Pivotal Cloud Foundry Overview

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Course Objectives

- To understand Pivotal Cloud Foundry and how to use it.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

What is Cloud Computing?

- **Cloud Computing is a general term used to describe a new class of network based computing that takes place over the Internet,**
  - basically a step on from Utility Computing
  - a collection/group of integrated and networked hardware, software and Internet infrastructure (called a platform).
  - Using the Internet for communication and transport provides hardware, software and networking services to clients
- **These platforms hide the complexity and details of the underlying infrastructure from users and applications by providing very simple graphical interface or API (Applications Programming Interface).**

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

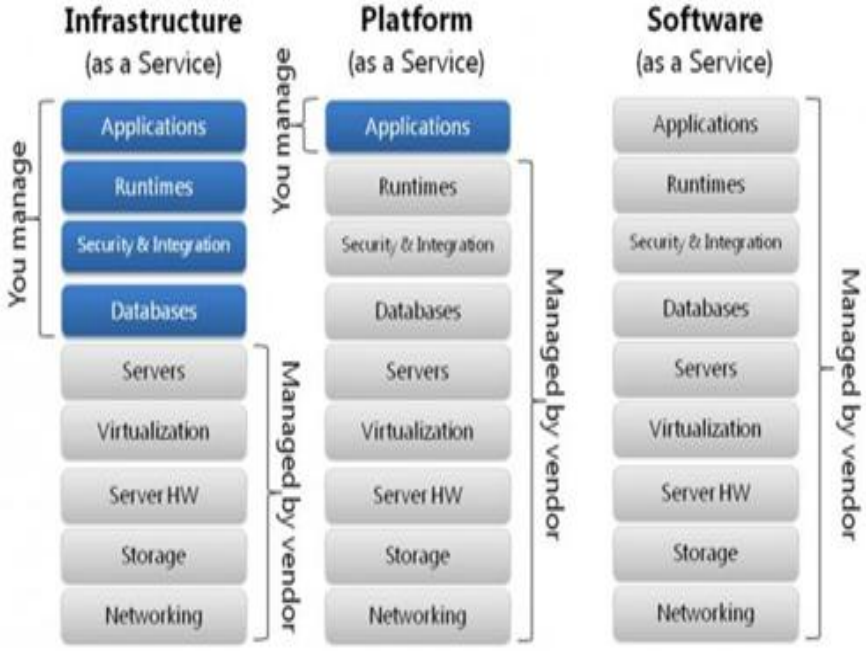
What is Cloud Computing?

- **In addition, the platform provides on demand services, that are always on, anywhere, anytime and any place.**
- **Pay for use and as needed, elastic**
  - scale up and down in capacity and functionalities
- **The hardware and software services are available to**
  - general public, enterprises, corporations and businesses markets

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Cloud Computing Service Models

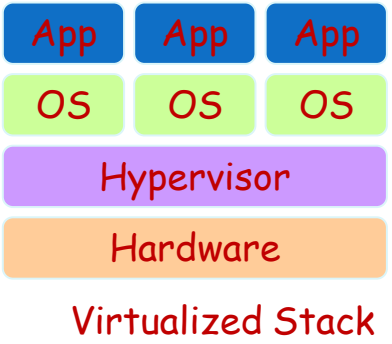
- **Infrastructure as a Service (IaaS), which provides only a base infrastructure, leaving the end user responsible for platform and environment configuration necessary to deploy applications. Amazon's AWS and Microsoft Azure are prime examples of IaaS.**
- **Software as a Service (SaaS) like Gmail or Salesforce.com.**
- **Platform as a Service (PaaS), which helps to reduce the development overhead (environment configuration) by providing a ready-to-use platform. PaaS services can be hosted on top of infrastructure provided by an IaaS.**



Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Virtualization

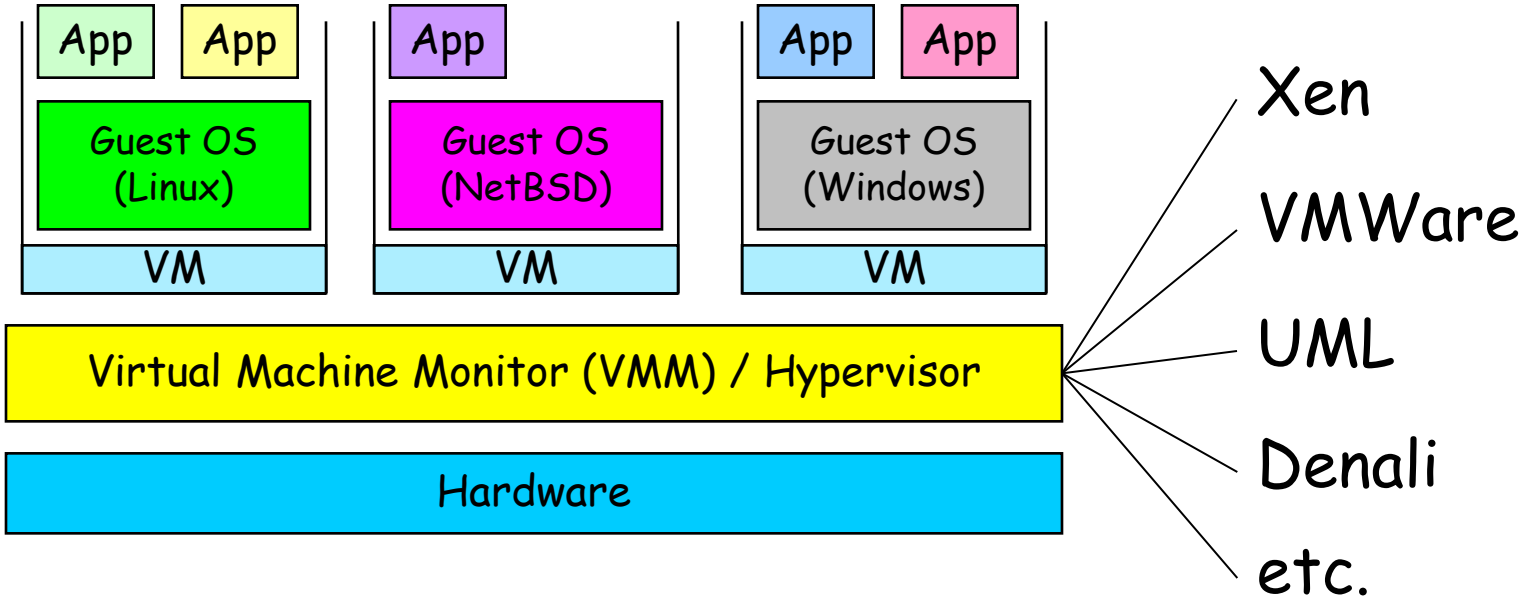
- **Virtual workspaces:**
  - An abstraction of an execution environment that can be made dynamically available to authorized clients by using well-defined protocols,
  - Resource quota (e.g. CPU, memory share),
  - Software configuration (e.g. O/S, provided services).
- **Implement on Virtual Machines (VMs):**
  - Abstraction of a physical host machine,
  - Hypervisor intercepts and emulates instructions from VMs, and allows management of VMs,
  - VMWare, Xen, etc.
- **Provide infrastructure API:**
  - Plug-ins to hardware/support structures



Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Virtual Machines

- VM technology allows multiple virtual machines to run on a single physical machine.



Performance: Para-virtualization (e.g. Xen) is very close to raw physical performance!

