
Pivotal
Cloud Native Operations
Pivotal

DEVELOPMENT

OPERATIONS

Multiple Languages — Java, Microsoft .NET, node.js, ...

Microservices Support — Spring Boot, Spring Cloud

Services Marketplace — Native, User Provided, Partner

App Deployment & Management — CI/CD Tools, ID, Security

Availability — Health, Metrics Patching

Visibility & Administration — Apps & Platform Dashboards

VALUE LINE

Operating System

Container Orchestration

Cloud API

Google | AWS | Azure | VMW | Openstack

Pivotal

DEVELOPMENT

Multiple Languages

Java

Microsoft .NET

node JS

...

Microservices Support

Spring Boot

Spring Cloud

Services Marketplace

Native

User Provided

Partner

OPERATIONS

App Deployment & Management

CI/CD Tools, ID, Security

Availability

Health, Metrics, Patching

Visibility & Administration

Apps & Platform Dashboards

Operating System

Container Orchestration

Cloud API

Google

AWS

Azure

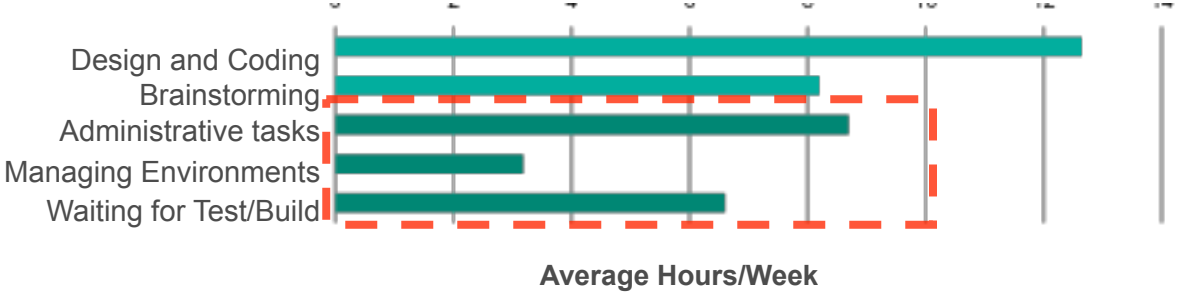
VMW

Openstack

Pivotal

Streamlining Developer Workflows

Reducing Overhead



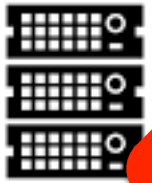
Source: "Electric Cloud LinkedIn Survey to software developers"

Snowflake Automation

- Complex/Custom Configurations
- Ad hoc, Incomplete Automation
- Redundant work for cross-cutting concerns
- Difficult to Port/Replicate
- Maintenance Overhead
- Infrastructure Failure Sensitive

Manual, Slow, Inflexible, Error-Prone

Structured Automation



Platform

- ✓ Consistent Contra
- ✓ Fully Automate Repeatable platform managed DevOps processes
- ✓ Develo + Ops Friendly Con ucts
- ✓ E y to move/re-architect apps
- ✓ Infrastructure Failure Agnostic

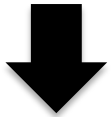
Automated, Fast, Flexible, Safe

Pivotal

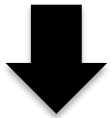
Full Stack IT Ticket Elimination

Code Complete & Tested

cf push



Speed
&
Consistency



Find available hosts	--- 2 Days
Install & configure runtime	--- 1 Day
Install & configure middleware	--- 1 Day
Pull application source code	--- ¼ Day
Retrieve dependant libraries	--- ¼ Day
Create application package	--- ¼ Day
Install, configure dependent service(s)	--- 2 Days
Deploy container to host(s)	--- ½ Day
Load environment variables	--- ¼ Day
Configure load balancer	--- 2 Days
Configure firewalls	--- 2 Days
Update service monitoring tools	--- 3 Days
Configure log collector	--- 1 Day
…	…

~45 seconds

Application in Production

~15+ Days

Deployment & Buildpacks

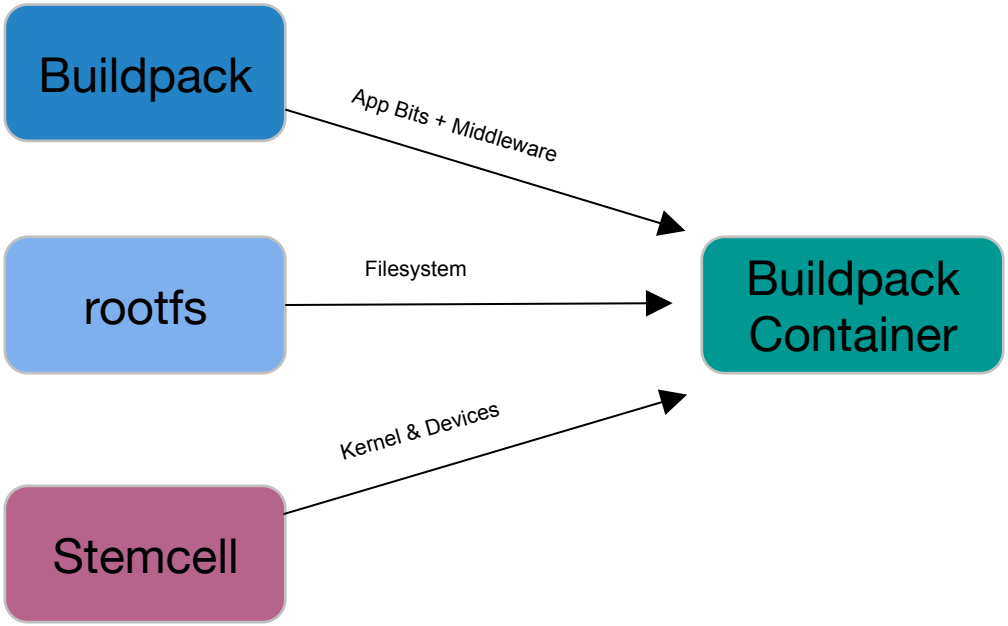
- Single tool-chain to detect, stage and run any application
- Reduce dev technical/security debt with buildpacks. Devs build apps, buildpacks build/maintain rest of the stack.
  - Buildpacks can provide an appropriate runtime and additional libraries for the apps to be run if needed.
  - Buildpacks package app and runtime into a single runnable artifact. The Platform then runs this within a new Linux container.
  - Platform managed rootfs is mounted in every container when using buildpacks as opposed to dev's creating/maintaining own Docker images with their choice of Linux.
- Helps devs be cloud native by adopting 12 factor guidelines

Ingredients of a Buildpack Container

- Buildpacks are scripts that can detect applications, install middleware runtimes and package app bits into it.
- Also provides scripts to run the application with its dependencies.

