



Pivotal PCF Basics

Yadhav Jayaraman – Sr. Platform Architect
yjayaraman@pivotal.io

The Pivotal value proposition

			
Developer Productivity <ul style="list-style-type: none">• Accelerate feedback loops by improving delivery velocity• Focus on applications, not infrastructure• Give developers the tools and frameworks to build resilient apps	Operational Efficiency <ul style="list-style-type: none">• Employ 500:1 developer to operator ratio• Perform zero-downtime upgrades• Runs the same way on every public/private cloud	Comprehensive Security <ul style="list-style-type: none">• Adopt a defense-in-depth approach• Continuously update platforms to limit threat impact• Apply the 3 R's + repair, replace, rotate	High Availability <ul style="list-style-type: none">• Run platforms that stay online under all circumstances• Scale up and down, in and out, through automation• Deploy multi-cloud resilience patterns

Speed

Deploy new code
thousands of times a month

- **Synchrony Financial:** 30% faster deployment
- **Verizon:** 86.8% faster time to market, 92.6% faster environment provisioning
- **Scotiabank:** 3k deployments per month
- **Allstate:** 11.5k production deployments per year
- **Liberty Mutual:** 1k production deployments per day

Domain Driven Dev @DomainDriven · Oct 5
10 months on #pivotal at #Scotiabank: 3k+ releases per month and 11 of 29
systems running #continuousDeployment
#SpringOne #bring



1 13 2 4 5



Danielle Burner
@dburner11
Make the right thing the easy thing. This is where having an opinionated platform is awesome. [@mattjcurry](#) of [@Allstate](#) on how [@pivotal](#) is a key ingredient for enabling successful #DevOps at scale #SpringOne



Speed

Deploy new code
thousands of times a month =
Faster MVPs and time to market

- **CoreLogic:** New data feature in 2 months
- **Boeing:** MVPs down from 1-3 years to 3-5 months
- **HCSC:** Reduced release cycles from 14 months to 14 weeks to 14 days
- **Mercedes Benz:** Reduced time from inception to production from ~30 days to 2-3 days



Stability & Scalability

Delivering enterprise SLAs and breakthrough operational efficiency



- **Discover** has over 32 applications on PCF, 250 developers
- **Northern Trust** has 750 microservices running on 5 PCF Foundations. 2 Platform operators support 250 Spring Boot Developers
- **T-Mobile** has 11k app Instances supported b 8 Platform operators
- **Comcast** now running 19k App Instances across 12 Foundations
- **Verizon** now at 1800 services, running across 16 Foundations
- **Allstate** supports 1027 developers

Security

Improve your security posture
with built-in capabilities

- **Express Scripts:** From 45 days to 5 days to upgrade 9 PCF Foundations, for 1 patch for 1 product
- **Verizon:** From patching every 6 months to weekly. 81.5% faster patching and scaling
- **Synchrony Financial:** 80% faster patching execution

The screenshot shows a LinkedIn post by Bob Gilther (@BobGilther - Dec 5). The post discusses impressive results from Verizon's transition from monolith to cloud-native. It includes a screenshot of a dashboard titled "Results" comparing "Test Time" and "Productivity".

Category	Initial State	Final State	% Improvement
Test Time (Time to market)	100%	10%	90%
Productivity (Deployment)	100%	100%	0%
By 2 Dev (Planning & testing)	100%	100%	0%

The dashboard also shows "Release Time (Productivity)" with the same data points. The post concludes with a note about "Measuring Success" and "Automated Updates".

Benefits And Costs



Operations productivity improvement:
\$5.8M



Developer productivity improvement:
\$31.2M



PCF platform cost:
\$16.4M

Benefits (Three-Year)



**ROI
135%**



**Benefits PV
\$38.4 million**



**NPV
\$22 million**

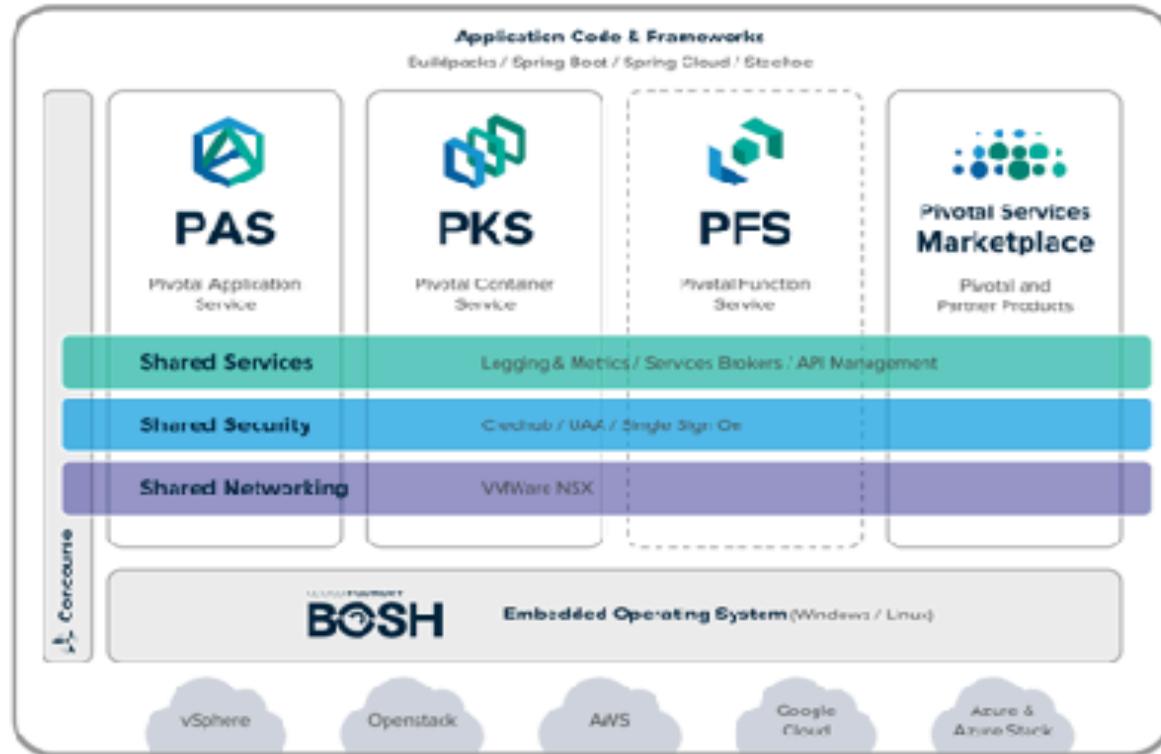
A Forrester Total Economic Impact™
Study Commissioned By Pivotal
October 2017

What is Cloud Foundry

Any App Every Cloud One Platform

PCF 2.0 — for everything that matters

Pivotal



Value
Line

Applications



BYO container, deployed and managed by platform

Elastic Container Runtime

Container Scheduling

Elastic (Auto) Scaling

Dev/Ops Self Service

Logging Metrics

Self Healing HA

Zero Downtime Patch/Upgrade

Managed Services

Database

Cache

Messaging

SSO

SCM/CI

NOSQL

Brokered Services

API Mgmt

Monitoring

NOSQL

SSO

Database

Messaging

Infrastructure Automation (BOSH)



Pivotal Application Service (PAS): A Runtime for Apps

Increase speed and deploy code to production thousands of times per month. Use PAS to run Java, .NET, and Node apps.



Pivotal
Application Service

Best runtime for Spring and Spring Boot — Spring's microservice patterns—and Spring Boot's executable jars—are ready-made for PAS.

Turnkey microservices operations and security — Spring Cloud Services brings microservices best practices to PAS. It includes Config Server, Service Registry, and Circuit Breaker Dashboard.

A native Windows and .NET experience — Use PAS to run new apps built with .NET Core. Run your legacy .NET Framework apps on PAS too, using the .NET Hosted Web Core buildpack. Push applications to containers running on Windows Server 2016.

Built for apps — PAS has everything to need to run apps. Buildpacks manage runtime dependencies, metrics, logging, and scaling are done for you. Multitenancy, and blue/green deployment patterns are built-in. Extend apps with a rich service catalog.

Container-ready — PAS supports the OCI format for Docker images. Run platform-built and developer-built containers.

Enterprise Ready Services



MySQL for PCF

- Enterprise-ready MySQL for your developers
- Automate database operations in developer workflows
- **NEW:** Leader-follower for multi-site HA



Pivotal Cloud Cache

- High performance, in-memory, data at scale for microservices
- Look-aside caches & HTTP session state caching
- **NEW:** WAN replication



RabbitMQ for PCF

- Easily connect distributed applications with the most widely deployed open source message broker
- Enable connected scalable, distributed applications
- **NEW:** On-demand clusters

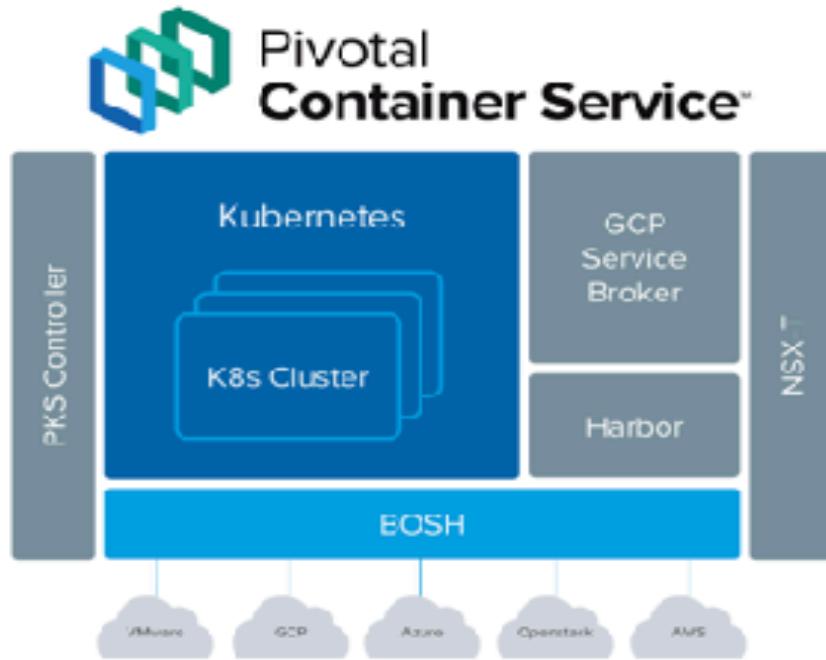


Redis for PCF

- In-Memory cache and datastore, configured for the enterprise
- Efficient provisioning matched to use cases

[BOSH Managed](#) | [On-Demand Provisioning](#) | [Dedicated Instances](#) | [Custom Service Plans](#)

Pivotal Container Service (PKS): A Runtime for Containers



Built with open-source Kubernetes — Constant compatibility with the current stable release of Kubernetes, operated by BOSH. No proprietary extensions.