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Advantages of Cloud Computing

- **Lower computer costs:**
  - You do not need a high-powered and high-priced computer to run cloud computing's web-based applications.
  - Since applications run in the cloud, not on the desktop PC, your desktop PC does not need the processing power or hard disk space demanded by traditional desktop software.
  - When you are using web-based applications, your PC can be less expensive, with a smaller hard disk, less memory, more efficient processor...
  - In fact, your PC in this scenario does not even need a CD or DVD drive, as no software programs have to be loaded and no document files need to be saved.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Advantages of Cloud Computing

- **Improved performance:**
  - With few large programs hogging your computer's memory, you will see better performance from your PC.
  - Computers in a cloud computing system boot and run faster because they have fewer programs and processes loaded into memory…
- **Reduced software costs:**
  - Instead of purchasing expensive software applications, you can get most of what you need for free-ish!
    - most cloud computing applications today, such as the Google Docs suite.
  - better than paying for similar commercial software
    - which alone may be justification for switching to cloud applications.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Advantages of Cloud Computing

- **Instant software updates:**
  - Another advantage to cloud computing is that you are no longer faced with choosing between obsolete software and high upgrade costs.
  - When the application is web-based, updates happen automatically
    - available the next time you log into the cloud.
  - When you access a web-based application, you get the latest version
    - without needing to pay for or download an upgrade.

- **Improved document format compatibility.**
  - You do not have to worry about the documents you create on your machine being compatible with other users' applications or OSes
  - There are potentially no format incompatibilities when everyone is sharing documents and applications in the cloud.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Advantages of Cloud Computing

- **Unlimited storage capacity:**
  - Cloud computing offers virtually limitless storage.
  - Your computer's current 1 Tbyte hard drive is small compared to the hundreds of Pbytes available in the cloud.
- **Increased data reliability:**
  - Unlike desktop computing, in which if a hard disk crashes and destroy all your valuable data, a computer crashing in the cloud should not affect the storage of your data.
    - if your personal computer crashes, all your data is still out there in the cloud, still accessible
  - In a world where few individual desktop PC users back up their data on a regular basis, cloud computing is a data-safe computing platform!

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Advantages of Cloud Computing

- **Universal document access:**
  - That is not a problem with cloud computing, because you do not take your documents with you.
  - Instead, they stay in the cloud, and you can access them whenever you have a computer and an Internet connection
  - Documents are instantly available from wherever you are
- **Latest version availability:**
  - When you edit a document at home, that edited version is what you see when you access the document at work.
  - The cloud always hosts the latest version of your documents
    - as long as you are connected, you are not in danger of having an outdated version

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Advantages of Cloud Computing

- **Easier group collaboration:**
  - Sharing documents leads directly to better collaboration.
  - Many users do this as it is an important advantages of cloud computing
    - multiple users can collaborate easily on documents and projects
- **Device independence.**
  - You are no longer tethered to a single computer or network.
  - Changes to computers, applications and documents follow you through the cloud.
  - Move to a portable device, and your applications and documents are still available.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

What is Cloud Foundry and Concepts

- **Cloud Foundry is an open platform as a service (PaaS), providing a choice of clouds, developer frameworks, and application services.**
- **Cloud Foundry makes it faster and easier to build, test, deploy and scale applications.**
- **It is an open source project and is available through a variety of private cloud distributions and public cloud instances.**
- **Cloud Foundry is available as a stand-alone software package. You can, of course, deploy it on Amazon's AWS, but you can also host it yourself on your own OpenStack server, or through HP's Helion or VMware's vSphere.**
- **Cloud Foundry is an open source cloud computing platform originally developed in-house at VMware. It is now owned by Pivotal Software, which is a joint venture made up of VMware, EMC, and General Electric.**

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

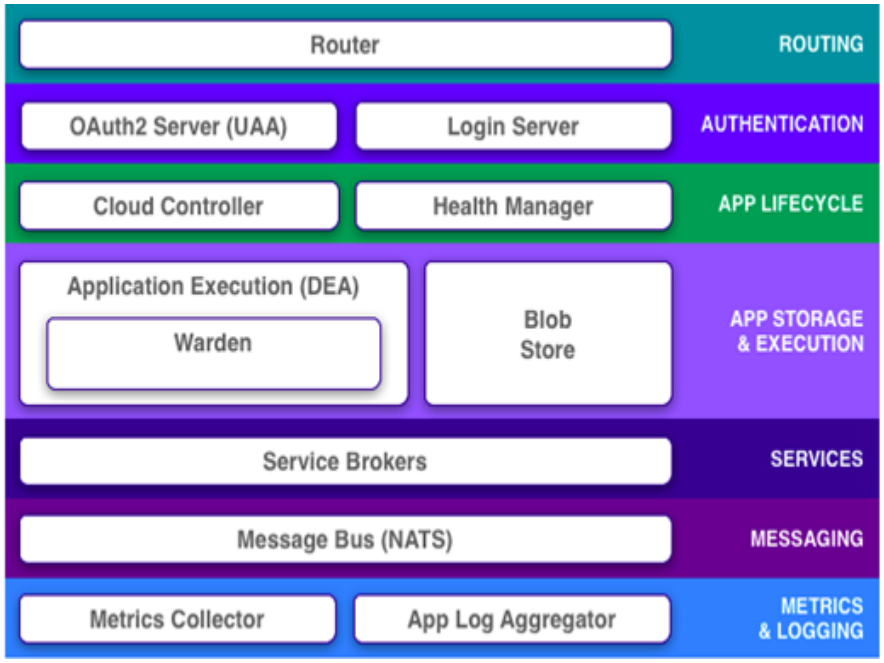
Key benefits of Cloud Foundry

- **Key benefits of Cloud Foundry:**
  - Application portability.
  - Application auto-scaling.
  - Centralized platform administration.
  - Centralized logging.
  - Dynamic routing.
  - Application health management.
  - Integration with external logging components like Elasticsearch and Logstash.
  - Role based access for deployed applications.
  - Provision for vertical and horizontal scaling.
  - Infrastructure security.
  - Support for various IaaS providers.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