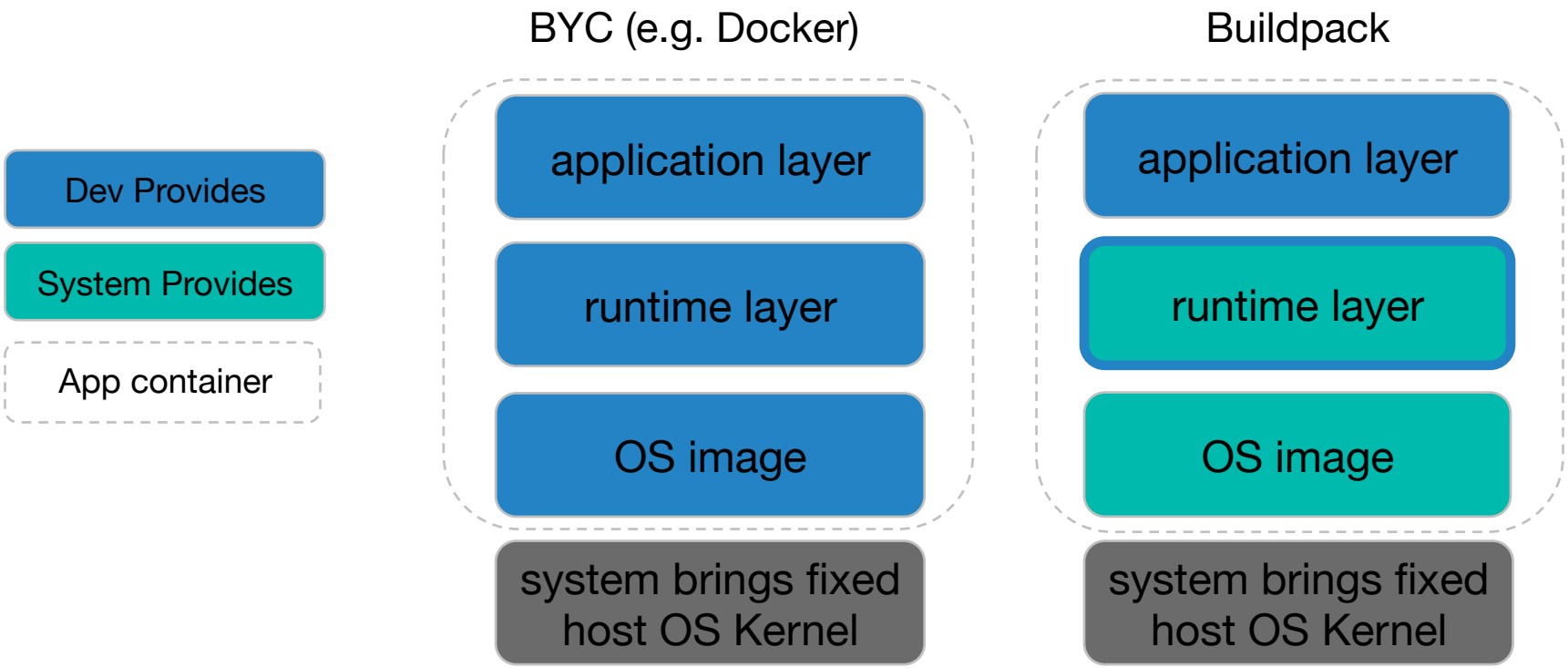
- Every container receives a Filesystem image purpose built for CF.
- Only necessary dependencies are added to reduce the attack surface and maintainability of these dependencies for security patches

- Provides kernel and its dependencies, devices, network interfaces, semaphores(for compilers)
- Primarily Ubuntu 14.04 with support for RHEL7/ CentOS7 and Windows Server 2012 coming soon.



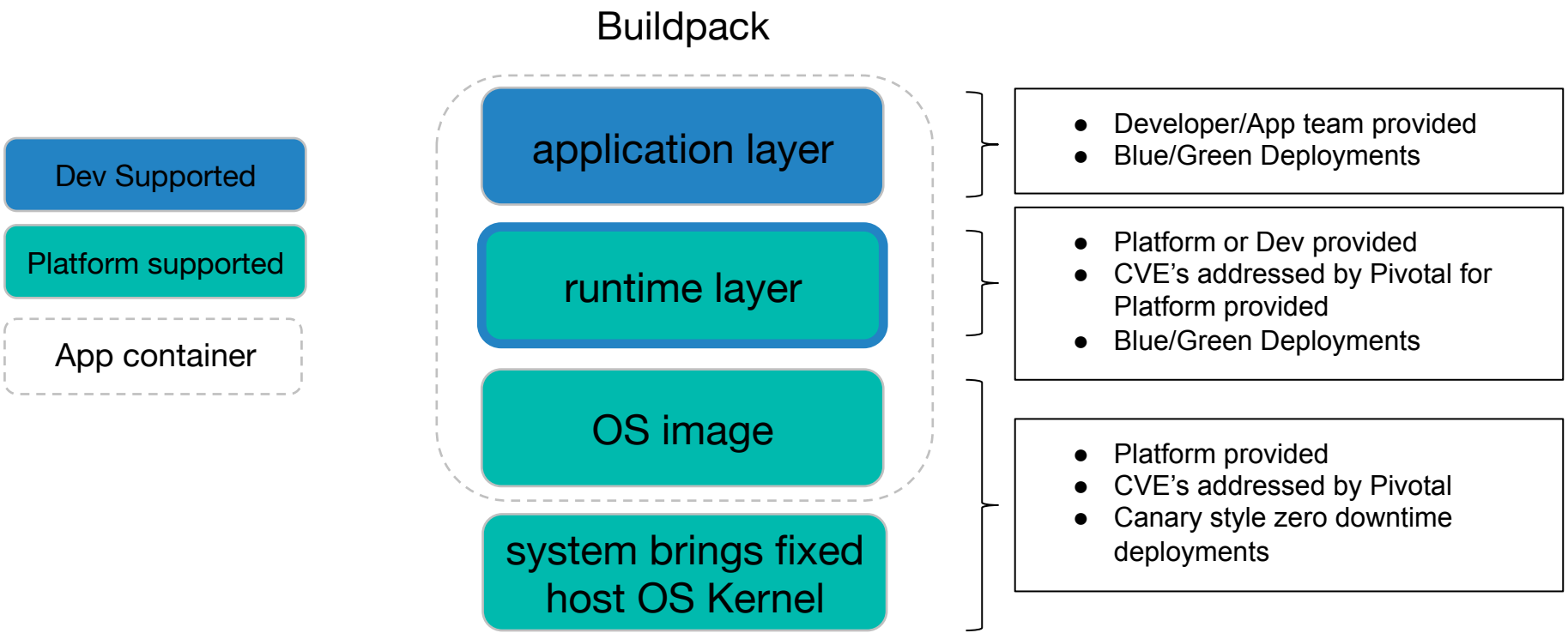
Pivotal

Bring your Container(BYC) vs Buildpacks



Pivotal

Day 2 Operations with Buildpacks



Pivotal

What is in a Buildpack?

