Blue-Green Deployments**

* What is a blue-green deployment? Why would you use it?

it's a technique that reduces downtime and risk by running two identical production environments called Blue and Green.

This technique can eliminate downtime due to application deployment. Blue-green deployment also reduces risk; if something unexpected happens with your new version on Green, you can immediately roll back to the last version by switching back to Blue.

* How do you map and unmap route with cf?

cf map-route APP\_NAME DOMAIN [--hostname HOSTNAME] [--path PATH]

cf unmap-route APP\_NAME DOMAIN [--hostname HOSTNAME] [--path PATH]

* How does route mapping enable a blue-green deployment?

by routing the existing domain to the Green

* What is the purpose of a Blue-Green deployment?

To achieve no downtime when managing multiple versions of your application

* How is the distribution of traffic handled when running multiple versions of an application?

It's based on the percentage of app instances that is running, and traffic will be distributed accordingly

* True of False. When releasing a new version of an application, adding a new non-nullable field to a database table is an acceptable Blue-Green strategy?

False. If it's a new non-nullable field without a default value then the old version of the application will run into problems when your application deploys.

* How would a rollback situation be handled using a Blue-Green deployment?

Unmap the production route from the next version of the application.

* What other design implications does running at least two versions of the same time have on your application?

Changes need to be backwards compatible and non-destructive

## Application Autoscaler

* What is the autoscaler?
* App Autoscaler is a marketplace service that ensures app performance and helps control the cost of running apps.
* It allows applications to be automatically scaled
* System load can be used as a trigger in place of manual interaction
* How would you use it?

service

* What can you configure?

configure the minimum no.of instances and max no. of instances

* Do you understand autoscaler schedules?

When there is an expected increase/decrease of traffic for the application during specific times of a day or on a particular day, it can be configured with Scheduled Limit Changes. For example: increasing the number of instances for Black Friday, decreasing the number of instances during off-business hours etc.,

<https://docs.pivotal.io/pivotalcf/1-12/appsman-services/autoscaler/using-autoscaler.html>

**Application Performance Monitor**

* What is performance monitoring? How is it implemented for Cloud Foundry applications?
* real-time monitoring of applications
* marketplace service
* Why is a buildpack involved?

to include the APM agent in the droplet

* What APM tools are available with Cloud Foundry?

New Relic, AppDynamics, Dynatrace

* Why must Articulate and Attendee service be restaged as opposed to restarted?

Because the droplet has to include the New Relic agent and to do that, our application droplet needs to be restaged

**Metrics**

* What is "PCF Metrics"? What information does it provide?

PCF built-in metrics information

* Container metrics: CPU, memory, disk %
* HTTP metrics: requests per second, HTTP errors per second, request latency
* App events: create, update, start, stop, and crash
* Do you understand how to use the metrics to spot a badly performing application? For example: CPU or memory usage?

1. Set up a custom chart to monitor the application performance (or)
2. Highlight the CPU Utilization spike keyword in the logs and debug the application to resolve what is causing the spike.

<http://docs.pivotal.io/pcf-metrics/1-3/using.html>

**Buildpacks**

* What is a buildpack? Why are they important?

a combination of scripts that assembles runtimes, containers, frameworks, and your application into a droplet.

droplets run inside PCF containers, which run inside Execution Agents (Cell VMs).

build packs are responsible for preparing the machine image for an application

* Can you name some buildpacks?

Official:

* Binary
* Go
* Java
* .Net Core
* Node.js
* PHP
* Python
* Ruby
* Staticfile
* How does Cloud Foundry know which buildpacks to run?

During staging, each buildpack has a position in a priority list (cf buildpacks). CF checks if the buildpack in position 1 is a compatible buildpack. If not, CF moves on to the next buildpacks until the correct buildpack is found. If no buildpack is compatible, cf push fails.

* How does a buildpack work? Are you aware of the scripts that run and how they might be written or modified?

scripts:

* bin/detect inspect application to see if the buildpack should be applied
* bin/compile creates droplet by combining application with runtime, container, packages, libraries (downloading them if necessary, kept in cache thereafter)
* bin/release build application start command
* Can you tell what buildpack was used when an application was deployed?

cf app APP\_NAME gives the details of the buildpack used for the application.

* Why might you customize a buildpack? In general, would you know what to do?

If your application uses a language or framework that the Cloud Foundry system

* So which script is responsible for building the droplet?

The answer is the compile script.

* How do buildpacks help with managing CVEs or security issues?

Instead of patching systems manually, simply upload the patch buildpacks and then restage your applications. It's much, much easier to manage that way

* Would a cf restart be sufficient instead of a cf restage? Why not?

it really wouldn't be sufficient, because the droplet needs to be rebuilt. If you recall, we set the configuration parameter to set the Java version. And when you do that and issue the cf restage command, that environment variable is available to the staging process, because it's being run in a container. All the same environment variables are available there. And it's going to go ahead and use that particular version of the JRE.

* What other items are easily customized with the Java buildpack?

We talked about one, the Tomcat version. Also, the JRE implementation. So, for instance, you could use Oracle. if you wanted to get into managing the trustStore, you could investigate the documentation on that as well.

**Route Services**

* What is a route service?

Route Services are a kind of Marketplace Service that developers can use to apply various transformations to application requests by binding an application’s route to a service instance.

* Why might you create a route service? Can you think of some examples?

Provide transformation or processing to requests before/after they reach an application Examples:

* + Authentication
  + Rate limiting
  + Logging
  + Caching
  + Transformation
* How would you create a route service?

cf create-user-provided-service SERVICE-INSTANCE -r ROUTE-SERVICE-URL

cf bind-route-service [domain-name] [route-service-name] --hostname [target-hostname]

* How does a route service work?

Binding a service instance to a route associates the route\_service\_url with the route in the Cloud Foundry router. All requests for the route are proxied to the URL specified by route\_service\_url

* How does a route service know where to send a request once it has performed its function?

With the Headers

* + X-CF-Forwarded-URL
  + X-CF-Proxy-Signature
  + X-CF-Proxy-Metadata

Used by the Router to validate the request and pass through to the application or reject it.