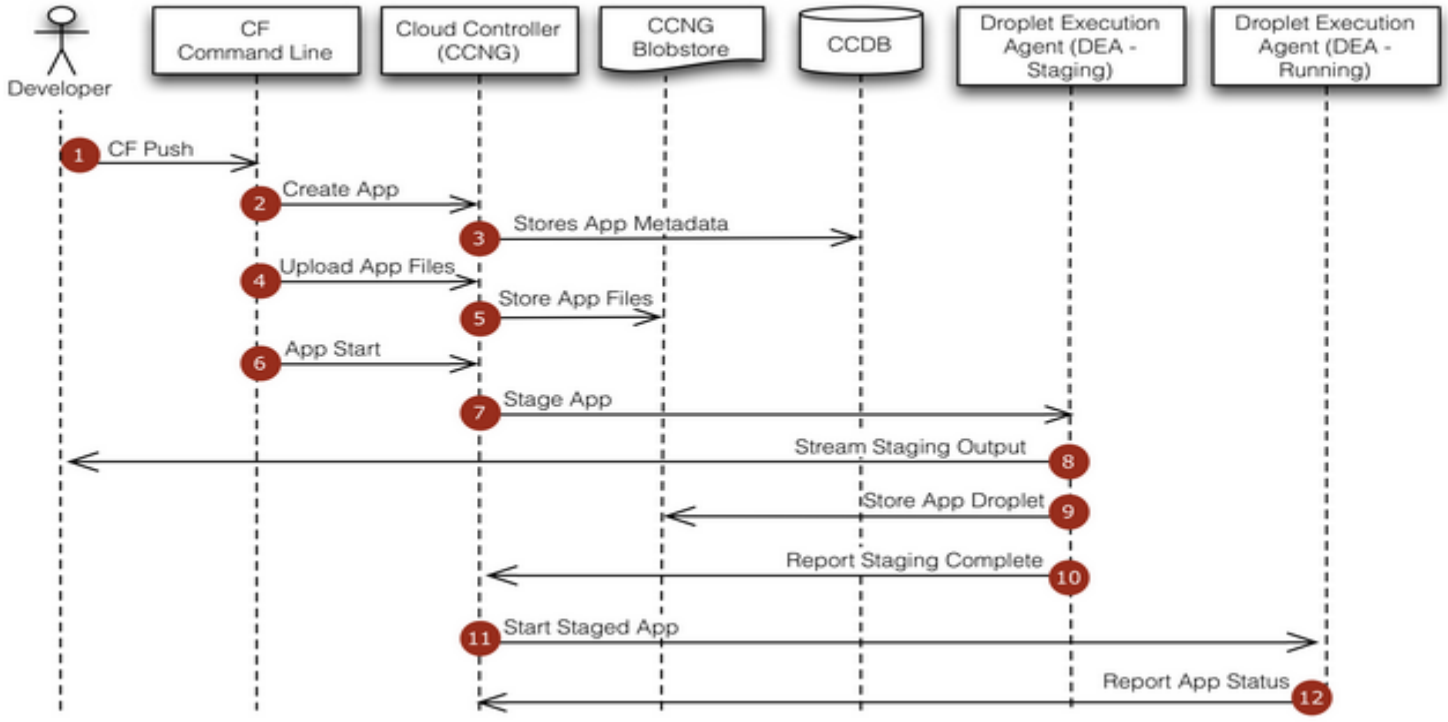
Cloud Foundry: understanding the core components

- Cloud Foundry components include a self-service application execution engine, an automation engine for application deployment and lifecycle management, and a scriptable command line interface (CLI), as well as integration with development tools to ease deployment processes. Cloud Foundry has an open architecture that includes a buildpack mechanism for adding frameworks, an application services interface, and a cloud provider interface.



Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Cloud Foundry: understanding the core components



Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Orgs, Spaces, Roles, and Permissions

- **CF uses a role-based access control (RBAC) system to grant Cloud Foundry users permissions appropriate to their role within an org or a space. Admins, Org Managers, and Space Managers can assign user roles using the cf CLI.**
- **Orgs :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.**
- **User Accounts :A user account represents an individual person within the context of a Cloud Foundry installation. A user can have different roles in different spaces within an org, governing what level and type of access they have within that space.**
- **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.**

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Orgs, Spaces, Roles, and Permissions

- **Roles and Permissions**
- **A user can have one or more roles. The combination of these roles defines the user's overall permissions in the org and within specific spaces in that org.**
- **Org Managers are managers or other users who need to administer the account.**
  - Note: An Org Manager needs explicit administrator permissions to perform certain actions. Org Auditors view but cannot edit user information and org quota usage information.
- **Org Billing Managers create and manage billing account and payment information.**
  - Note: The Billing Manager role is only relevant for Cloud Foundry environments deployed with a billing engine.
- **Space Managers are managers or other users who administer a space within an org.**
- **Space Developers are application developers or other users who manage applications and services in a space.**
- **Space Auditors view but cannot edit the space.**

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Considerations for Designing and Running an Application in the Cloud

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Considerations for Designing and Running an Application in the Cloud

- **Application Design for the Cloud**
  - Avoid Writing to the Local File System
  - Cookies Accessible across Applications
  - Port Limitations
  - Cloud Foundry Updates and Your Application
- **Ignore Unnecessary Files When Pushing**
- **Run Multiple Instances to Increase Availability**
- **Using Buildpacks**

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Application Design for the Cloud

- The Twelve Factors Apps guidelines

- I. Codebase : One codebase tracked in revision control, many deploys
- II. Dependencies : Explicitly declare and isolate dependencies
- III. Config :Store config in the environment
- IV. Backing services : Treat backing services as attached resources
- V. Build, release, run : Strictly separate build and run stages
- VI. Processes : Execute the app as one or more stateless processes
- VII. Port binding : Export services via port binding
- VIII. Concurrency : Scale out via the process model
- IX. Disposability : Maximize robustness with fast startup and graceful shutdown
- X. Dev/prod parity : Keep development, staging, and production as similar as possible
- XI. Logs : Treat logs as event streams
- XII. Admin processes : Run admin/management tasks as one-off processes

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Understanding Application Deployment

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Deploy an Application

- Overview of Deployment Process
- Step 1: Prepare to Deploy
- Step 2: Know Your Credentials and Target
- Step 3: (Optional) Configure Domains
- Step 4: Determine Deployment Options
- Define Deployment Options
- Set Environment Variables
- Configure Runtime Hooks (available in Diego cf-release v238)
- Step 5: Push the Application
- Step 6: (Optional) Configure Service Connections
- Step 7: Troubleshoot Deployment Problems

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Overview of Deployment Process

- You deploy an application to Cloud Foundry by running a push command from a Cloud Foundry command line interface (CLI).

- Uploads and stores application files
- Examines and stores application metadata
- Creates a “droplet” (the Cloud Foundry unit of execution) for the application
- Selects an appropriate droplet execution agent (DEA) to run the droplet
- Starts the application

- An application that uses services, such as a database, messaging, or email server, is not fully functional until you provision the service and, if required, bind the service to the application. (More on services will be covered in later)

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Step 1: Prepare to Deploy

- **Before you deploy your application to Cloud Foundry, make sure that:**
  - Your application is cloud-ready. Cloud Foundry behaviors related to file storage, HTTP sessions, and port usage may require modifications to your application.
  - All required application resources are uploaded. For example, you may need to include a database driver.
  - Extraneous files and artifacts are excluded from upload. You should explicitly exclude extraneous files that reside within your application directory structure, particularly if your application is large.
  - An instance of every service that your application needs has been created.
  - Your Cloud Foundry instance supports the type of application you are going to deploy, or you have the URL of an externally available buildpack that can stage the application.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Step 2: Know Your Credentials and Target

- **Before you can push your application to Cloud Foundry you need to know:**
  - The API endpoint for your Cloud Foundry instance. Also known as the target URL, this is the URL of the Cloud Controller in your Elastic Runtime instance.
  - Your username and password for your Cloud Foundry instance.
  - The organization and space where you want to deploy your application. A Cloud Foundry workspace is organized into organizations, and within them, spaces. As a Cloud Foundry user, you have access to one or more organizations and spaces.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Step 3: (Optional) Configure Domains

- **Cloud Foundry directs requests to an application using a route, which is a URL made up of a host and a domain.**
  - The name of an application is the default host for that application, unless you specify the host name with the -n flag.
  - Every application is deployed to an application space that belongs to a domain. Every Cloud Foundry instance has a default domain defined. You can specify a non-default, or custom, domain when deploying, provided that the domain is registered and is mapped to the organization which contains the target application space.
  - Note: CF allows app names, but not app URLs, to include underscores. CF converts underscores to hyphens when setting a default app URL from an app name.