- Idea and format of a buildpack was borrowed from Heroku. Many Buildpacks supported on Cloud Foundry derived from Heroku Buildpacks.

- Buildpacks are built with a simple API of three ***bash scripts***
  - ./bin/detect app_directory - Inspect app bits to determine type of application.
  - ./bin/compile app_directory cache_directory - Installing the appropriate run-time, packages and libraries as well as packaging the application bits with the runtime if needed.
  - ./bin/release - Returns a YAML payload that describes the default way of how the application needs to be started.

- Usage within Cloud Foundry :-

  "$ cf push *app_name"* – Detect scripts of the buildpacks are called to get the application type

  "$ cf push *app_name* -b *buildpack_name"* – Uses the specified installed buildpack.

  "$ cf push *app_name* -b *buildpack_url"* – Uses a buildpack hosted at the specified URL.

Cloud Foundry Supported Buildpack

- Java – Web(Tomcat), Java, Spring, Play, Grails or any JVM language
- Staticfile – HTML, JavaScript or CSS. Supported by NGINX
- PHP – php-cli, php-cgi, php-fpm. Supported by Apache & NGINX.
- Binary – Run arbitrary binary web servers

Heroku Derived Buildpacks

- Ruby – Ruby, Rack & Rails
- Go – Go apps
- Node.js – node.js apps
- Python – Python apps powered by pip

Support any Language & Runtime

- Community buildpacks - Built by the CF community to support other languages and runtimes like Websphere, JBoss, ASP.NET, Swift, Haskell, tomEE, R, meteor, Jetty, Clojure, Mono
- Heroku 3rd party buildpacks - Built by the Heroku community to support multiple languages and runtimes.
- These supported, community or Heroku buildpacks can serve as an extension point for you to customize and run your applications in any language.
- You can also build your own custom buildpack by following the Buildpack API discussed before.

Continuous Delivery Automation

Continuous Delivery Pipeline

Tool Chain

Gitlab

Jenkins

Artifactory

Concourse

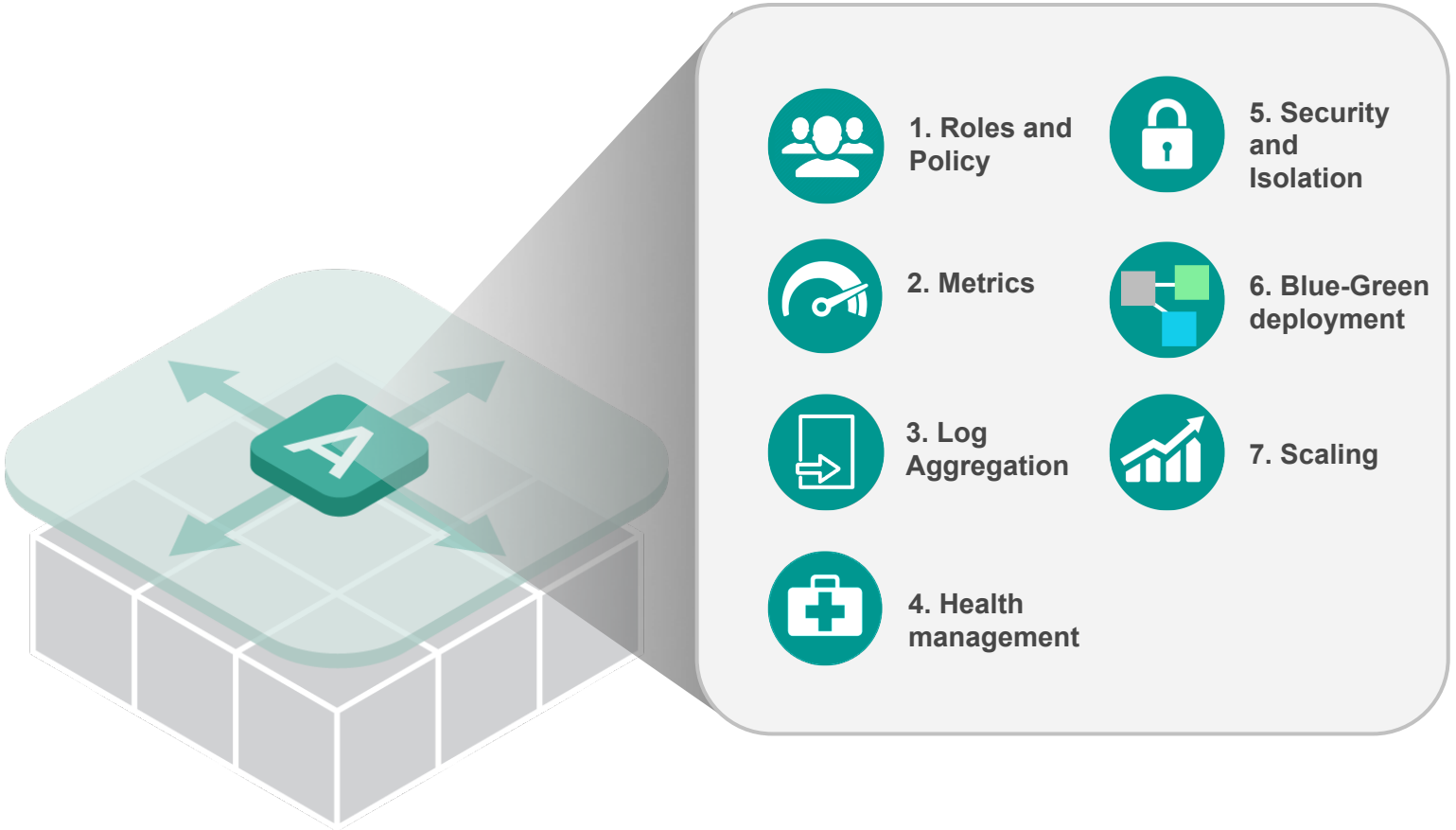
Pivotal Cloud Foundry

(API Driven)

Pivotal HA Services

Pivotal

Everything to Deploy and Manage the App



Pivotal

Improve Enterprise Security Culture

- Makes it easier to do the right things
- Prevents vulnerabilities due to
  - Leaked/misused credentials
  - Misconfigured/Unpatched software
  - Slow & rigid processes