Production-ready — Highly available from apps to infrastructure, no single points of failure. Built-in health checks, scaling, auto-healing and rolling upgrades.

Multicloud — BOSH provides a reliable and consistent operational experience. For any cloud.

Network management and security out-of-the-box with VMware NSX-T. Multi-cloud, multi-hypervisor.

GCP APIs access — The GCP Service Broker allows apps to transparently access Google Cloud APIs, from anywhere. Easily move workloads to/from Google Container Engine (GKE).

Fully automated Ops — Fully automated deploy, scale, patch, upgrade. No downtime. Use CI/CD pipelines to deploy your platform, too.

Pivotal Function Service (PFS): A Runtime for Functions

Execute functions in response to events. Use PFS to handle web events, event-based integration, and large scale streaming data.



**Pivotal
Function Service™**

Trigger functions via HTTP/Message Broker — PFS is architected to support event stream processing, connecting to message topics via a language-neutral, function container interface.

Run functions anywhere — PFS lets you easily run functions on-premises and in the public cloud for maximum flexibility.

Use modern DevOps workflows — PFS allows you to use familiar, container-based workflows for serverless scenarios.

Pluggable event brokers — PFS can be connected easily with popular message brokers such as Kafka, RabbitMQ, Google Pub/Sub, and AWS Kinesis.

Polycast — PFS supports the authoring of functions in your chosen framework - Node.js, Spring/Java, or Shell.

Kubernetes Native — PFS runs natively on top of Kubernetes, making it easy to trigger code or containers in response to events.

Using the right abstraction is key for results



Container
Orchestrator
(CaaS)

Application
Platform
(PaaS)

Serverless
Functions
(FaaS)

IaaS

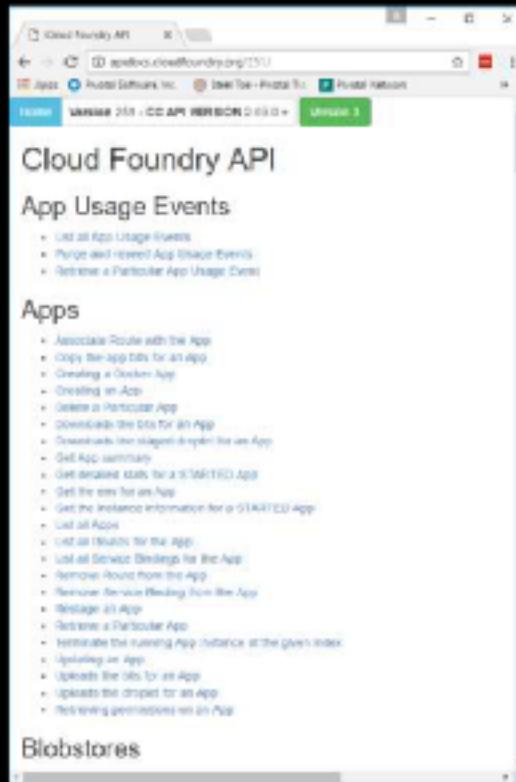
Pivotal

Demo

Interacting with PCF

Cloud Controller API

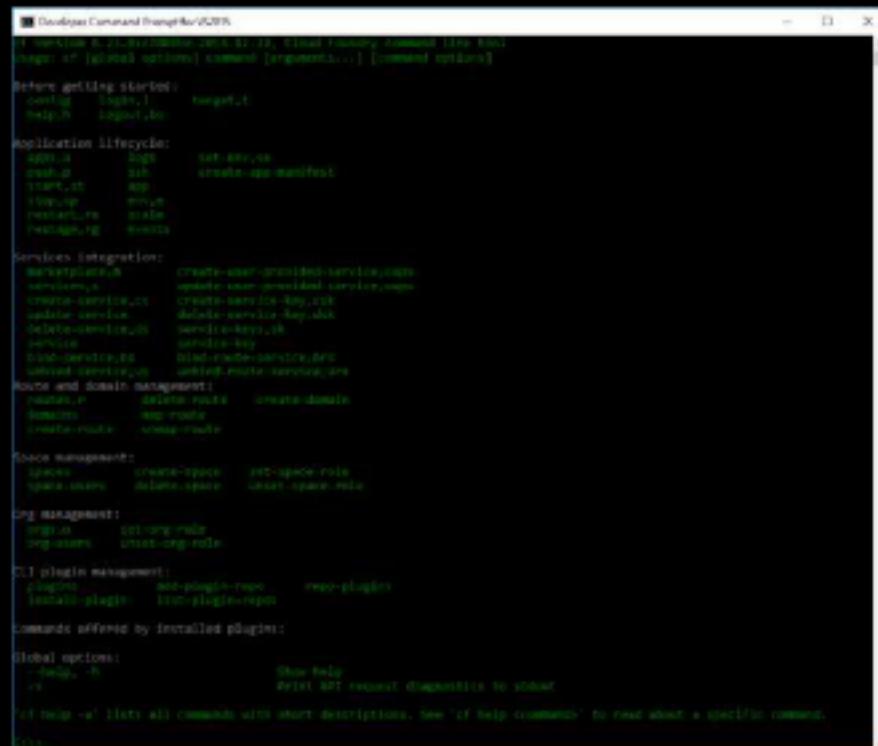
- Cloud Controller (CC) component of Elastic Runtime manages all Cloud Foundry APIs
- CF CLI and other clients like Apps Manager directly call this API
- Before accessing the CC API, you must get an access token from the User Account and Authentication (UAA) server
- <http://apidocs.cloudfoundry.org>



CLI – Command Line Interface

- Command line utility providing easy access to the Cloud Controller API.
- Scriptable
- Fully documented

```
cf help -a  
cf help <command>  
cf api http://foo.bar.com/  
cf login <username>
```

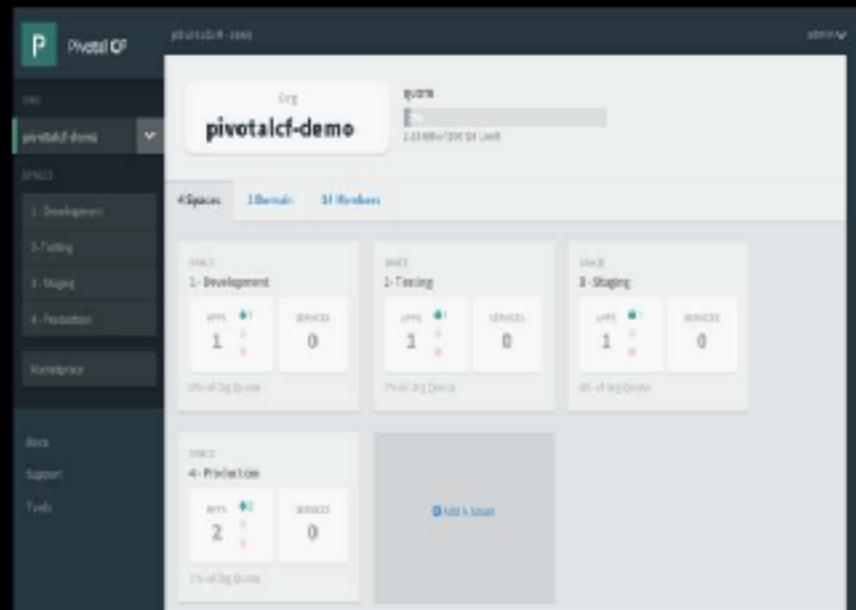


A screenshot of a Windows Command Prompt window titled "Developer Command Prompt for VS2010". The window displays the output of the "cf help" command. The output is a large block of text organized into sections: "Before getting started", "Application lifecycle", "Services integration", "Route and domain management", "Space management", "Org management", "CF plugin management", "Commands affected by installed plugins", "Global options", and "See also". Each section lists commands and their descriptions. At the bottom, there is a note about viewing detailed descriptions for specific commands.

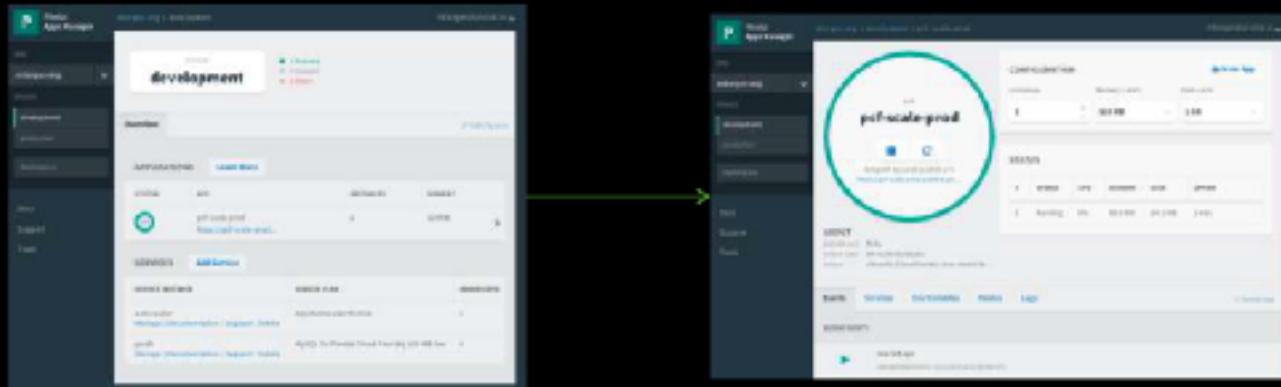
```
Developer Command Prompt for VS2010  
Administrator: C:\Windows\system32\cmd.exe /k cf help  
Usage: cf [global options] <command> [<arguments>] [<command options>]  
  
Before getting started:  
  config     config target  
  help       help logon  
  
Application lifecycle:  
  app        logs      get-env  
  build     push      create-app-manifest  
  start     stop      delete  
  start-stc  app      start  
  stop      metrics   metrics  
  resources  scale    scaling  
  package   events   events  
  
Services integration:  
  service-create  create-user-provided-service  
  service-list   update-user-provided-service  
  service-control  create-service-key  
  update-service  delete-service-key  
  delete-service  delete-service  
  service-key    service-key  
  bind-service   bind-service  
  unbind-service  unbind-service  
  
Route and domain management:  
  routes     delete-route  create-domain  
  domains   map-route  
  create-route  reverse-route  
  
Space management:  
  spaces    create-space  get-space-role  
  space-users  delete-space  update-space-role  
  
Org management:  
  orgs     get-org-role  
  organizations  update-org-role  
  
CF plugin management:  
  plugin   add-plugin-repo  rm-plugin  
  install-plugin  rm-plugin-repo  
  
Commands affected by installed plugins:  
  
Global options:  
  --help     -h          Show help  
  -v         --version  
  
See also:  
  cf help -a  Lists all commands with short descriptions. See 'cf help command' to view about a specific command.  
  cf help
```

Pivotal Apps Manager

- Manage Organizations, users, applications and Spaces
- Monitor applications logs, services and routes
- Access Service Marketplace, create services and bind to applications