- **The URL for your app must be unique from other apps hosted by Elastic Runtime. Use the following options with the cf CLI to help create a unique URL:**
  - -n to assign a different HOST name for the app
  - --random-route to create a URL that includes the app name and random words
- **Note: Use cf help push to view other options for this command.**

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Step 4: Determine Deployment Options

- **Before you deploy, you need to decide on the following:**

  - Name: the name of your application.
  - Instances: Generally speaking, the more instances you run, the less downtime your application will experience. If your application is still in development, running a single instance can simplify troubleshooting. For any production application, we recommend a minimum of two instances.
  - Memory Limit: The maximum amount of memory that each instance of your application can consume. If an instance exceeds this limit, Cloud Foundry restarts the instance.
  - Note: Initially, Cloud Foundry immediately restarts any instances that exceed the memory limit. If an instance repeatedly exceeds the memory limit in a short period of time, Cloud Foundry delays restarting the instance.
  - Start Command: This is the command that Cloud Foundry uses to start each instance of your application. This start command varies by application framework.
  - Subdomain (host) and Domain: The route, which is the combination of subdomain and domain, must be globally unique. This is true whether you specify a portion of the route or allow Cloud Foundry to use defaults.
  - Services: Applications can bind to services such as databases, messaging, and key-value stores. Applications are deployed into application spaces. An application can only bind to a service that has an existing instance in the target application space.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Define Deployment Options

- You can define deployment options on the command line, in a manifest file, or both together.
- When you deploy an application while it is running, Cloud Foundry stops all instances of that application and then deploys. Users who try to run the application get a “404 not found” message while cf push runs. Stopping all instances is necessary to prevent two versions of your code from running at the same time.
- Cloud Foundry uploads all application files except version control files with file extensions .svn, .git, and .darcs. To exclude other files from upload, specify them in a .cfignore file in the directory where you run the push command. This technique is similar to using a .gitignore file.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Set Environment Variables

- Environment variables are key-value pairs defined at the operating system level.
- These key-value pairs provide a way to configure the applications running on a system.
- You can set environment variables for an application in one of three ways:

1. Using the cf CLI. See the following example:
   - $ cf set-env APP-NAME ENV-VAR-NAME ENV-VAR-VALUE
   - Setting env variable 'ENV-VAR-NAME' to 'ENV-VAR-VALUE' for app spring-music in org example / space development as user@example.com...
   - OK
- TIP: Use 'cf restage' to ensure your env variable changes take effect

2. In an application manifest.
3. In a .profile file. (more in Configure Runtime Hooks section)

- Cloud Foundry sources environment variables in the following order: the application manifest, then the cf CLI, and finally the .profile file. For example, an environment variable set in the .profile file overwrites an environment variable set in the application manifest and the cf CLI.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Configure Runtime Hooks

- To execute runtime hooks, insert a .profile file into the root of your application directory. This file should contain bash script that performs application-specific initialization tasks.

- When an application container starts, a bash shell executes the .profile file after sourcing the application's environment variables from the application manifest and the cf CLI. Because the .profile script executes after the buildpack, the script also has access to the language runtime environment created by the buildpack.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Step 5: Push the Application

- Run the following command to deploy an application without a manifest:

  cf push APP-NAME

- If you provide the application name in a manifest, you can reduce the command to cf push. Since all you have provided is the name of your application, cf push sets the number of instances, amount of memory, and other attributes of your application to the default values. You can also use command-line options to specify these and additional attributes.

- **Note: When deploying your own apps, avoid generic names like my-app. Cloud Foundry uses the app name to compose the route to the app, and deployment fails unless the app has a globally unique route.**

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Step 5: Push the Application

- The cf push command pushes a new app or syncs changes to an existing app.
- USAGE
- Push a single app (with or without a manifest):
  - cf push APP_NAME [-b BUILDPACK_NAME] [-c COMMAND] [-d DOMAIN] [-f MANIFEST_PATH] [--docker-image DOCKER_IMAGE] [-i NUM_INSTANCES] [-k DISK] [-m MEMORY] [--hostname HOST] [-p PATH] [-s STACK] [-t TIMEOUT] [-u HEALTH_CHECK_TYPE] [--route-path ROUTE_PATH] [--no-hostname] [--no-manifest] [--no-route] [--no-start] [--random-route]
- Push multiple apps with a manifest:
  - cf push [-f MANIFEST_PATH]
- ALIAS
  - p

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Step 5: Push the Application

- Options:

  -b : Custom buildpack by name (e.g. my-buildpack) or Git URL (e.g. https://github.com/cloudfoundry/java-buildpack.git) . To use built-in buildpacks only, specify default or null

  -c : Startup command, set to null to reset to default start command

  -d : Domain (e.g. example.com)

  --docker-image, -o : Docker-image to be used (e.g. user/docker-image-name)

  -f : Path to manifest

  --health-check-type, -u : Application health check type (e.g. port or none)

  --hostname, -n : Hostname (e.g. my-subdomain)

  -i : Number of instances

  -k : Disk limit (e.g. 256M, 1024M, 1G)

  -m : Memory limit (e.g. 256M, 1024M, 1G)

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Step 5: Push the Application

--no-hostname : Map the root domain to this app

--no-manifest: Ignore manifest file

--no-route : Do not map a route to this app and remove routes from previous pushes of this app

--no-start : Do not start an app after pushing

-p : Path to app directory or to a zip file of the contents of the app directory

--random-route: Create a random route for this app

--route-path : Path for the route

-s : Stack to use (a stack is a pre-built file system, including an operating system, that can run apps)

-t :Maximum time (in seconds) for CLI to wait for application start, other server side timeouts may apply

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Step 6: (Optional) Configure Service Connections

- If you bound a service to the application that you deployed, you might need to configure your application with the service URL and credentials. (more on this in services session)

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Step 7: Troubleshoot Deployment Problems

- **You can troubleshoot your application in the cloud using the cf CLI. (more info on this in Troubleshooting Application Deployment and Health section)**

- The application deployment process involves uploading, staging, and starting the app in a container. Your app must successfully complete each of these phases within certain time limits. The default time limits for the phases are as follows:
  - Upload: 15 minutes
  - Stage: 15 minutes
  - Start: 60 seconds
- **Note**: Your administrator can change these defaults. Check with your administrator for the actual time limits set for app deployment.
- Developers can change the time limit for starting apps through an application manifest or on the command line.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Buildpacks

- Buildpacks provide framework and runtime support for your applications. Buildpacks typically examine user-provided artifacts to determine what dependencies to download and how to configure applications to communicate with bound services.

- When you push an application, Cloud Foundry automatically detects which buildpack is required and installs it on the Droplet Execution Agent (DEA) where the application needs to run.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Buildpacks

System Build packs

Name	Supported Languages, Frameworks, and Technologies
Java	Grails, Play, Spring, or any other JVM-based language or framework
Ruby	Ruby, JRuby, Rack, Rails, or Sinatra
Node.js	Node or JavaScript
Binary	*n/a*
Go	*Go*
PHP	Cake, Symfony, Zend, Nginx, or HTTPD
Python	Django or Flask
Staticfile	HTML, CSS, JavaScript, or Nginx

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Buildpacks

- **Custom Buildpacks**

- **If your application uses a language or framework that the Cloud Foundry system buildpacks do not support, do the following:**

  - Use a Cloud Foundry Community Buildpack.
  - Use a Heroku Third-Party Buildpack.
  - Customize an existing buildpack or create your own custom buildpack. A common development practice for custom buildpacks is to fork existing buildpacks and sync subsequent patches from upstream.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Routes and Domains

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Routes and Domains

- **Routes**
  - The Elastic Runtime router routes requests to applications by associating an app with an address, known as a route. This association is called a mapping: for example, the cf CLI command for associating an app and route is cf map-route.
- **Domains**
  - Domains indicate to a developer that requests for any route created from the domain will be routed to Elastic Runtime. This requires DNS to be configured out-of-band to resolve the domain name to the IP address of a load balancer configured to forward requests to the CF routers.