Self-Service Provisioning + RBAC

Enhancing Speed & Consistency of Infrastruc

QA

Ops

Dev

CLI (cf Push, cf bind, cf start

REST

Apps Manager

User Role	Org Manager	Org Auditor	Space Manager	Space Develop
Scope of operation	Org	Org	Space	Space
Add and edit users and roles	✓		✓	
View users and roles	✓	✓	✓	✓
Create and assign Org and Space quota plans	✓			
View Org quota plans	✓	✓	✓	✓
Create Orgs	†	†	†	†
View Orgs	✓	✓	✓	✓
Edit, rename, and delete Orgs	✓			
Create Spaces	✓			
View Spaces	✓		✓	
Edit Spaces	✓		✓	
Delete Spaces	✓			
Rename Spaces	✓		✓	
View the status, number of instances, service bindings, and resource use of applications	✓		✓	✓
Add private domains†	✓			
Deploy, run, and manage applications				✓
Instantiate and bind services to applications				✓
Associate routes†, instance counts, memory allocation, and disk limit of applications				✓
Rename applications				✓

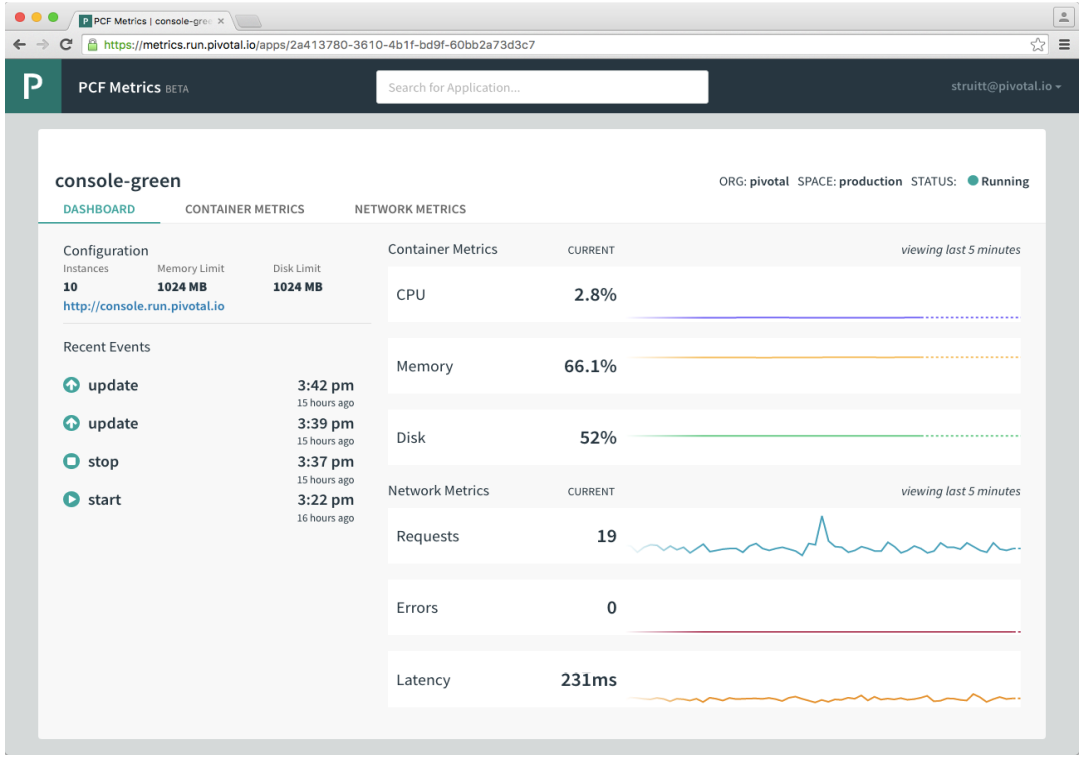
Name	Description	Valid Values	Example Value
name	The name you use to identify the plan	A sequence of letters, digits, and underscore characters. Quota plan names within an account must be unique.	silver_quota
memory_limit	Maximum memory usage allowed	An integer and a unit of measurement like M, MB, G, or GB	2048M
app_instance_limit	Maximum app instances allowed	An integer	25
non_basic_services_allowed	Determines whether users can provision instances of non-free service plans. Does not control plan visibility. When false, non-free service plans may be visible in the marketplace but instances can not be provisioned.	`true` or `false`	true
total_routes	Maximum routes allowed	An integer	500
total_reserved_route_ports	Maximum routes with reserved ports	An integer not greater than total_routes	60
total_services	Maximum services allowed	An integer	25
trial_db_allowed	Legacy Field. Value can be ignored.	`true` or `false`	true

Organization B

Pivotal

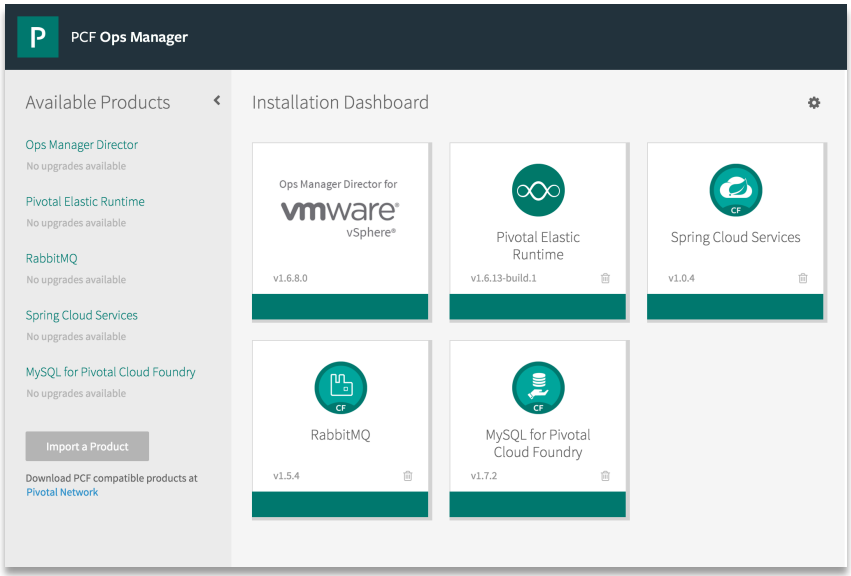
Fast-feedback Monitoring Capabilities

- Integrated, near real-time view of key app, services & platform metrics
- Devs & Ops have shared understanding of system health / availability
- No embedded agent
- Simply select an app to watch streaming data