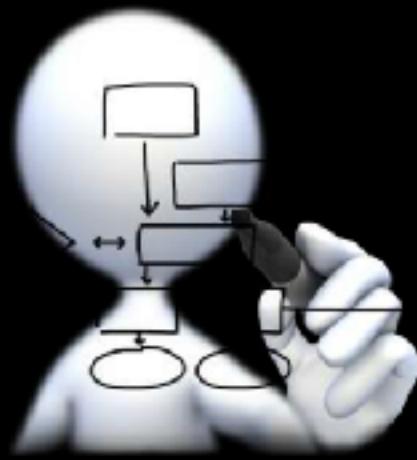
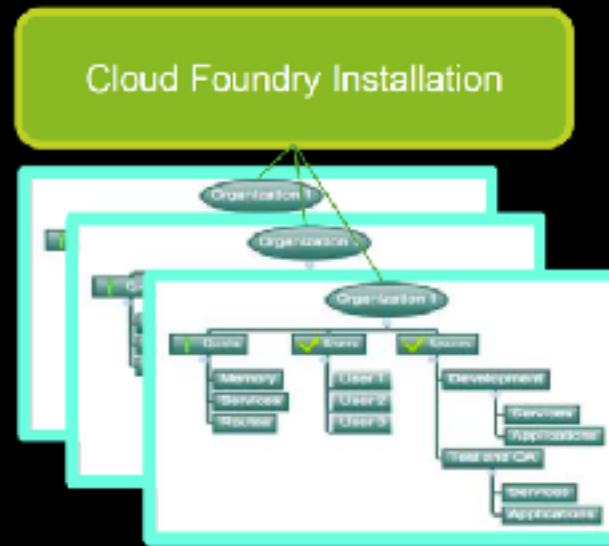


Pivotal Apps Manager – App View



- Drill into a space to see all application and services instances
- Then drill into an application to see configuration, status, event, logging, routes, environment variables and service instances bound to the application

Orgs, Spaces, Users and Quotas



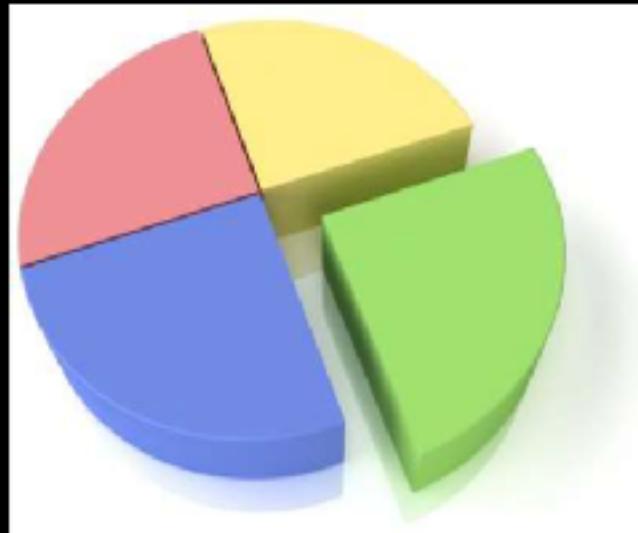
Organizations

- Top-most administrative unit
- Logical division within a Pivotal Cloud Foundry Install / Foundation
 - Typically a company, department, application suite or large project
- Each organization has its own users and assigned quota
- User permissions / Roles are specified per space within an organization
- Sub-divided into spaces



Quotas

- Different quota limits (e.g. small, enterprise, default, runaway) can be assigned per Organization
- Quotas define
 - Total Memory
 - Total # of services
 - Total # of Routes
- Sub-divided into spaces



Spaces

- Logical sub-division within an organization
- Users authorized at an organization level can have different roles per space
- Services and Applications are created / target per Space
- Same service name can have different meaning per space



Users and Roles

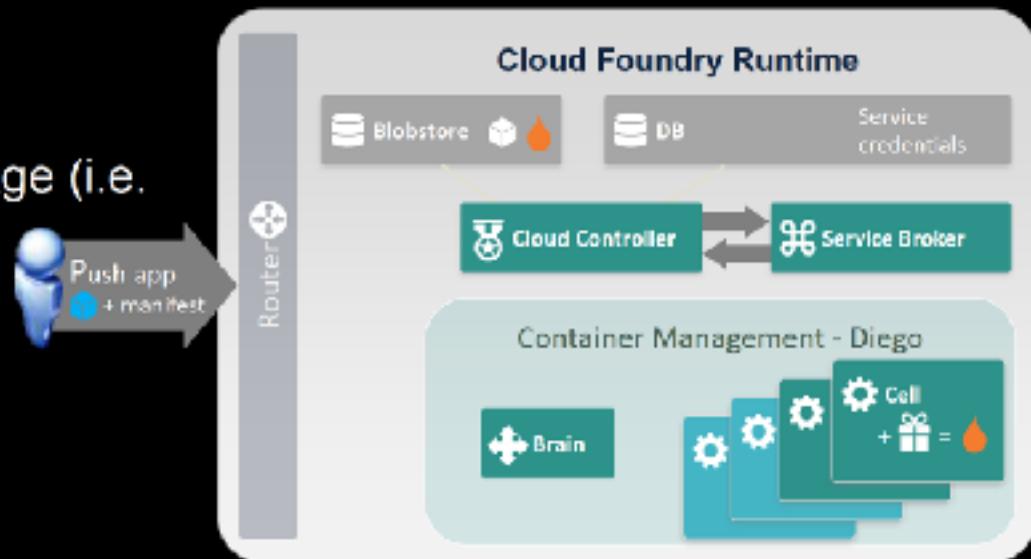
- Users are members of an organization
 - Usually they are operators or developers (not application end users)
 - Users are sent an email invite and asked to create an account
- Users have specific organization and space roles
 - Organization roles grant permissions in an organization
 - Space roles grant permissions in a particular space
 - A combination defines the user's overall permissions



Running Apps

Pushing an Application

1. Upload app bits and metadata
2. Bind services
3. Stage application
4. Save staged application image (i.e. droplet 🔥)
5. Deploy image to container
6. Manage applications health



```
cf push appname -p <path to bits>
cf push appname -f <manifest> -p <path to bits>
```

Manifest Files

- Application manifests tell cf push what to do with applications
- What OS stack to run on: Windows or Linux
- How many instances to create and how much memory to use.
- Helps automate deployment, specially of multiple apps at once
- Can list services to be bound to the application
- YAML format – <http://yaml.org>

```
1  ->
2  # all applications use these settings and services
3  domain: shared-domain.com
4  memory: 1G
5  instances: 1
6  services:
7  - clockwork-mysql
8  applications:
9  - name: springtck
10 host: tock09876
11 path: ./spring-music/build/libs/spring-music.war
12 - name: springtick
13 host: tick09875
14 path: ./spring-music/build/libs/spring-music.war
```

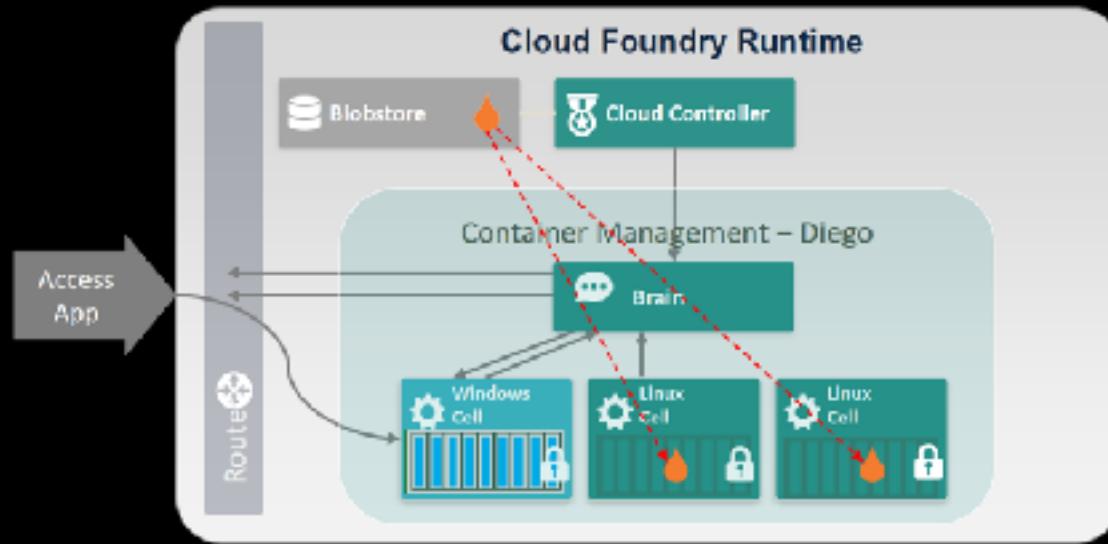
Staging an Application – Applying Buildpacks

- Buildpacks build container images
- Buildpacks take care of
 - Detecting which type of application is being pushed
 - Installing the appropriate run-time
 - Installing required dependencies or other artifacts
 - Creating the command used to start the application
- Lots of Buildpacks
 - Staticfile
 - Java
 - Ruby
 - Nodejs
 - Go
 - Python
 - PHP
 - .NET Core
 - Binary

Why Buildpacks

- Control what frameworks/runtimes are used on the platform
- Provides consistent deployments across environments
 - Stops deployments from piling up at operation's doorstep
 - Enables a self-service platform
- Eases ongoing operations burdens:
 - Security vulnerability is identified
 - Subsequently fixed with a new buildpack release
 - Restage applications

Deploying Image to Container



CF Push

Behind the scenes with

\$ cf push



Upload

'cf push' uploads the application bits and metadata to the Cloud Controller

Stage

Behind the scenes the code goes through a set of staging scripts called **BuildPacks** which create a container blueprint called '**Droplet**'

Distribute

'**Diego**' which is an auction based workload scheduler spins up containers and with the '**Droplet**' image, and alerts the '**Cloud Controller**' that the app is ready to receive traffic

Route

Your app receives an entry in the routing tier which provides a dynamic loadbalancer

Eliminating Service Tickets

```
D:\pcf\PCF-demo-1>cf push
Using manifest file D:\pcf\PCF-demo-1\manifest.yml
Updating app pcfdemo-1 in org PCF-Org-01 / space development as H141869...
OK
Using route pcfdemo.apps.pcf.dce
Uploading app files from: D:\pcf\PCF-demo-1\target\pcfdemo.war
Uploading 615.4K, 66 files
Done uploading
OK
Binding service myrabbit to app pcfdemo-1 in org PCF-Org-01 / space development as H141869...
OK
Binding service mylogger to app pcfdemo-1 in org PCF-Org-01 / space development as H141869...
OK
Binding service myscale to app pcfdemo-1 in org PCF-Org-01 / space development as H141869...
OK
Stopping app pcfdemo-1 in org PCF-Org-01 / space development as H141869...
OK
Starting app pcfdemo-1 in org PCF-Org-01 / space development as H141869...
----> Downloaded app buildpack cache (4.0K)
----> Java Buildpack Version: v3.0 (offline) | https://github.com/cloudfoundry/java-buildpack.git
java-buildpack
----> Downloading OpenJDK JRE 1.8.0_40 from https://download.run.pivotal.io/openjdk/trusty/x86_64/openjdk-1.8.0_40.tar.gz (found in cache)
      Expanding OpenJDK JRE to java-buildpack/openjdk/jre (0.9s)
----> Downloading Spring Auto Reconfiguration 1.7.0.RELEASE from https://download.run.pivotal.io/auto-reconfiguration/auto-reconfiguration-1.7.0.RELEASE.jar (found in cache)
      Modifying WEB-INF/web.xml for Auto Reconfiguration
----> Downloading tomcat instance 8.0.21 from https://download.run.pivotal.io/tomcat/tomcat-8.0.21.tar.gz (found in cache)
      Expanding tomcat instance to java-buildpack/tomcat/tomcat-8.0.21 (0.9s)
----> Downloading Tomcat Lifecycle Support 2.4.0.RELEASE from https://download.run.pivotal.io/tomcat-lifecycle-support/tomcat-lifecycle-support-2.4.0.RELEASE.jar (found in cache)
----> Downloading Tomcat Logging Support 2.4.0.RELEASE from https://download.run.pivotal.io/tomcat-logging-support/tomcat-logging-support-2.4.0.RELEASE.jar (found in cache)
----> Downloading Tomcat Access Logging Support 2.4.0.RELEASE from https://download.run.pivotal.io/tomcat-access-logging/tomcat-access-logging-2.4.0.RELEASE.jar (found in cache)
----> Uploading droplet (60M)
1 of 1 instances running
App started
OK

App pcfdemo-1 was started using this command "JAVA_HOME=$PWD/.java-buildpack/open_jdk_jre JAVA_OPTS=-Djava.io.tmpdir=$TMPDIR -XX:MemoryError=SPAWN -java-buildpack/open_jdk_jre/bin/killjava.sh -Xms382293K -Xmx382293K -XX:MaxMetaspaceSize=64M -XX:MetaspaceSize=64M -Xss999K -Djava.security.egd=file:///dev/urandom -Daccess.logging.enabled=false -Dhttp.port=8080" - $java-buildpack/tomcat/tomcat-8.0.21/bin/catalina.sh run"
Showing health and status for app pcfdemo-1 in org PCF-Org-01 / space development as H141869...
OK
Showing health and status for app pcfdemo-1 in org PCF-Org-01 / space development as H141869...
OK
requested state: started
instances: 1
usage: 512M x 1 instances
uri: pcfdemo.apps.pcf.dce.honeywell.com
last uploaded: Mon Aug 3 08:19:12 UTC 2015
stack: clinuxfs2
       state     since          CPU    memory   disk      details
v0  running  2015-08-02 07:19:38 PM  0.0x  345.8M of 512M  138.3M of 1G
D:\pcf\PCF-demo-1>
```

Firewall & Routes

Service Bindings

App Lifecycle

Runtime Installation & Config

Middleware Installation & Config

Application Installation & Config

App Lifecycle

Logging, Health, Telemetry

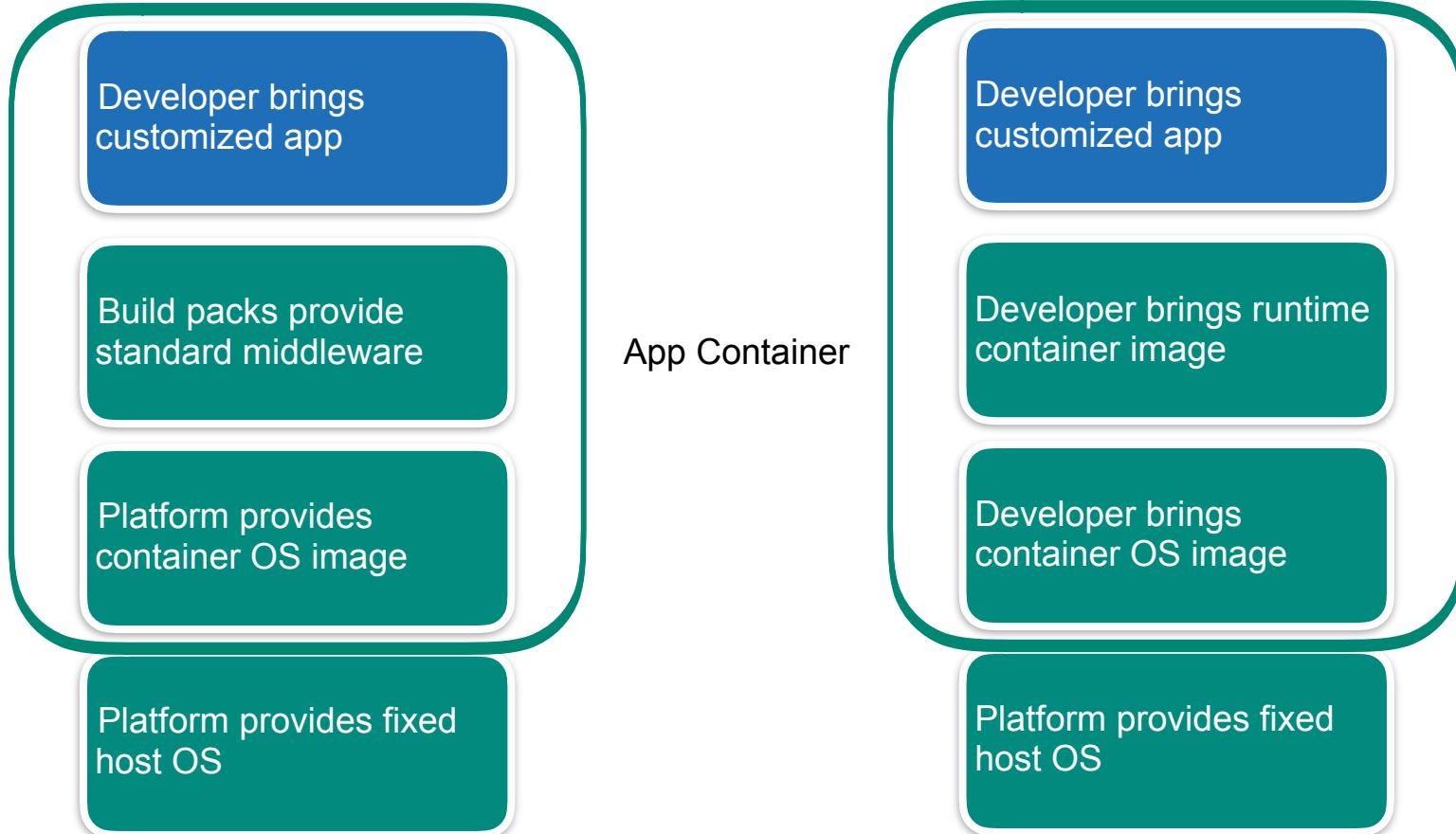
Container Environment Variables

- Used to communicate apps environment/config to deployed container
 - `VCAP_APPLICATION`
 - Application attributes – version, instance index, limits, URLs, etc.
 - `VCAP_SERVICES`
 - Bound services – name, label, credentials, etc.
 - `CF_INSTANCE_*`
 - `CF_INSTANCE_ADDR`, `CF_INSTANCE_INDEX`, etc.

VCAP_APPLICATION

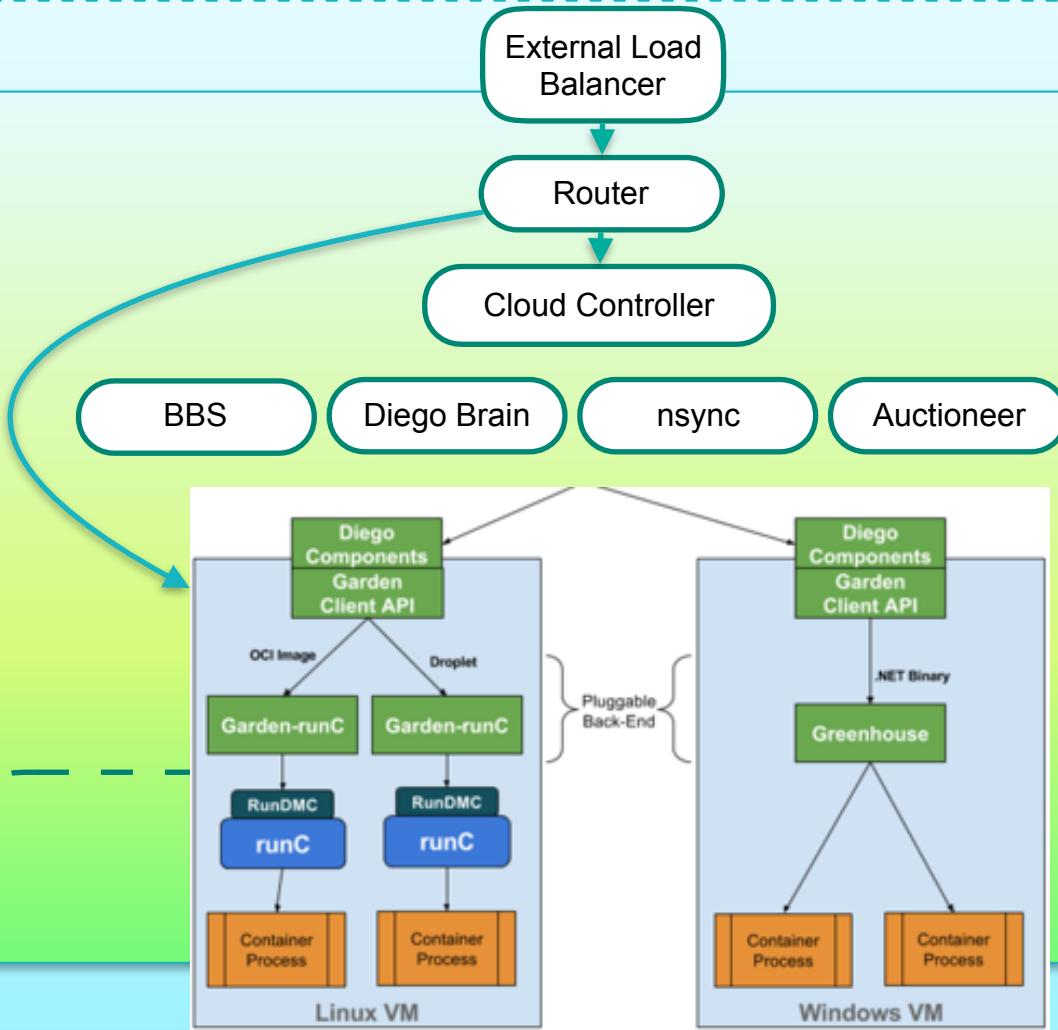
```
"VCAP_APPLICATION": {  
  "application_id": "95bb5b8e-3d3b-4753-85ee-7d9d503aef7c",  
  "application_name": "fortuneService",  
  "application_uris": [  
    "fortuneservice-glotto-logic-neigh.apps.testcloud.com"  
  ],  
  "application_version": "40933f4c-75c5-4c61-b369-018febb0a347",  
  "cf_api": "https://api.system.testcloud.com",  
  "limits": {  
    "disk": 1024,  
    "fds": 16384,  
    "mem": 512  
  },  
  "name": "fortuneService",  
  "space_id": "86111584-e059-4eb0-b2e6-e89uu260453d",  
  "space_name": "test",  
  "uris": [  
    "fortuneservice-glotto-logic-neigh.apps.testcloud.com"  
  ],  
  "users": null,  
  "version": "40933f4c-75c5-4c61-b369-018febb0a347"  
}
```

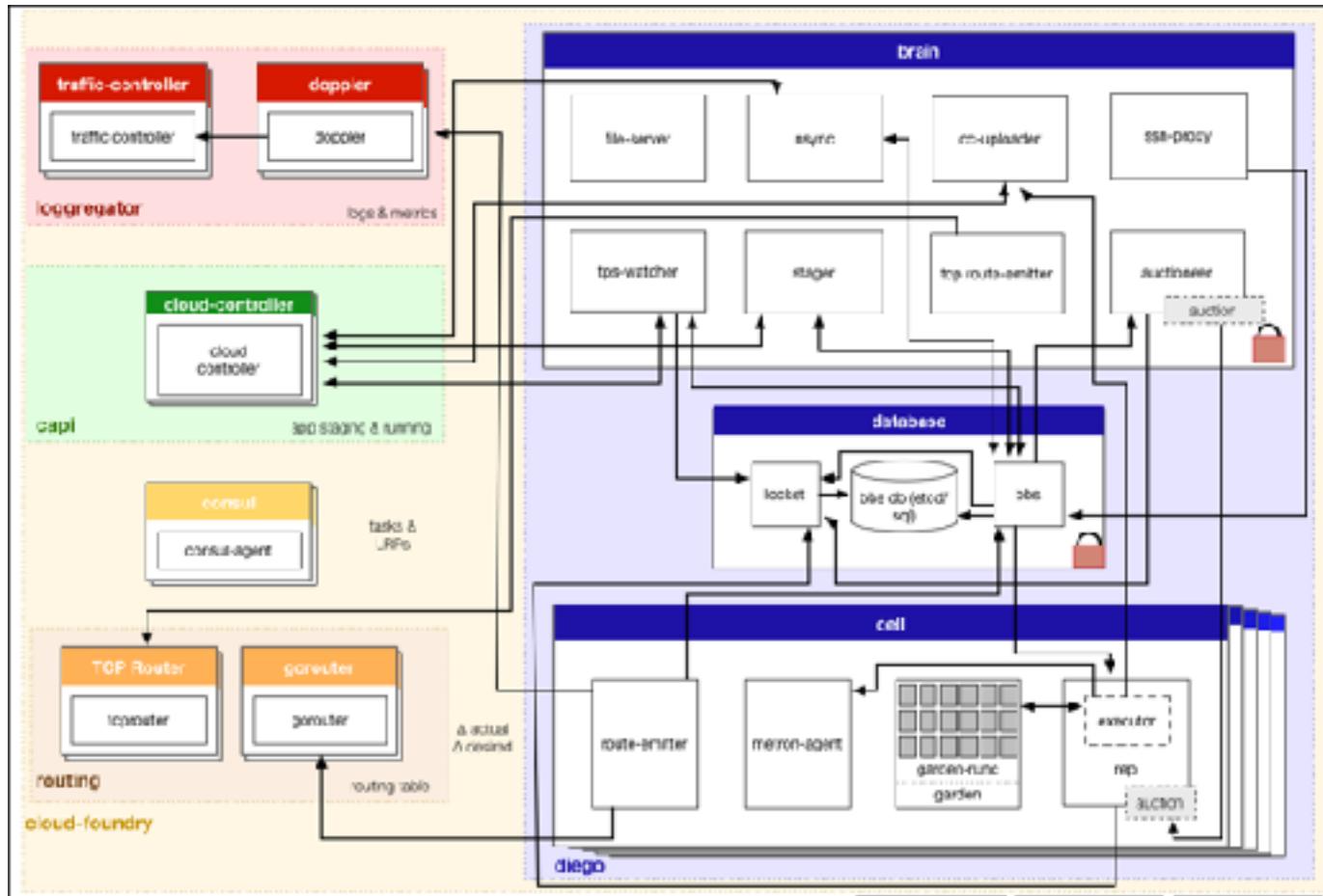
2 Workflows: Runtime vs No-Runtime



VNET

Subnets





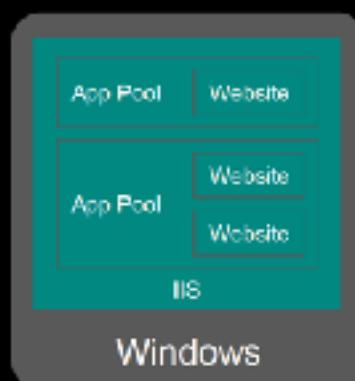
Resilience and High Availability

Why Containers

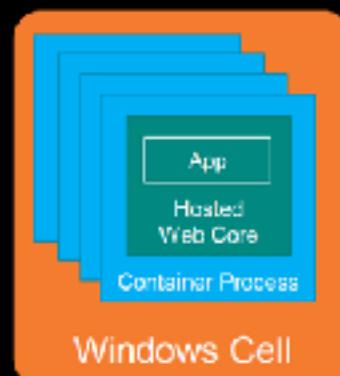
- Containers are OS level virtualization (i.e. process isolation)
- They are small and allow for much higher packing density
- They are easy to move around and to replicate
- They do not have any redundant or unnecessary operating system elements; they don't need the care and feeding of a large OS stack.
- They are lightweight and have fast startup times,
- Well suited for building hyper-scale, highly resilient infrastructure
- Typical container image is 10s of MB
- Containers start in msecs

Windows Cells on Cloud Foundry

Traditional Windows Architecture

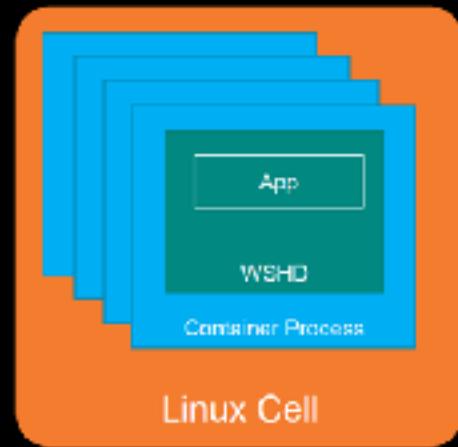


Cloud Foundry Architecture

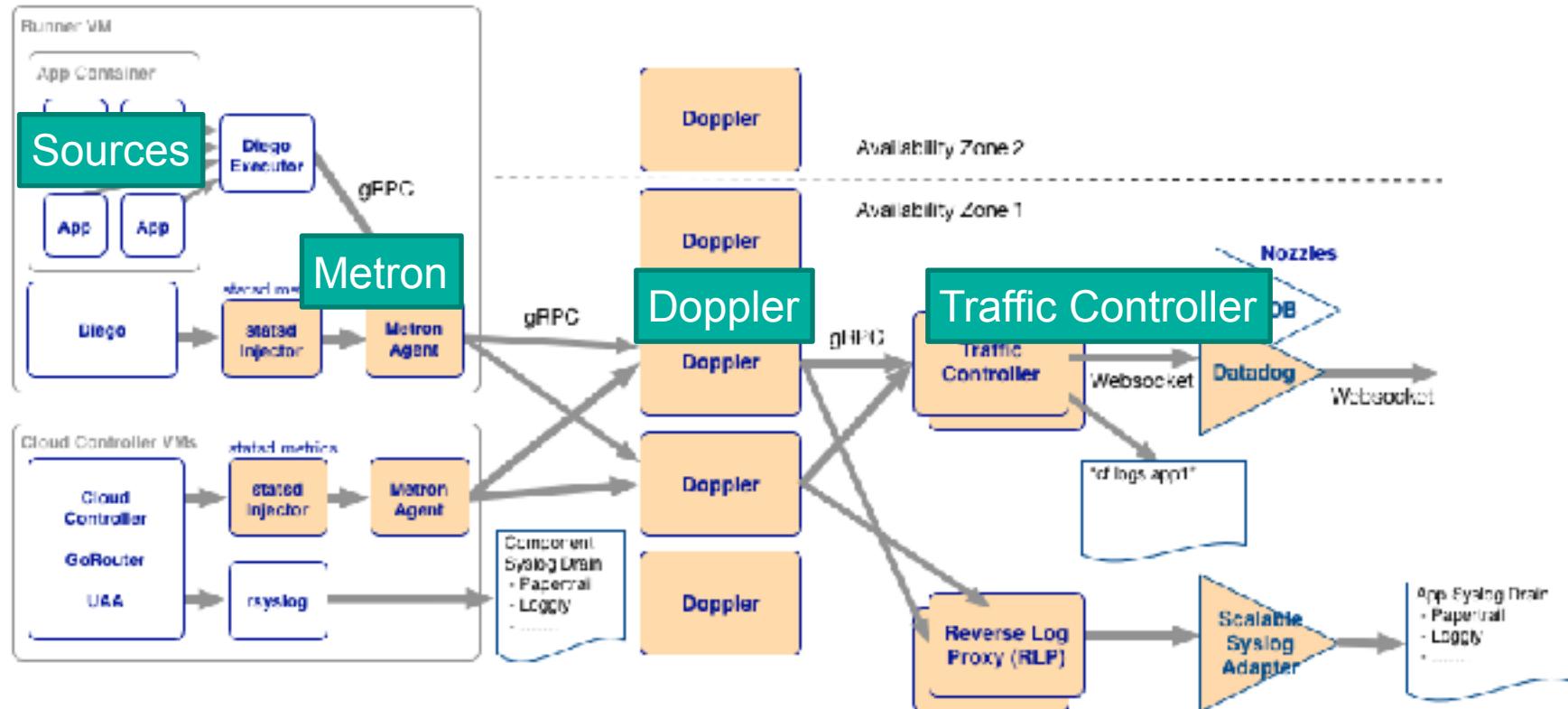


Linux Cells on Cloud Foundry

Cloud Foundry Architecture



Debugging Apps



Application

PCF Metrics

Network

Container

Log
Events

TX Trace
(coming soon)

Platform

PCF JMX Bridge

API
Controller

Router

VM

Workload
Director

3rd Party

Tier 1

Dynatrace

App
Dynamics

Splunk

Tier 2

New Relic

Interscope

Knowtify

3rd Party

VMware

Data Dog

Any JMX
Visualizer

Logging

PCF Log Search

Search

Filter

Visualize

3rd Party

14+ Days

External Logs

Auditing

PCF Extensions

Anti Virus

Route Services

3rd Party

Syslog Aggregator

Apigee

Custom

Logs

- Cloud Foundry aggregates an applications logs
 - Application logs should be written to STDOUT /STDERR
- Use the CLI to view an applications logs
 - `cf logs APP_NAME` – allows you to tail and applications logs
 - `cf logs APP_NAME -recent` – allows you to view recent logs

```
% cf logs pcf-scale-prod --recent
Connected, displaying recent logs for app pcf-scale-prod in org abergen-erg / space development as abergen@pivotal-lab...
2015-12-29T20:32:08.07+0000 [App/0]      00T Updated app with guid b828500c-1e91-495f-8628-54874239795e ([{"route": "v1/pcf-scale/v1.2.0/auth", "method": "GET", "path": "/v1/pcf-scale/v1.2.0/auth"}, {"route": "v1/pcf-scale/v1.2.0/auth", "method": "POST", "path": "/v1/pcf-scale/v1.2.0/auth"}, {"route": "v1/pcf-scale/v1.2.0/auth", "method": "PUT", "path": "/v1/pcf-scale/v1.2.0/auth"}, {"route": "v1/pcf-scale/v1.2.0/auth", "method": "DELETE", "path": "/v1/pcf-scale/v1.2.0/auth"}], [{"method": "GET", "path": "/v1/pcf-scale/v1.2.0/auth"}])
2015-12-29T20:40:02.03+0000 [App/0]      00T pcf-scale-v1.2.0-auth [http://pivotallab-104-40-02-48888]:8080 GET / HTTP/1.1 200 5355 0 1ms /v1/pcf-scale/v1.2.0/auth
at /v1/pcf-scale/v1.2.0/auth?username=9840.890&password=4E7A-8B7-1e287139C9E8&spec=a/0e92f524+e324+4c01/5ebe88ba542e/1464/pcf-scale/v1.2.0/auth?username=9840.890&password=4E7A-8B7-1e287139C9E8
2015-12-29T20:40:02.03+0000 [App/0]      00T Intel Mac OS X 10.11.2 App/WebKit/537.36 (KHTML, like Gecko) Chrome/47.0.2526.80 Safari/537.36 192.168.5.164:4081 X-Forwarded-For:178.2.160.5.17 X-Forwarded-For:10.0.2.150/4.1e91-8628-54874239795e
2015-12-29T20:40:02.03+0000 [App/0]      00T response_localhost:10.0.2.150/4.1e91-8628-54874239795e
2015-12-29T20:40:02.03+0000 [App/0]      ERR 192.168.5.1. 192.168.5.38 - - [29/Dec/2015:02:40:02 +0000] "GET / HTTP/1.1" 200 5355 0.846s
2015-12-29T20:40:02.03+0000 [App/0]      ERR 192.168.5.1. 192.168.5.38 - - [29/Dec/2015:02:40:02 +0000] "GET /v1/pcf-scale/v1.2.0/auth" 200 133014 0.402s
2015-12-29T20:40:02.03+0000 [RTRW]      00T pcf-scale-v1.2.0-auth [http://pivotallab-104-40-02-48888]:8080 GET /v1/pcf-scale/v1.2.0/auth?username=9840.890&password=4E7A-8B7-1e287139C9E8
2015-12-29T20:40:02.03+0000 [App/0]      00T pcf-scale-v1.2.0-auth [http://pivotallab-104-40-02-48888]:8080 GET /v1/pcf-scale/v1.2.0/auth?username=9840.890&password=4E7A-8B7-1e287139C9E8
2015-12-29T20:40:02.03+0000 [App/0]      00T Intel Mac OS X 10.11.2 App/WebKit/537.36 (KHTML, like Gecko) Chrome/47.0.2526.80 Safari/537.36 192.168.5.164:4081 X-Forwarded-For:178.2.160.5.17 X-Forwarded-For:10.0.2.150/4.1e91-8628-54874239795e
2015-12-29T20:40:02.03+0000 [App/0]      00T 192.168.5.1. 192.168.5.38 - - [29/Dec/2015:02:40:02 +0000] "GET /v1/pcf-scale/v1.2.0/auth" 200 51595 1.3139s
2015-12-29T20:40:02.03+0000 [RTRW]      00T pcf-scale-v1.2.0-auth [http://pivotallab-104-40-02-48888]:8080 GET /v1/pcf-scale/v1.2.0/auth?username=9840.890&password=4E7A-8B7-1e287139C9E8
2015-12-29T20:40:02.03+0000 [App/0]      00T Intel Mac OS X 10.11.2 App/WebKit/537.36 (KHTML, like Gecko) Chrome/47.0.2526.80 Safari/537.36 192.168.5.164:4081 X-Forwarded-For:178.2.160.5.17 X-Forwarded-For:10.0.2.150/4.1e91-8628-54874239795e
```

Health

- Cloud Foundry proactively monitors health of application containers
 - Restarts them if they fail
- Use the CLI to view an applications health & status
 - `cf app APP_NAME` – allows you view health status of an application

```
>>> cf app pcf-scale-prod
Showing health and status for app pcf-scale-prod in org mborges-org / space development as mborges@pivotal.io...
OK

requested state: started
instances: 1/1
usage: 128M x 1 instances
urls: pcf-scale-prod.south.re.pivotal.io, pcf-scale-v1_2.south.re.pivotal.io
last uploaded: Tue Dec 22 23:56:58 UTC 2015
stack: cflinuxfs2
buildpack: Ruby

     state      since          cpu    memory       disk      details
#0  running   2015-12-22 06:02:59 PM  0.1%  80.0M of 120M  04.2M of 16
```

Application Events

- Cloud Foundry records all changes to an application as events
 - Container state changes
 - Configuration changes
- Use the CLI to view an applications events
 - `cf events APP_NAME` – allows you view recent events of an application

```
[cf] events pdf-scale-prod
Getting events for app pdf-scale-prod in org abergo@org / space development as abergo@pivotal.io...
time          event        actor           description
2015-12-23T20:32:00.00+0000 audit.app.update  abergo@pivotal.io
2015-12-23T20:32:00.00+0000 audit.app.add-route abergo@pivotal.io
2015-12-23T18:47:55.00+0000 audit.app.update  abergo@pivotal.io  state: STARTED
2015-12-22T18:47:55.00+0000 audit.app.update  abergo@pivotal.io  state: STOPPED
2015-12-22T17:55:55.00+0000 audit.app.update  abergo@pivotal.io  state: STARTED
2015-12-22T17:55:45.00+0000 audit.app.add-route abergo@pivotal.io
2015-12-22T17:55:44.00+0000 audit.app.create   abergo@pivotal.io  instances: 1, memory: 128, state: STOPPED, environment_json: PRIVATE DATA HIDDEN
```

Services

What is a Service?

- Allows resources to be easily provisioned on-demand
- Typically an external “component” necessary for applications
 - Database, cache, message queue, microservice, etc.
- Can be a persistent, stateful layer



Types of Services

- Managed - Fully integrated, with fully lifecycle management
- User-Provided – Created and managed external to the platform



Managed Services

- Integrated with Cloud Foundry
 - Implements a required API for which the cloud controller is the client
- **Service Broker** implements the required API
 - Advertise a catalog of service offerings and service plans
 - Handle calls from the Cloud Controller
 - Fetch catalog
 - Create service instances
 - Bind applications to service instances
 - Unbind applications from service instances
 - Delete service instances

User Provided Services

- Service instances managed outside of Cloud Foundry
- Behave like other service instances once created
- Familiar CLI commands ('create-service') provide service instance configuration

EXAMPLE: AN ORACLE DATABASE MANAGED OUTSIDE OF, AND UNKNOWN TO CLOUD FOUNDRY

Examples of Managed Services



Pivotal Network

<https://network.pivotal.io/>



Pivotal Cloud Foundry Service Broker for
AWS



GemFire for PCF



Push Notification for PCF



MySQL for PCF



Redis for PCF



Pivotal Tracker for PCF



Session State Caching Powered by
GemFire for PCF



RabbitMQ for PCF

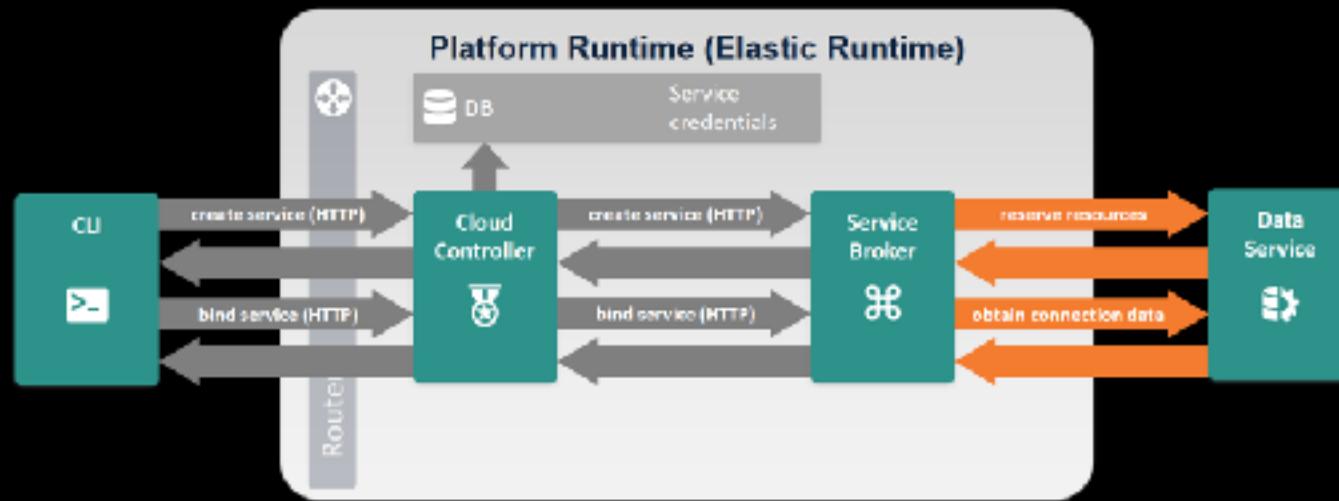


Spring Cloud Services for PCF



Single Sign-On for PCF

Creating and Binding Services



Container Environment Variables

- Used to communicate apps environment/config to deployed container
 - VCAP_APPLICATION
 - Application attributes – version, instance index, limits, URLs, etc.
 - VCAP_SERVICES
 - Bound services – name, label, credentials, etc.
 - CF_INSTANCE_*
 - CF_INSTANCE_ADDR, CF_INSTANCE_INDEX, etc.

VCAP_SERVICES

```
"VCAP_SERVICES": {  
  "p-identity": [  
    {  
      "credentials": {  
        "client_id": "e3ca311d-999b-4e4f-b056-b50138cff9f",  
        "client_secret": "a995365e-d7b7-4727-95b8-463df2842f64",  
        "auth_domain": "https://sso1.login.run.haas-76.pez.pivotal.io"  
      },  
      "syslog_drain_url": null,  
      "label": "p-identity",  
      "provider": null,  
      "plan": "sso1",  
      "name": "sso",  
      "tags": []  
    }  
  ]  
}
```

Domains and Routes

Domains

- Each Cloud Foundry installation has a default app domain
- Domains provide a namespace from which to create routes
- Requests for any routes created from the domain will be routed to Cloud Foundry.
- Domains can be shared or private in regards to PCF organizations

The screenshot shows the Pivotal Pipeline Manager interface. On the left, a sidebar menu includes 'mborges.org' (selected), 'development', 'production', 'Marketplace', 'Docs', 'Support', and 'Tools'. The main content area displays organization details for 'mborges-org':

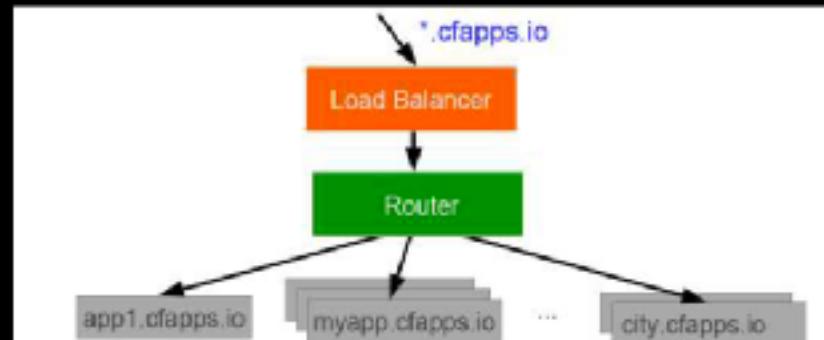
- ORG:** mborges-org
- QUOTA:** 128 MB of 5 GB Used
- Spaces:** 2 Spaces
- Domains:** 1 Domain
- Members:** 2 Members

A button labeled 'Add a Domain' is visible. Below this, a table lists domains:

NAME	OWNER
southlife.pivotal.io	mborges

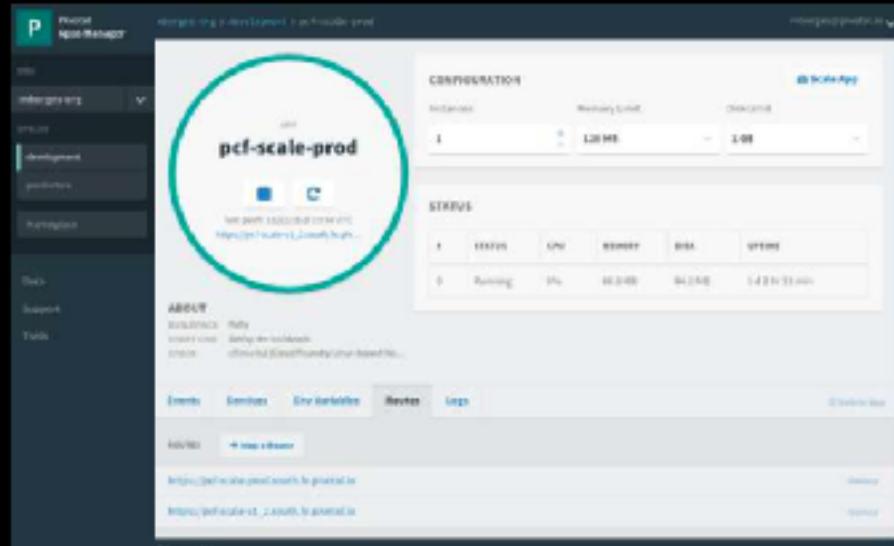
Domains – Behind the Scenes

- A wildcard entry (*) is added to the DNS for the app domain
- That DNS entry points to a load balancer (or Cloud Foundry's HA Proxy), which points to the Cloud Foundry Router
- The Router uses the subdomain to map to application instance(s)

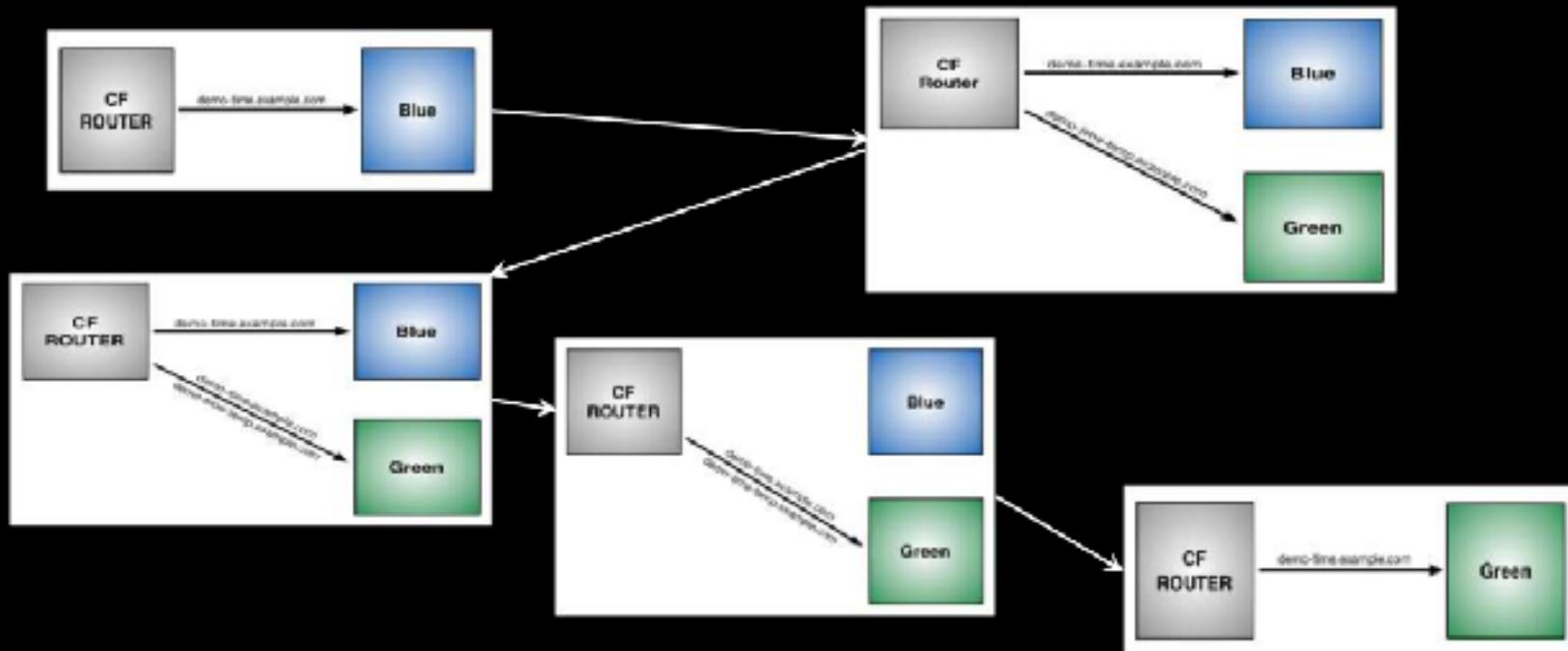


Routes

- HTTP requests are routed to apps pushed by associating a URL with an application, known as route
- Many app instances can be mapped to a single route resulting in load balanced requests
- Routes belong to a space
- Application can have multiple routes



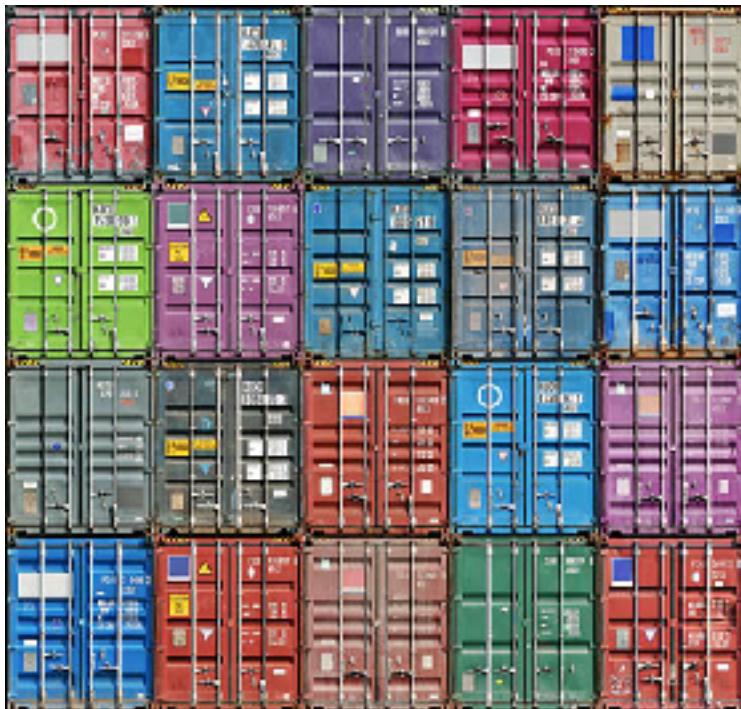
Blue-Green Deployments



<https://docs.pivotal.io/pivotalcf/1-8/devguide/deploy-apps/blue-green.html>

Containers

Container Principles



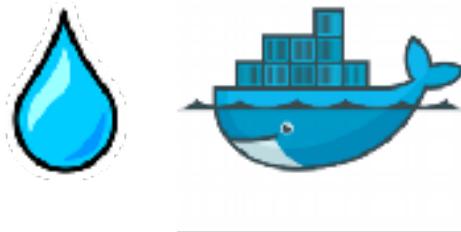
Control Resources
Isolate and Secure Processes

1. Quick To Create
2. Resource Consolidation
3. App Portability

Container Principles

File System

Docker Images
Droplets+Stack



Management

API-CLI

Docker-CLI
Garden



Runtime

Docker-Engine
Guardian

Implementation

RunC



cgroups

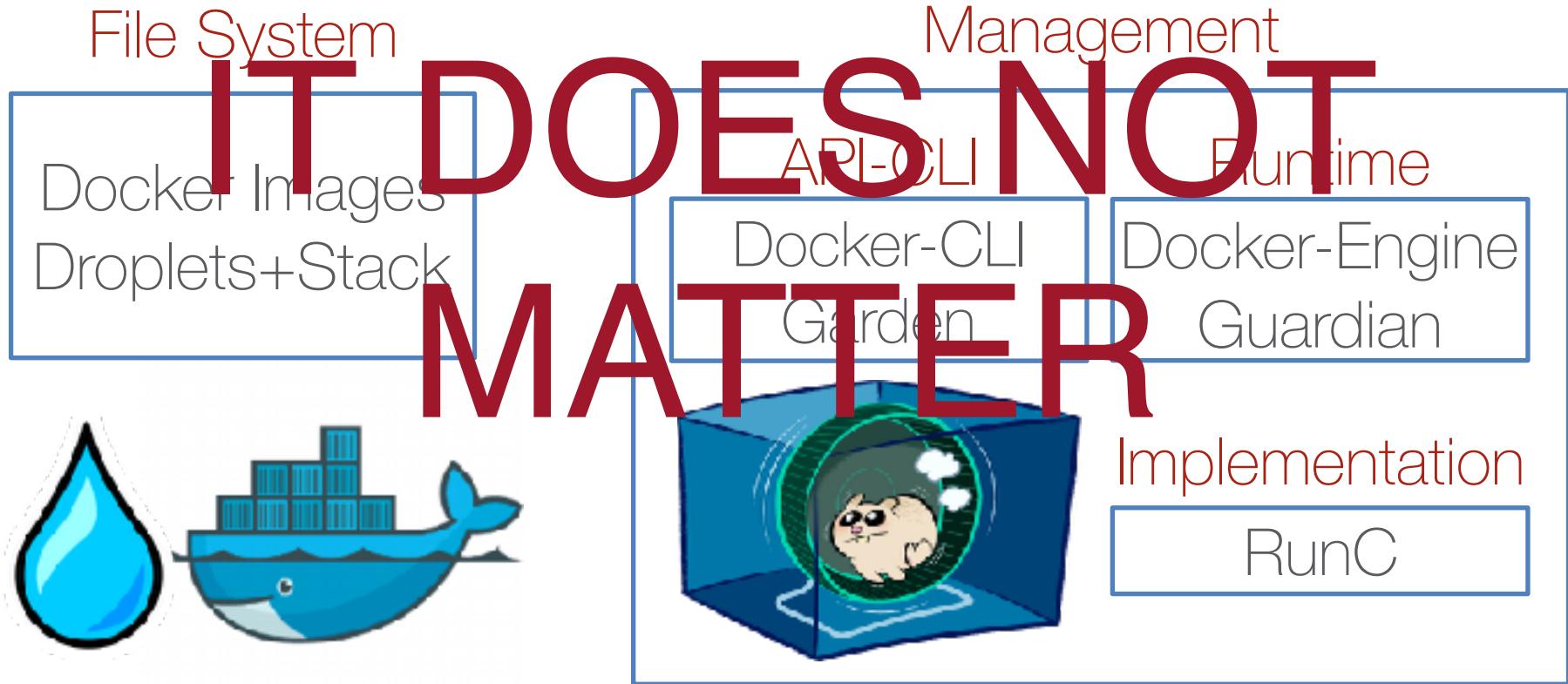
namespaces

Containers Misconceptions

Walls
Resource Limits
Namespace



Container Misconceptions





Read how our performance benchmark (250K Containers in Prod!) for [@CloudFoundry](#) supports real-world scenarios.

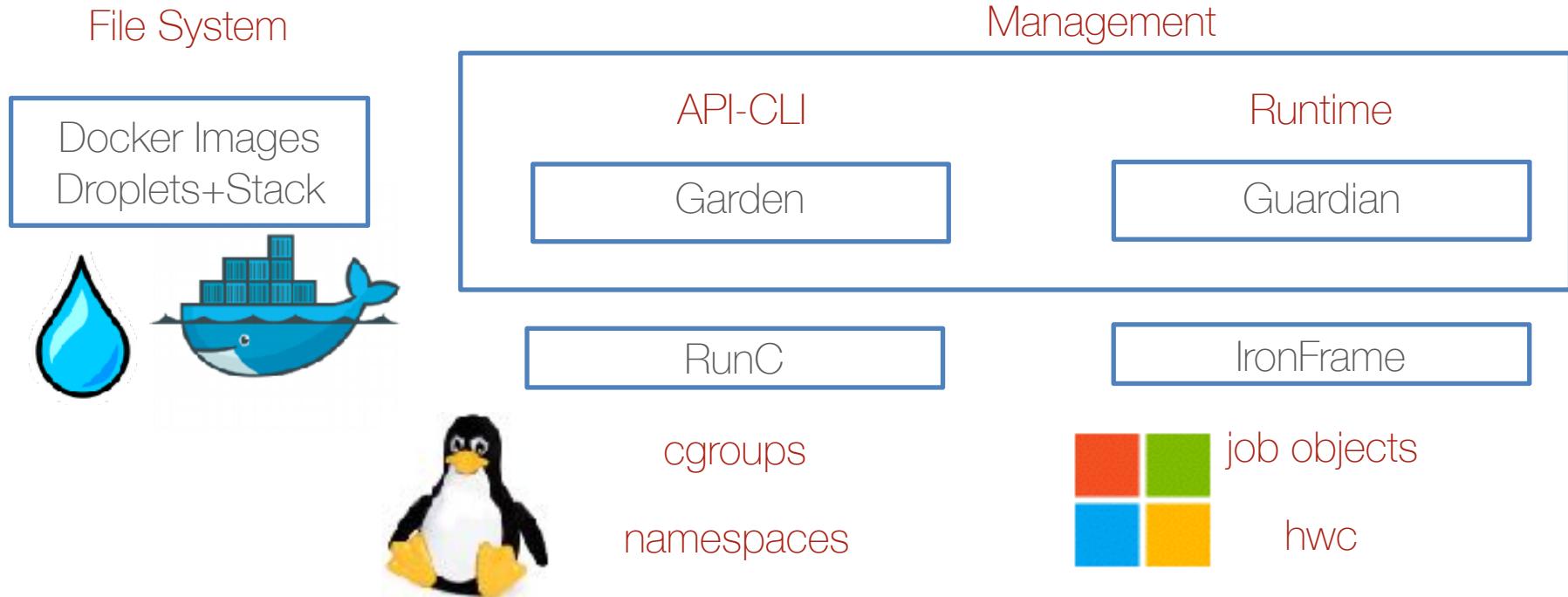


250k Containers In Production: A Real Test For The Real World

Real world conditions and results for a Cloud Foundry performance test that sustained an astonishing 250,000 containers in one environment. #cloudfoundry ...

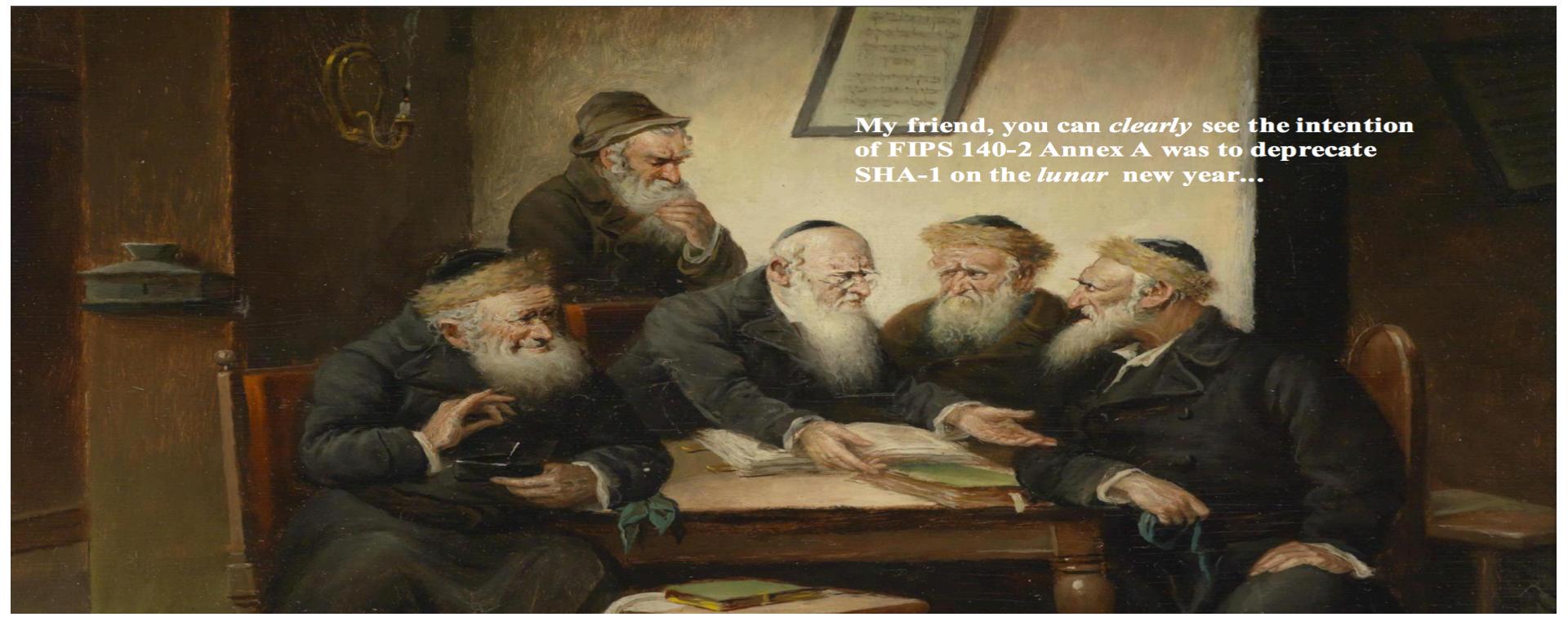
blog.pivotal.io

Containers In Cloud Foundry



Security

Compliance and Security



My friend, you can *clearly* see the intention
of FIPS 140-2 Annex A was to deprecate
SHA-1 on the *lunar* new year...

Speed is the new security

Automation Pays Dividends



1. Consistency via pipelines
2. Continuous Deployment
3. Auditing
4. Security/Immutability
5. Upgradability
6. Recovery

Platform Trust

1. Repave, Repair, Rotate - more secure architecture
2. Zero downtime deployments and upgrades
3. Explicit, known dependencies
4. Homogeneity in the environment, no drift
5. Combined CVE / vulnerability management
6. Federated login with enterprise directories
7. Control access to services and other resources
8. Enforce policies (server, runtime, libraries, etc.)
9. IPSec on every network segment
10. Custom monitoring agents
11. RunC integration for AppArmor and user namespaces

Additional Host Security

1. Automated Pipelines to track and update Stemcells and buildpacks
2. Stemcell hardening
3. Vulnerability Scan - Nessus scan
4. Integrity and Intrusion management – TripWire
5. Anti-virus
6. Constant automated code quality testing
7. Automated Pen testing scans
8. Dashboard and alerts (e.g. <https://compliance-viewer.18f.gov/>)

Bold Audacious Moves

60%

Computing
workloads to
the cloud

75%

Technology
staff writing
code

50%

Code to
production
in a day

In Summary