  - List Domains for an Org

When creating a route, developers will select from domains available to them. Use the cf domains command to view a list of available domains for the targeted org:

```
$ cf domains
Getting domains in org my-org as user@example.com... OK
name                    status   type
shared-domain.example.com       shared
tcp.shared-domain.example.com   shared   tcp
private-domain.example.com      owned
```


Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Routes and Domains

- **Create an HTTP Route with Hostname**
  - In Elastic Runtime, a hostname is the label that indicates a subdomain of the domain associated with the route. Given a domain shared-domain.example.com, a developer can create the route myapp.shared-domain.example.com in space my-space by specifying the hostname myapp with the cf create-route command:

```
$ cf create-route my-space shared-domain.example.com --hostname myapp
Creating route myapp.shared-domain.example.com for org my-org / space my-space as username@example.com...
OK
```

- **This command instructs Elastic Runtime to only route requests to apps mapped to this route for the following URLs:**
  - http://myapp.shared-domain.example.com
  - https://myapp.shared-domain.example.com
  - Any path under either of the above URLs, such as http://myapp.shared-domain.example.com/bar

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Routes and Domains

- **Create an HTTP Route with a Path**
- Developers can use paths to route requests for the same hostname and domain to different apps.

```
$ cf create-route my-space shared-domain.example.com --hostname store --path
products
Creating route store.shared-domain.example.com/products for org my-org / space my-
space as username@example.com...
OK
```


Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Routes and Domains

- **Map a Route to your Application**
- **For an app to receive requests to a route, developers must map the route to the app with the cf map-route command. If the route does not already exist, this command creates it.**
- For example –

```
$ cf map-route products shared-domain.example.com --hostname store --path products
```

Route	Application
store.shared-domain.example.com/products	products

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Routes and Domains

- **Map a Route with Application Push**
  - Developers can map a route to their app with the cf push command.

```
$ cf push myapp -d private-domain.example.com --hostname foo
```

- **Map a Route Using Application Manifest**
  - Developers can map a route to their app with a manifest by editing the route attribute to specify the host, domain, port and/or path components of the route.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Routes and Domains

- **List Routes**
- **Developers can list routes for the current space with the cf routes command. A route is uniquely identified by the combination of hostname, domain, port, and path.**

```
$ cf routes
Getting routes as user@private-domain.example.com ...
space     host      domain                    port  path       type  apps
my-space  myapp     shared-domain.example.com                      myapp
my-space  myapp     private-domain.example.com                     myapp
my-space  store     shared-domain.example.com       /products      products
my-space  store     shared-domain.example.com       /orders         orders
my-space  store     shared-domain.example.com                      storefront
```

- **Developers can only see routes in spaces where they are members.**

Capgemini CONSULTING.TECHNOLOGY.OUTSOURCING

Routes and Domains

Unmap a Route

- Developers can remove a route from an app using the cf unmap-route command. The route remains reserved for later use in the space where it was created until the route is deleted.
- To unmap an HTTP route from an app, identify the route using the hostname, domain, and path:

```
$ cf unmap-route tcp-app private-domain.example.com --hostname myapp --path mypath
```

Delete a Route

- Developers can delete a route from a space using the cf delete-route command.
- To delete a HTTP route, identify the route using the hostname, domain, and path:

```
$ cf delete-route private-domain.example.com --hostname myapp --path mypath
```


Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Stacks and Changing Stacks

- A stack is a prebuilt root filesystem (rootfs) that supports a specific operating system. For example, Linux-based systems need **/usr** and **/bin** directories at their root and Windows needs **\windows**. The stack works in tandem with a buildpack to support applications running in compartments.

- **Available Stacks**
  - The Linux cflinuxfs2 stack is derived from Ubuntu Trusty 14.04.
  - The Windows stack windows2012R2 supports .NET apps.

- **Use the cf stacks command to list the stacks available in a deployment.**

```
$ cf stacks
Getting stacks in org MY-ORG / space development as developer@example.com...
OK

name            description
cflinuxfs2      Cloud Foundry Linux-based filesystem
windows2012R2   Windows Server 2012 R2
```

- **To specify a different stack, append -s STACKNAME to the push command. For example, you can ensure that Windows application MY-WIN-APP deploys to a Windows-based cell by running cf push MY-WIN-APP -s windows2012R2.**

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Deploying with Application Manifests

- Application manifests tell cf push what to do with applications. This includes everything from how many instances to create and how much memory to allocate to what services applications should use.

- A manifest can help you automate deployment, especially of multiple applications at once.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Deploying with Application Manifests

- Application manifests tell cf push what to do with applications. This includes everything from how many instances to create and how much memory to allocate to what services applications should use.

- A manifest can help you automate deployment, especially of multiple applications at once.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Deploying with Application Manifests

- **By default, the cf push command deploys an application using a manifest.yml file in the current working directory.**

```
$ cf push
Using manifest file /path_to_working_directory/manifest.yml
```

- **If your manifest is located elsewhere, use the -f option to provide the path to the filename.**

```
$ cf push -f ./some_directory/some_other_directory/alternate_manifest.yml
Using manifest file
/path_to_working_directory/some_directory/some_other_directory/alternate_manifest.yml
```

- **If you provide a path with no filename, the filename must be manifest.yml.**

```
$ cf push -f ./some_directory/some_other_directory/
Using manifest file
/path_to_working_directory/some_directory/some_other_directory/manifest.yml
```


Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Deploying with Application Manifests

- **Manifests are written in YAML. The manifest below illustrates some YAML conventions, as follows:**
  - The manifest may begin with three dashes.
  - The applications block begins with a heading followed by a colon.
  - The application name is preceded by a single dash and one space.
  - Subsequent lines in the block are indented two spaces to align with name.

```
---
applications:
- name: nifty-gui
  memory: 512M
  host: nifty
```

- **A minimal manifest requires only an application name.**
- **Manifest is not mandatory. You can specify application name on cf push command line.**

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Deploying with Application Manifests

- **Always Provide an Application Name to cf push**
  - cf push requires an application name, which you provide either in a manifest or at the command line.
  - The command cf push locates the manifest.yml in the current working directory by default, or in the path provided by the -f option.

- **If you do not use a manifest, the minimal push command looks like this:**

```
$ cf push my-app
```

- **Note: When you provide an application name at the command line, cf push uses that application name whether or not there is a different application name in the manifest. If the manifest describes multiple applications, you can push a single application by providing its name at the command line; the cf CLI does not push the others. Use these behaviors for testing.**

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Deploying with Application Manifests

- **Precedence Between Manifests, Command Line Options, and Most Recent Values**

- **cf push follows rules of precedence when setting attribute values:**
  - Manifests override most recent values, including defaults.
  - Command line options override manifests.
  - In general, you can think of manifests as just another input to cf push, to be combined with command line options and most recent values.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Deploying with Application Manifests

- The host attribute
- This segment of a route helps to ensure that the route is unique. If you do not provide a hostname, the URL for the app takes the form of APP-NAME.DOMAIN.

```
---
...
host: my-app
```

- The command line option that overrides this attribute is -n.