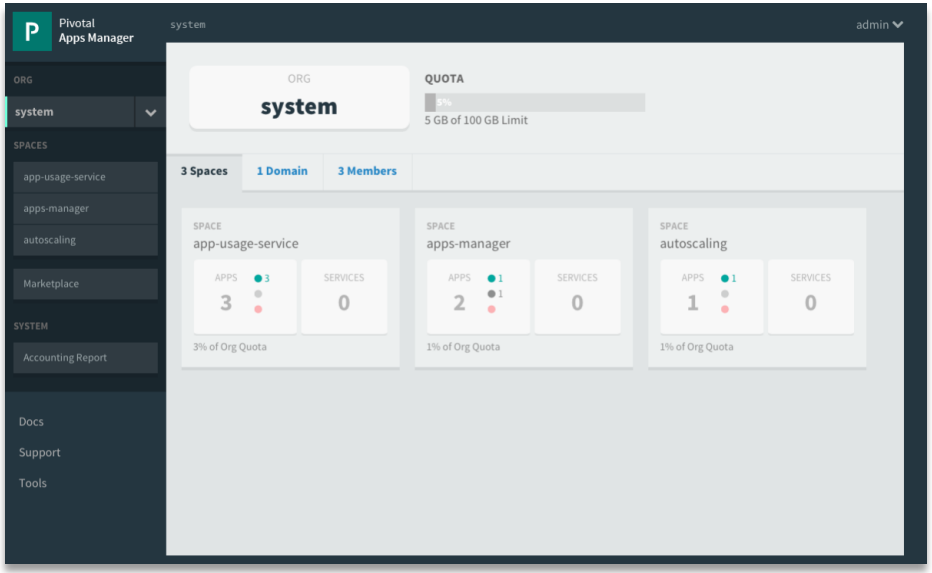


Pivotal

Comprehensive Visibility & Administration



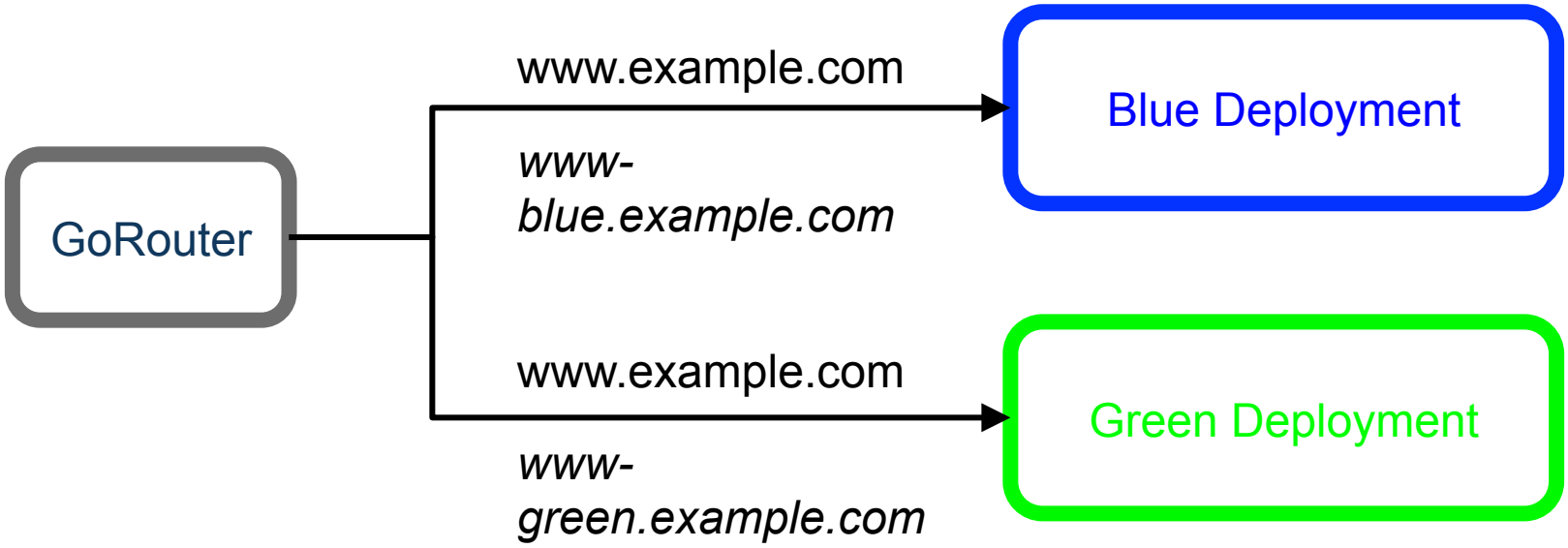
Operations Manager



Applications Manager

Pivotal

Blue / Green Deployments



Pivotal

Application Scaling

- Applications can be scaled 2 ways
  - Vertically - add more memory / disk per instance
  - Horizontally - add more instances
- Horizontally scaling can be manual or automatic
  - Autoscaling is bound as a service to an application
  - Autoscaling supports HTTP Throughput, HTTP Latency, CPU Utilization and scheduling

Platform Scaling

- Platform scaling is performed by the cloud operator
- Scaling is done in the resource config tab in Ops Manager
- For elastic runtime, only a few components typically need to be scaled for capacity
  - GoRouter
  - DEA / Cell
  - Loggregator / Doppler

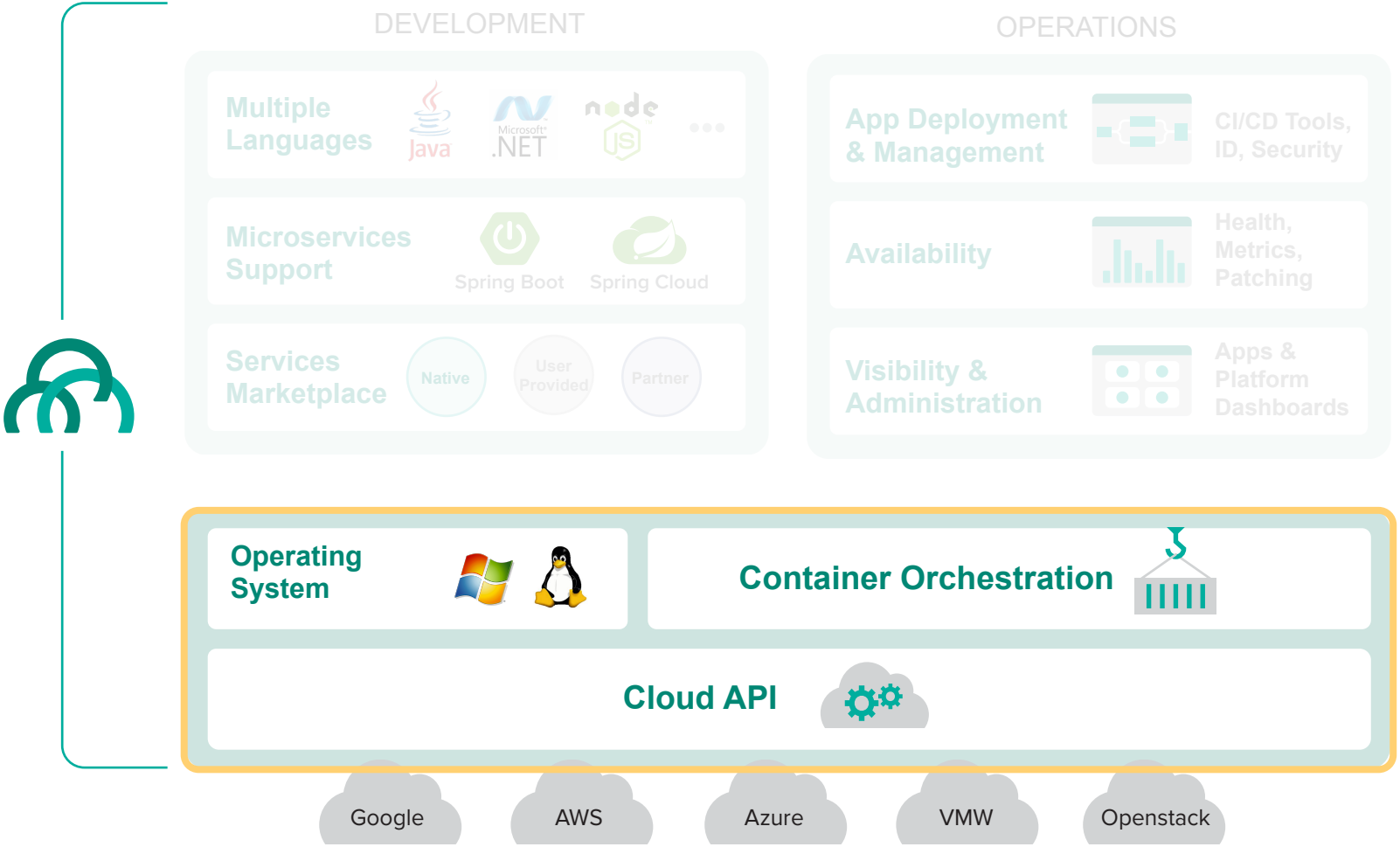
More information on platform scaling can be found here:
https://docs.pivotal.io/pivotalcf/concepts/high-availability.html#capacity

Platform Scaling

- File Storage Config
- IPs and Ports
- Security Config
- MySQL Proxy Config
- Cloud Controller
- External Endpoints
- SSO Config
- LDAP Config
- SMTP Config
- Experimental Features
- Errands
- Resource Config
- Stemcell

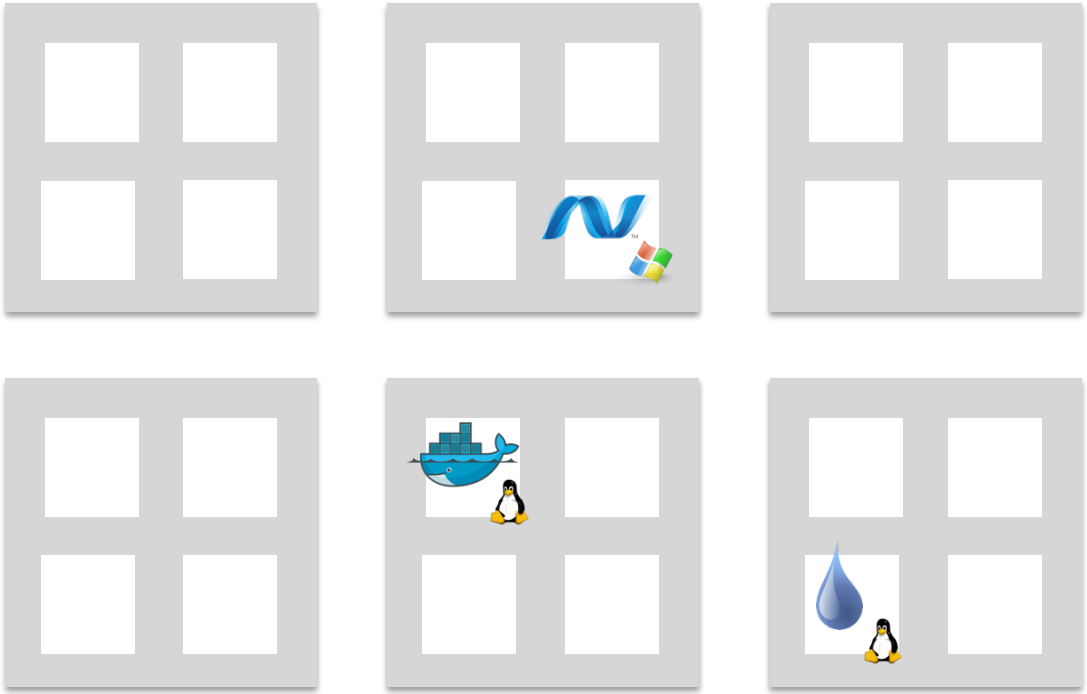
JOB	INSTANCES	CPU	RAM (MB)	EPHEMERAL DISK (MB)	PERSISTENT DISK (MB)
NATS	1	1	1024	2048	0
consul	1	1	1024	2048	1024
etcd	1	1	1024	2048	1024
NFS Server	1	1	1024	2048	102400
Cloud Controller Database	1	1	1024	2048	2048
UAA Database	1	1	1024	2048	8192
Apps Manager Database	1	1	1024	2048	1024
Cloud Controller	1	1	4096	20480	0
HAProxy	1	1	1024	2048	0
Router	1	1	1024	2048	0
Health Manager	1	1	1024	2048	0
Clock Global	1	1	1024	2048	0
Cloud Controller Worker	1	1	1024	2048	0
Collector	1	1	1024	2048	0
UAA	1	1	1024	2048	0
MySQL Proxy	1	1	1024	4096	0
MySQL Server	1	2	8192	30000	100000
DEA	6	2	16384	32768	0
Doppler Server	1	1	1024	2048	0
Loggregator Trafficcontroller	1	1	1024	2048	0
Push Apps Manager	1	1	1024	1024	0
Push App Usage Service	1	1	1024	1024	0
Run Smoke Tests	1	1	1024	1024	0

Pivotal

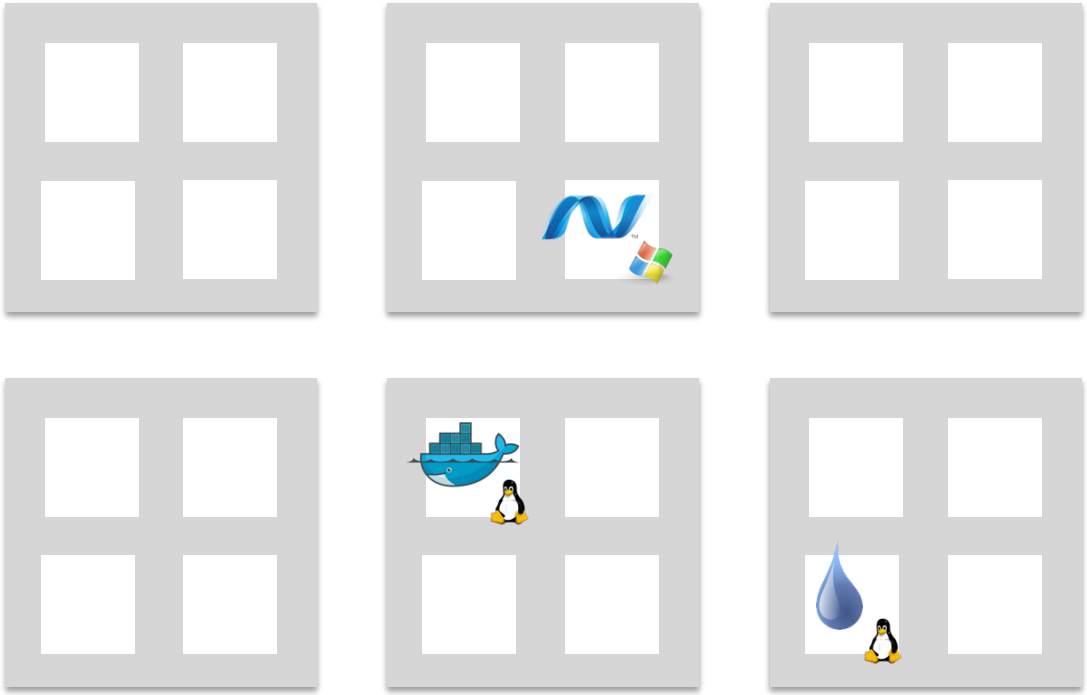

DEVELOPMENT
OPERATIONS
Multiple Languages
Java
Microsoft .NET
node JS
Microservices Support
Spring Boot
Spring Cloud
Services Marketplace
Native
User Provided
Partner
App Deployment & Management
CI/CD Tools, ID, Security
Availability
Health, Metrics, Patching
Visibility & Administration
Apps & Platform Dashboards
Operating System
Container Orchestration
Cloud API
Google
AWS
Azure
VMW
Openstack