- The hosts attribute
- Each hostname generates a unique route for the app. hosts can be used in conjunction with host. If you define both attributes, Cloud Foundry creates routes for hostnames defined in both host and hosts.

```
---
...
hosts:
- app_host1
- app_host2
```

- The command line option that overrides this attribute is -n.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Deploying with Application Manifests

- The no-hostname attribute
- By default, if you do not provide a hostname, the URL for the app takes the form of APP-NAME.DOMAIN. If you want to override this and map the root domain to this app then you can set no-hostname as true.

```
---
...
no-hostname: true
```

- The command line option that corresponds to this attribute is --no-hostname.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Scaling an Application Using cf scale

- **What is scaling**
  - Factors such as user load, or the number and nature of tasks performed by an application, can change the disk space and memory the application uses.
  - For many applications, increasing the available disk space or memory can improve overall performance.
  - Similarly, running additional instances of an application can allow the application to handle increases in user load and concurrent requests.
  - These adjustments are called scaling an application.

- **Use cf scale to scale your application up or down to meet changes in traffic or demand.**

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Scaling an Application Using cf scale

- **Scaling Horizontally**
  - Horizontally scaling an application creates or destroys instances of your application.
  - Incoming requests to your application are automatically load balanced across all instances of your application, and each instance handles tasks in parallel with every other instance. Adding more instances allows your application to handle increased traffic and demand.
  - Use cf scale APP -i INSTANCES to horizontally scale your application. Cloud Foundry will increase or decrease the number of instances of your application to match INSTANCES.

```
$ cf scale myApp -i 5
```


Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Scaling an Application Using cf scale

- **Scaling Vertically**
  - Vertically scaling an application changes the disk space limit or memory limit that Cloud Foundry applies to all instances of the application.
  - Use cf scale APP -k DISK to change the disk space limit applied to all instances of your application. DISK must be an integer followed by either an M, for megabytes, or G, for gigabytes.

```
$ cf scale myApp -k 512M
```

  - Use cf scale APP -m MEMORY to change the memory limit applied to all instances of your application. MEMORY must be an integer followed by either an M, for megabytes, or G, for gigabytes.

```
$ cf scale myApp -m 1G
```


Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Cloud Foundry Environment Variables

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

- **View Environment Variables**
  - Use the cf env command to view the Cloud Foundry environment variables for your application.
  - The VCAP_APPLICATION and VCAP_SERVICES variables provided in the container environment
  - The user-provided variables set using the cf set-env command.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Cloud Foundry Environment Variables

- **Some of the important environment variables:**
  - CF_INSTANCE_ADDR:The CF_INSTANCE_IP and CF_INSTANCE_PORT of the app instance in the format IP:PORT.
  - CF_INSTANCE_GUID:The GUID of the application.
  - CF_INSTANCE_INDEX:The index number of the app instance.
  - CF_INSTANCE_IP:The external IP address of the host running the app instance.
  - CF_INSTANCE_PORT: The external (host-side) port corresponding to the internal (container-side) port with value PORT.
  - HOME:Root folder for the deployed application.
  - MEMORY_LIMIT: The maximum amount of memory that each instance of the application can consume. You specify this value in an application manifest or with the cf CLI when pushing an application. The value is limited by space and org quotas.
  - PORT: The port on which the application should listen for requests.
  - PWD: Identifies the present working directory, where the buildpack that processed the application ran.
  - TMPDIR: Directory location where temporary and staging files are stored.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Cloud Foundry Environment Variables

- **Some of the important environment variables:**
  - USER: The user account under which the application runs.
  - VCAP_APPLICATION: his variable contains the associated attributes for a deployed application. Results are returned in JSON format.

Example :

VCAP_APPLICATION={"instance_id":"fe98dc76ba549876543210abcd1234",
"instance_index":0,"host":"0.0.0.0","port":61857,"started_at":"2013-08-12
00:05:29 +0000","started_at_timestamp":1376265929,"start":"2013-08-12 00:05:29
+0000","state_timestamp":1376265929,"limits":{"mem":512,"disk":1024,"fds":16384}
,"application_version":"ab12cd34-5678-abcd-0123-abcdef987654","application_name"
:"styx-james","application_uris":["styx-james.a1-app.cf-app.com"],"version":"ab1
2cd34-5678-abcd-0123-abcdef987654","name":"my-app","uris":["my-
app.example.com"]
,"users":null}

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Cloud Foundry Environment Variables

- **VCAP_SERVICES:**
  - For bindable services Cloud Foundry will add connection details to the VCAP_SERVICES environment variable when you restart your application, after binding a service instance to your application.
  - The results are returned as a JSON document that contains an object for each service for which one or more instances are bound to the application. The service object contains a child object for each service instance of that service that is bound to the application. The attributes that describe a bound service are defined in the table below.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Cloud Foundry Environment Variables

- **VCAP_SERVICES example :**

```
VCAP_SERVICES=
{
 "elephantsql": [
  {
   "name": "elephantsql-c6c60",
   "label": "elephantsql",
   "tags": [
    "postgres",
    "postgresql",
    "relational"
   ],
   "plan": "turtle",
   "credentials": {
    "uri": "postgres://exampleuser:examplepass@babar.elephantsql.com:5432/exampleuser"
   }
  }
 ]
}
```


Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Using Blue-Green Deployment to Reduce Downtime and Risk

- Blue-green deployment is a release technique that reduces downtime and risk by running two identical production environments called Blue and Green.
- At any time, only one of the environments is live, with the live environment serving all production traffic. For this example, Blue is currently live and Green is idle.
- As you prepare a new release of your software, deployment and the final stage of testing takes place in the environment that is *not* live: in this example, Green.
- Once you have deployed and fully tested the software in Green, you switch the router so all incoming requests now go to Green instead of Blue. Green is now live, and Blue is idle.
- This technique can eliminate downtime due to application deployment. In addition, blue-green deployment reduces risk: if something unexpected happens with your new release on Green, you can immediately roll back to the last version by switching back to Blue.

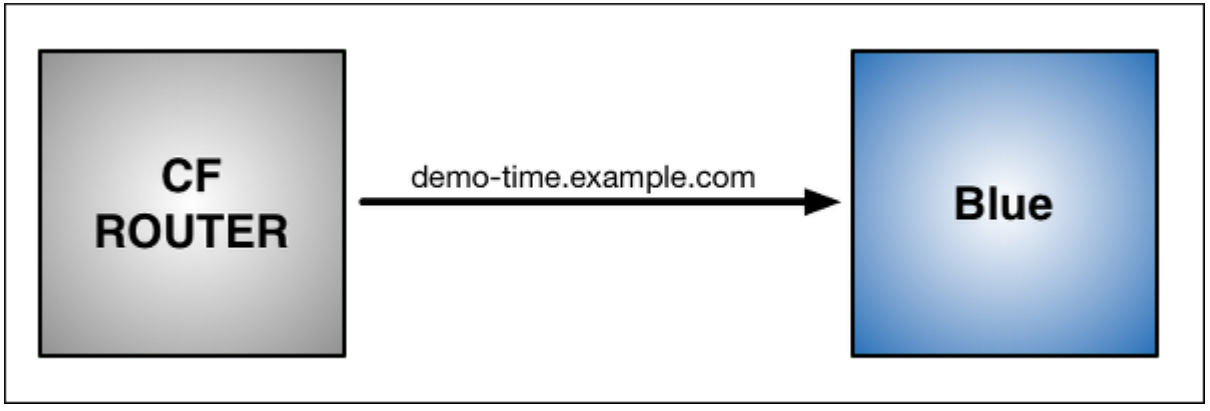
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Using Blue-Green Deployment to Reduce Downtime and Risk

- **Blue-Green Deployment with Cloud Foundry Example Steps**
- **Step 1: Push an App**
  - Use the cf CLI to push the application. Name the application "Blue" with the subdomain "demo-time."

```
$ cf push Blue -n demo-time
```

  - Blue is now running on Cloud Foundry.
  - The CF Router sends all traffic for demo-time.example.com traffic to Blue.