Pivotal

Container Scheduler Handles Workloads

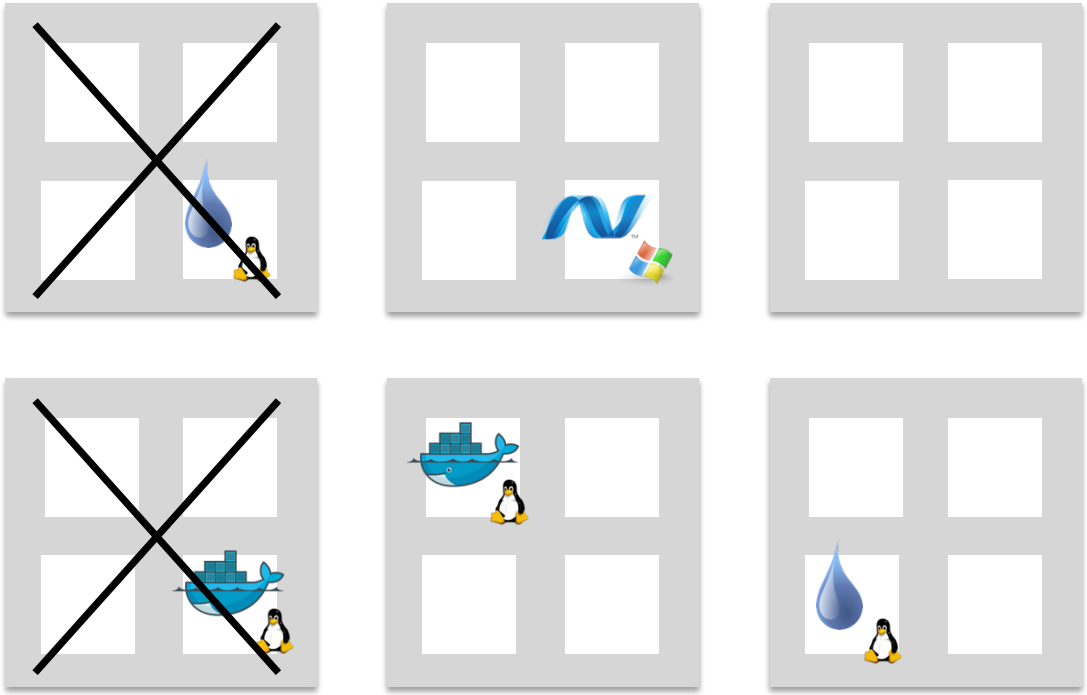


Container Scheduler Handles Workloads



Dynamic load balancing

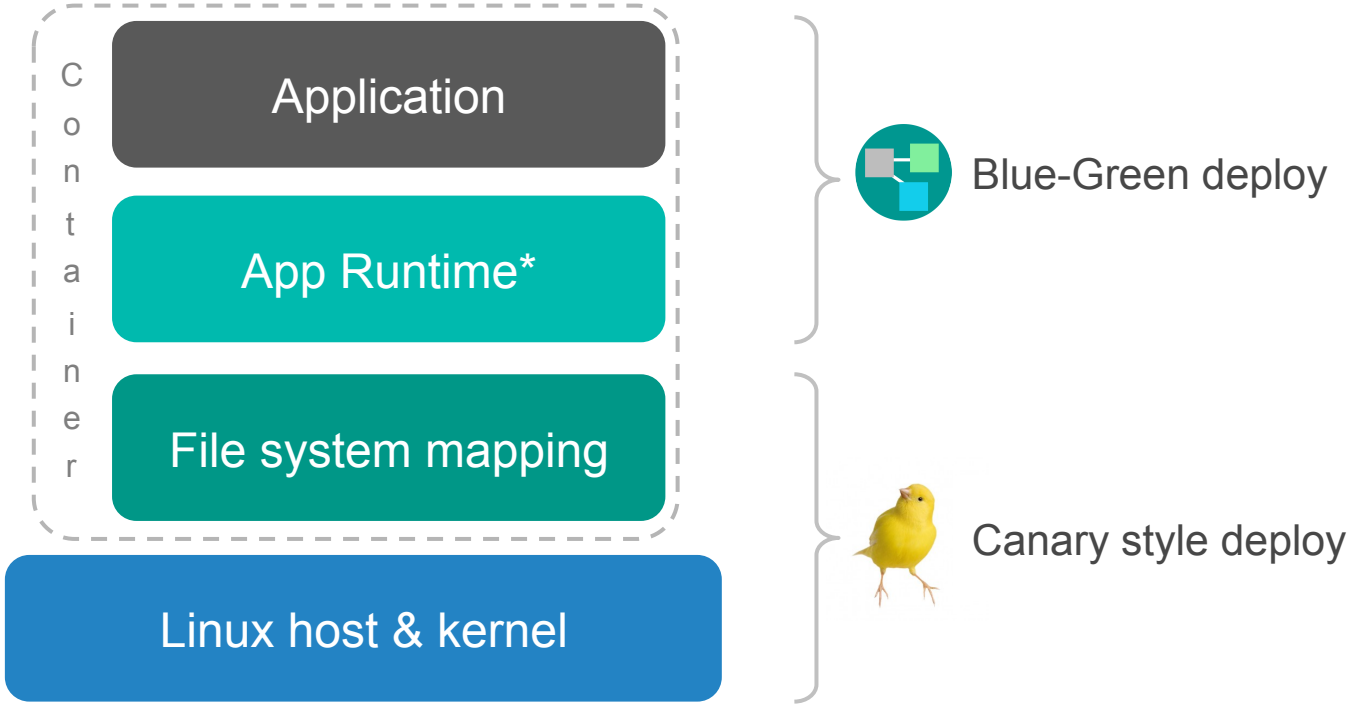
Container Scheduler Handles Workloads



Dynamic load balancing

Remediation and rebalance of workloads

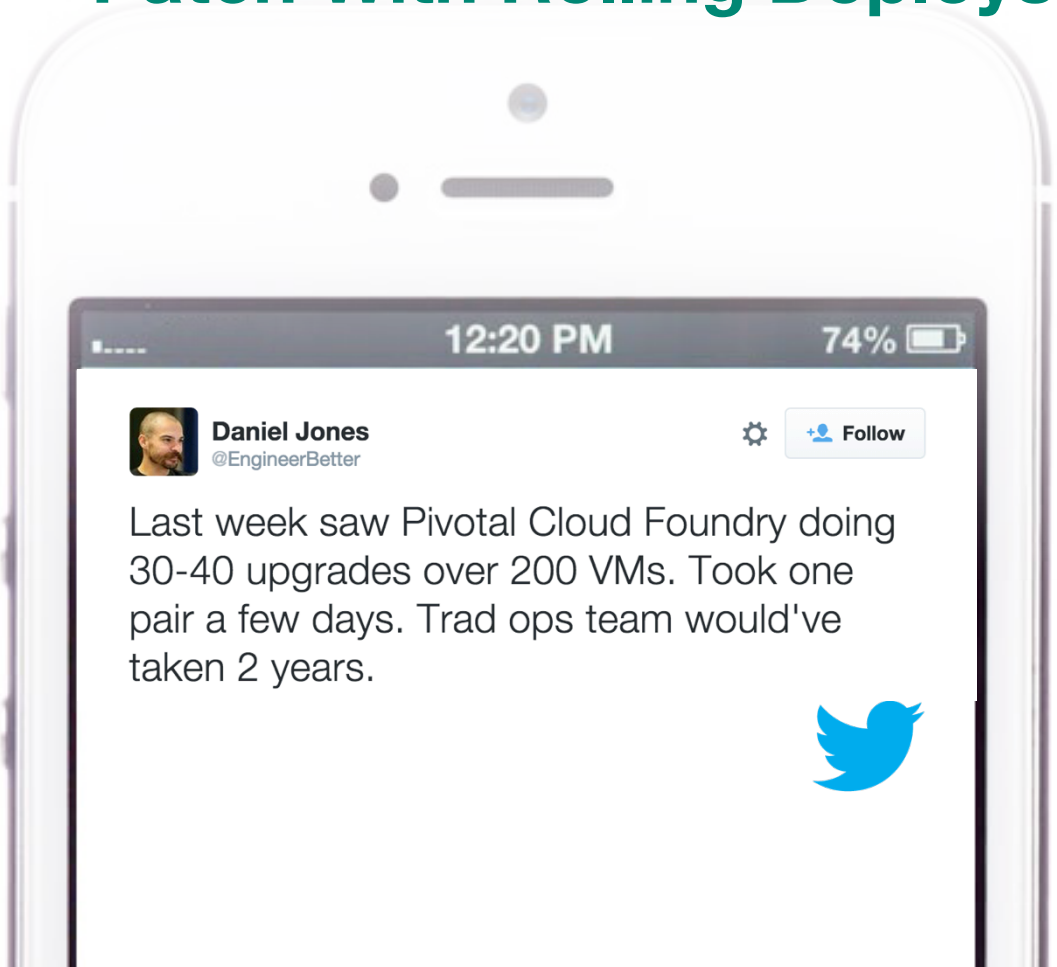
Each Layer Upgradable with No Downtime



* e.g. Embedded webserver, app configurations, JRE, agents for services packaged as buildpacks

Pivotal

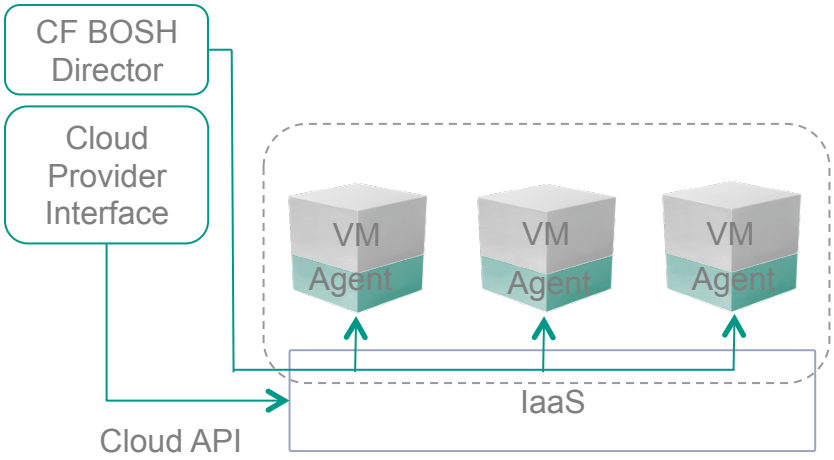
Patch With Rolling Deploys



Pivotal

“Cloud is about **how computing is done, not where**”

Paul Maritz , Founding CEO



Multi-cloud Advantage:
Cloud agnostic view of app & platform operations

Pivotal

Q&A

Pivotal®


Pivotal®
Transforming How The World Builds Software