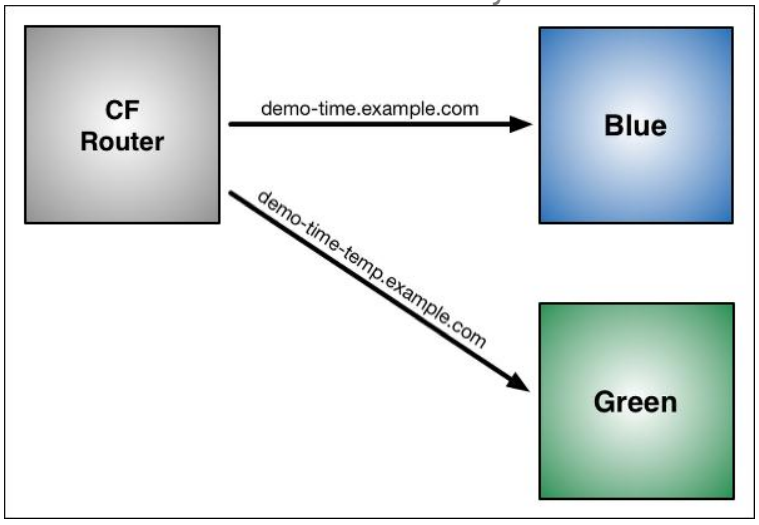


Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Using Blue-Green Deployment to Reduce Downtime and Risk

- **Step 2: Update App and Push**
  - Now make a change to the application. First, replace the word "Blue" on the web page with "Green," then rebuild the source file for the application.
  - Run cf push again, but use the name "Green" for the application and provide a different subdomain to create a temporary route:

    $ cf push Green -n demo-time-temp
  - Two instances of our application are now running on Cloud Foundry: the original Blue and the updated Green.
  - The CF Router still sends all traffic for demo-time.example.com traffic to Blue. The router now also sends any traffic for demo-time-temp.example.com to Green.



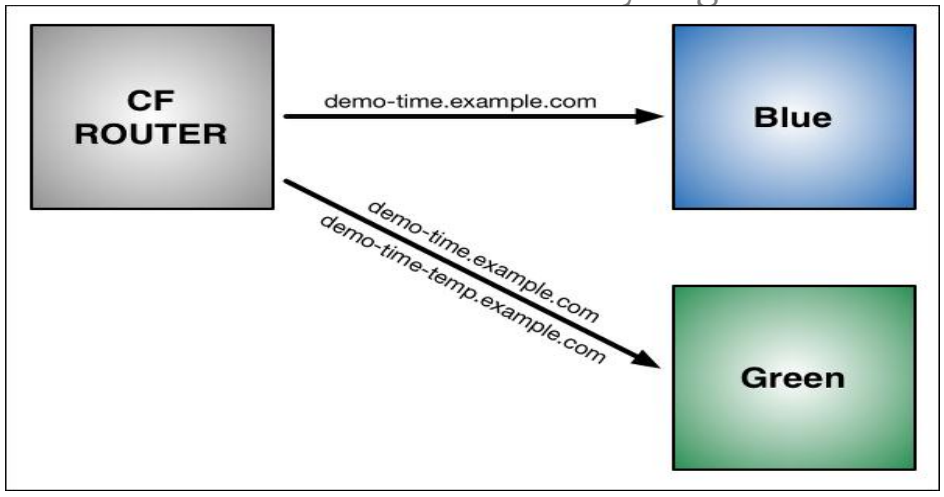
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Using Blue-Green Deployment to Reduce Downtime and Risk

- **Step 3: Map Original Route to Green**
  - Now that both apps are up and running, switch the router so all incoming requests go to the Green app and the Blue app. Do this by mapping the original URL route (demo-time.example.com) to the Green application using the cf map-route command.

```
$ cf map-route Green example.com -n demo-time
    Binding demo-time.example.com to Green... OK
```

  - The CF Router continues to send traffic for demo-time-temp.example.com to Green.
  - The CF Router immediately begins to load balance traffic for demo-time.example.com



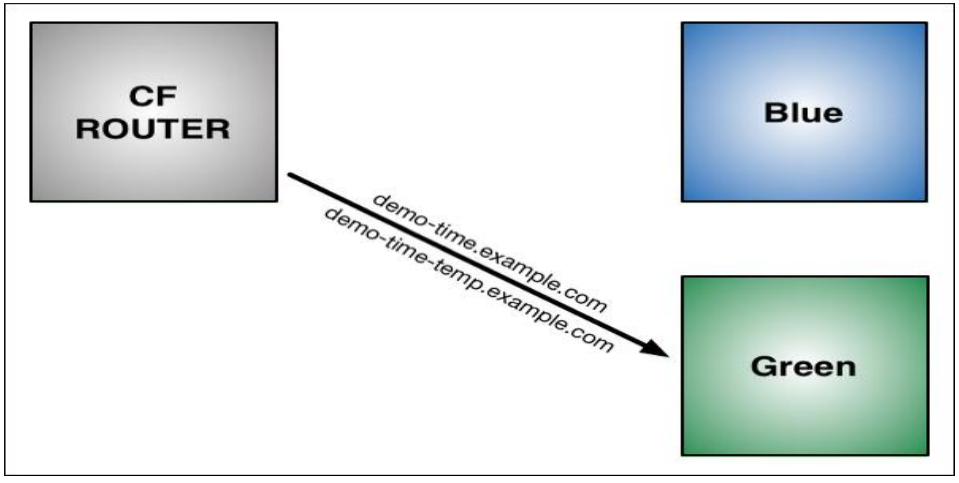
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Using Blue-Green Deployment to Reduce Downtime and Risk

- **Step 4: Unmap Route to Blue**
  - Once you verify Green is running as expected, stop routing requests to Blue using the cf unmap-route command:

```
$ cf unmap-route Blue example.com -n demo-time
Unbinding demo-time.example.com from blue... OK
```

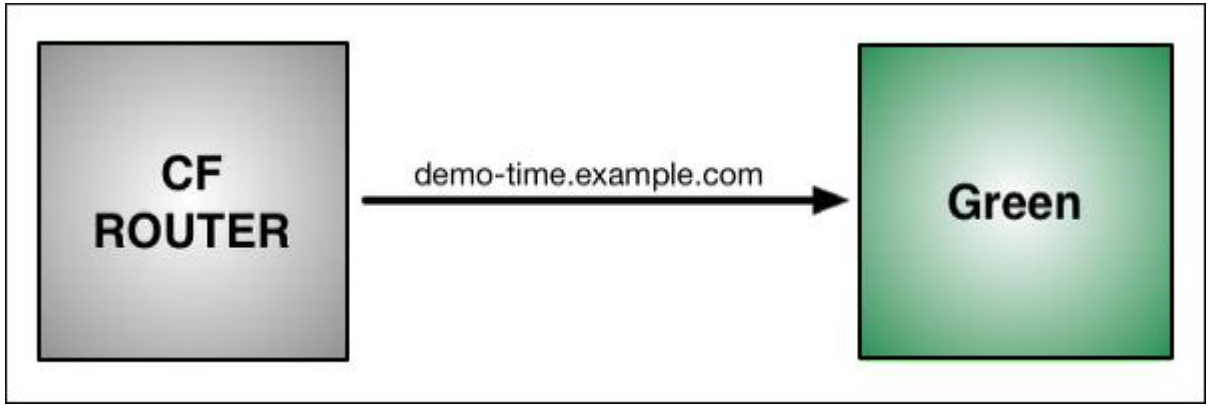
  - The CF Router stops sending traffic to Blue. Instead, it routes all traffic to demo-time.example.com to Green:between Blue and Green.



Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Using Blue-Green Deployment to Reduce Downtime and Risk

- **Step 5: Remove Temporary Route to Green**
  - You can now use cf unmap-route to remove the route to demo-time-temp.example.com.
  - You can also decommission Blue, or keep it in case you need to roll back your changes.



Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Cloud Foundry Command Line Interface (CLI)

- **Commands for Listing Users**
  - These commands take an org or space as an argument:

    cf org-users , cf space-users

- **Commands for Managing Roles**
  - These commands require Elastic Runtime admin permissions and take username, org or space, and role as arguments:

    cf set-org-role , cf unset-org-role, cf set-space-role, cf unset-space-role
  - Available roles are "OrgManager", "BillingManager", "OrgAuditor", "SpaceManager", "SpaceDeveloper", and "SpaceAuditor".

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Services and Service Instances

- Cloud Foundry offers a marketplace of services, from which users can provision reserved resources on-demand.
- Examples of resources services provide include databases on a shared or dedicated server, or accounts on a SaaS application.
- These resources are known as Service Instances and the systems that deliver and operate these resources are known as Services.
- Think of a service as a factory that delivers service instances.
- Cloud Foundry enables users to automatically provision credentials needed to reach a service instance. These credentials can be managed automatically for use by applications on Cloud Foundry, or managed manually for use by external and local clients.
- Service instance credentials can be delivered automatically to applications running on Cloud Foundry in an environment variable.
- Credentials managed manually are known as Service Keys.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Services and Service Instances

- **Managing Service Instances with the CLI**
- **List Marketplace Services**
  - After targeting and logging into Cloud Foundry, you can view what services are available to your targeted organization with the command cf marketplace.
  - Available services may differ between organizations and between Cloud Foundry marketplaces.

```
$ cf marketplace
Getting services from marketplace in org my-org / space test as me@example.com...
OK

service        plans          description
p-mysql         100mb, 1gb        A DBaaS
p-riakcs        developer         An S3-compatible object store
```


Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Services and Service Instances

- **Managing Service Instances with the CLI**
- **List Marketplace Services**
  - After targeting and logging into Cloud Foundry, you can view what services are available to your targeted organization with the command cf marketplace.
  - Available services may differ between organizations and between Cloud Foundry marketplaces.

```
$ cf marketplace
Getting services from marketplace in org my-org / space test as me@example.com...
OK

service        plans          description
p-mysql         100mb, 1gb        A DBaaS
p-riakcs        developer         An S3-compatible object store
```


Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Services and Service Instances

Creating Service Instances

- You can create a service instance with the command:

  cf create-service SERVICE PLAN SERVICE_INSTANCE

- SERVICE The service you choose.
- PLAN Service plans are a way for providers to offer varying levels of resources or features for the same service.
- SERVICE_INSTANCE A name you provide for your service instance. This is an alias for the instance which is meaningful to you. Use any series of alpha-numeric characters, hyphens (-), and underscores (_). You can rename the instance at any time.

$ cf create-service rabbitmq small-plan my_rabbitmq

Creating service my_rabbitmq in org console / space development as user@example.com...

OK

Capgemini CONSULTING.TECHNOLOGY.OUTSOURCING

Services and Service Instances

Instance Tags

- Some services provide a list of tags that Cloud Foundry delivers in the VCAP_SERVICES Environment Variable. These tags provide developers with a more generic way for applications to parse VCAP_SERVICES for credentials. Developers may provide their own tags when creating a service instance by including a comma-separated list of tags with the -t flag.

```
$ cf create-service my-db-service small-plan my-db  -t "prod, workers”
 Creating service my-db in org console / space development as user@example.com...
OK
```


Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Services and Service Instances

- **List Service Instances**

- **You can list the service instances in your targeted space with the command cf services. The output includes any bound apps, along with the state of the last requested operation for the service instance.**

```
$ cf services
Getting services in org my-org / space test as user@example.com...
OK

name          service    plan        bound apps    last operation
mybucket      p-riakcs   developer   myapp          create succeeded
mydb          p-mysql     100mb                     create succeeded
```


Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Services and Service Instances

- **Get Details for a Particular Service Instance**
  - Details include dashboard urls, if applicable, and operation start and last updated timestamps.

cf service mydb

Service instance: mydb

Service: p-mysql

Plan: 100mb

Description: MySQL databases on demand

Documentation url:

Dashboard: https://p-mysql.example.com/manage/instances/cc4eab9d-aff4-4beb-bc46-123f2a02dcf1

Last Operation

Status: create succeeded

Message:

Started: 2015-05-08T22:59:07Z

Updated: 2015-05-18T22:01:26Z

Capgemini CONSULTING.TECHNOLOGY.OUTSOURCING

Services and Service Instances

- **Bind a Service Instance**

- **Depending on the service, you can bound service instances to applications and/or routes.**

```
$ cf bind-service my-app mydb
Binding service mydb to my-app in org my-org / space test as me@example.com...
OK
TIP: Use 'cf push' to ensure your env variable changes take effect

$ cf restart my-app
```

Note: You must restart or in some cases re-push your application for changes to be applied to the VCAP_SERVICES environment variable and for the application to recognize these changes.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Services and Service Instances

- **Bind a Service Instance**

- **Binding with Application Manifest**

As an alternative to binding a service instance to an application after pushing an application, you can use the application manifest to bind the service instance during push. Using the manifest to bind service instances to routes is also not supported.

```
services:
 - test-mysql-01
```

The following excerpt from the cf push command and response demonstrates that the cf CLI reads the manifest and binds the service instance to an app called test-msg-app.

```
$ cf push
Using manifest file /Users/Bob/test-apps/test-msg-app/manifest.yml
...
Binding service test-mysql-01 to test-msg-app in org My-Org / space development as Bob@example.com
OK
```


Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Services and Service Instances

- **Unbind a Service Instance**

- **Unbinding a service instance from an application removes the credentials created for your application from the VCAP_SERVICES environment variable.**

```
$ cf unbind-service my-app mydb
Unbinding app my-app from service mydb in org my-org / space test as me@example.com...
OK
```

- **Note: You must restart or in some cases re-push your application for changes to be applied to the VCAP_SERVICES environment variable and for the application to recognize these changes.**

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Services and Service Instances

- **Rename a Service Instance**

```
$ cf rename-service mydb mydb1
Renaming service mydb to mydb1 in org my-org / space test as me@example.com...
OK
```

- **Keep in mind that upon restarting any bound applications, the name of the instance will change in the VCAP_SERVICES environment variable. If your application depends on the instance name for discovering credentials, changing the name could break your application's use of the service instance.**

- **Upgrade/Downgrade Service Plan**

```
$ cf update-service mydb -p new-plan
Updating service instance mydb as me@example.com...
OK
```


Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Services and Service Instances

- **Delete a Service Instance**

- **Deleting a service instance unprovisions the service instance and deletes all data associated with the service instance.**

```
$ cf delete-service mydb

Are you sure you want to delete the service mydb ? y
Deleting service mydb in org my-org / space test as me@example.com...
OK
```


Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Thanks

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING