Managing Applications in Cloud Foundry

A closer look at practical Cloud Foundry usage

Log Management, APM Integration, Autoscaling, Zero-downtime deployments

Pivotal

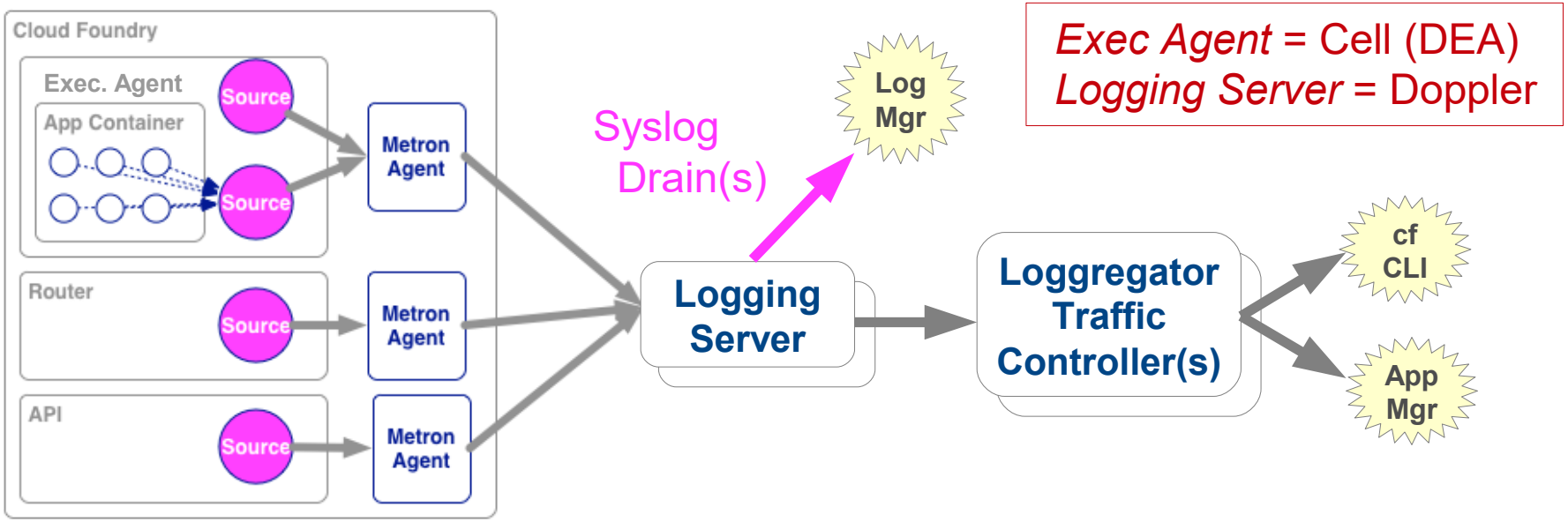
Roadmap

- **Log Management**
- Application Performance Management
- Autoscaling
- Zero-Downtime Deployments

Pivotal™

Recall: Log Aggregation Architecture

- Collects log output from app instances, CF components
- Aggregates into a consolidated log
- Sinks to cf logs, App Mgr, *third-party log managers*



Why Third-Party Log Managers?

- Recommended approach
  - Can store far more logging information than CF
  - Allow for persistence, storage, *searching*, *analyzing*, metrics
- Variety of third-party log managers supported:

sumologic papertrail

logstash vmware vCenter Log Insight splunk>storm™

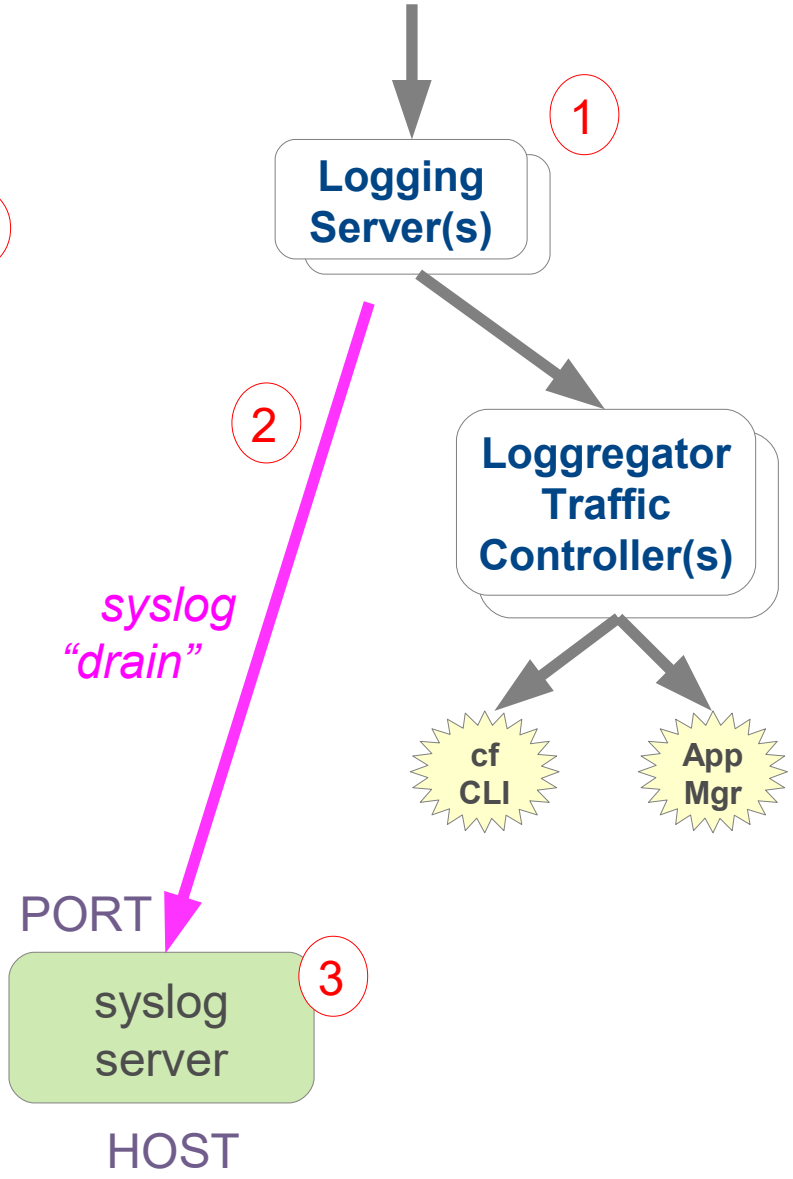
Connecting to Third-Party Log Managers

- Setup Log Manager, determine **`HOST`** and **`PORT`**.
  - Process varies according to vendor
- Create User Provided Service with a Syslog drain:
  - **`cf cups <SERVICE> -l syslog://<HOST>:<PORT>`**
- Bind to application, restage.
  - Cloud Foundry sinks loggregator output to this drain for this application
  - *See next slide ...*

Pivotal™

How It Works

- All output for app collected by Logging (Doppler) server (1)
- Loggregator opens socket to HOST:PORT (2)
  - Sends all log info for app to socket in syslog format
- Received by third-party syslog server (3)



Example: PWS and PaperTrail

- **PaperTrail:** Cloud-based Log Manager

a) Create account at https://papertrailapp.com
b) Use the “Add System” button
   - Papertrail will provide you the URL to use for your syslog drain
   - Example: logs2.papertrailapp.com:41846

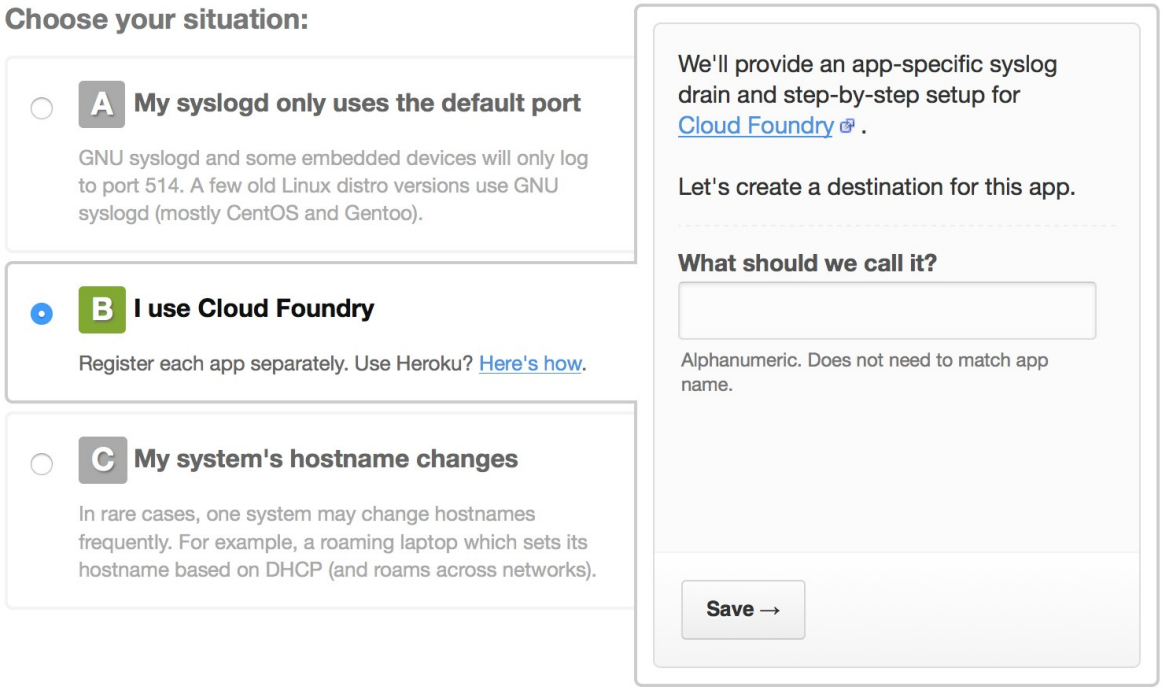
Pivotal™

Example: PWS and Papertrail

c) Click the “Alternatives” link

d) Select “*Cloud Foundry*” option

e) Name your system



Example: PWS and PaperTrail

f) Setup user defined service using Papertrail's URL:

MyCFSystem will log to **logs2.papertrailapp.com:15957**.

g) Create User Provided Service with a Syslog drain:

```
cf cups the-drain -l
syslog://logs2.papertrailapp.com:15957
```

h) Bind to application, restart:

```
cf bind-service the-app the-drain
cf restart the-app
```

About Syslog

- De facto standard for logging on Unix/Linux
  - Can log to a file or a server *syslogd* (via a protocol)
  - Splunk, Papertrail and others provide syslog servers
- To log to syslog
  - Generate a TCP or UDP message in the right format
  - Open a socket to your syslog server and send
- Higher level logging options exist
  - https://github.com/cloudfoundry-community/java-loggregator
  - Output handlers for Java logging or log4j/logback

Pivotal™

Lab

Configuring Third-Party Log Management Tools with Cloud Foundry

Pivotal™

Roadmap

- Log Management
- **Application Performance Management**
- Autoscaling
- Zero-Downtime Deployments

Pivotal™

Application Performance Monitoring

- Logs and analysis only takes you so far
- Important to have real-time monitoring of applications
  - Uptime, performance, etc.
- Application Performance Monitoring (APM) Tools
  - Monitor your application while running
  - Several choices available in Cloud Foundry
    - PWS – New Relic and AppDynamics
    - Pivotal Spring Insight

Pivotal Web Services and New Relic

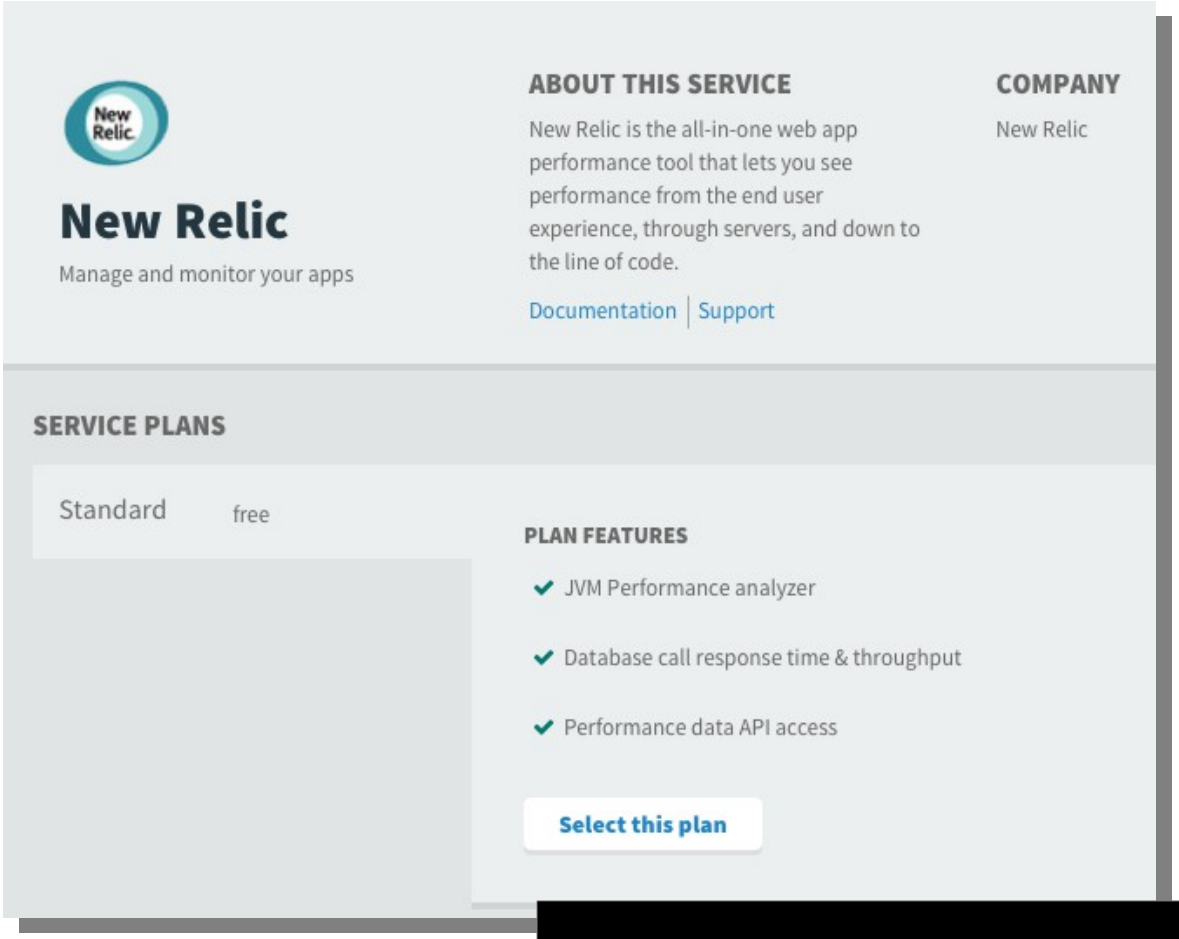
New Relic

- PWS offers simple interface to New Relic
  - Available as Marketplace Service
  - Tracks different instances of application
  - Monitors down to the line of code

- How To Use:
  - Create New Relic service in desired space
  - Bind to desired Application(s)
  - Re-stage application
    - Java Buildpack includes New Relic Agent, others may not
  - APM available as a link from within PWS

Pivotal

Creating the New Relic Service

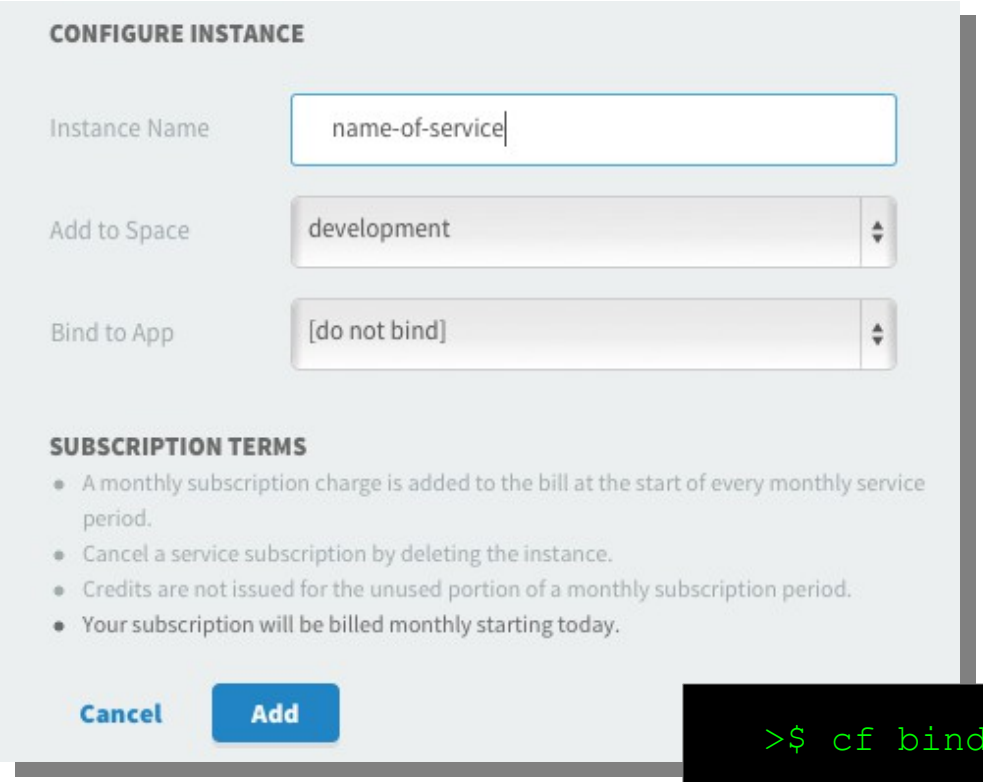


- Use App Manager Console
- Use **cf** CLI

```
>$ cf create-service newrelic standard apm
```

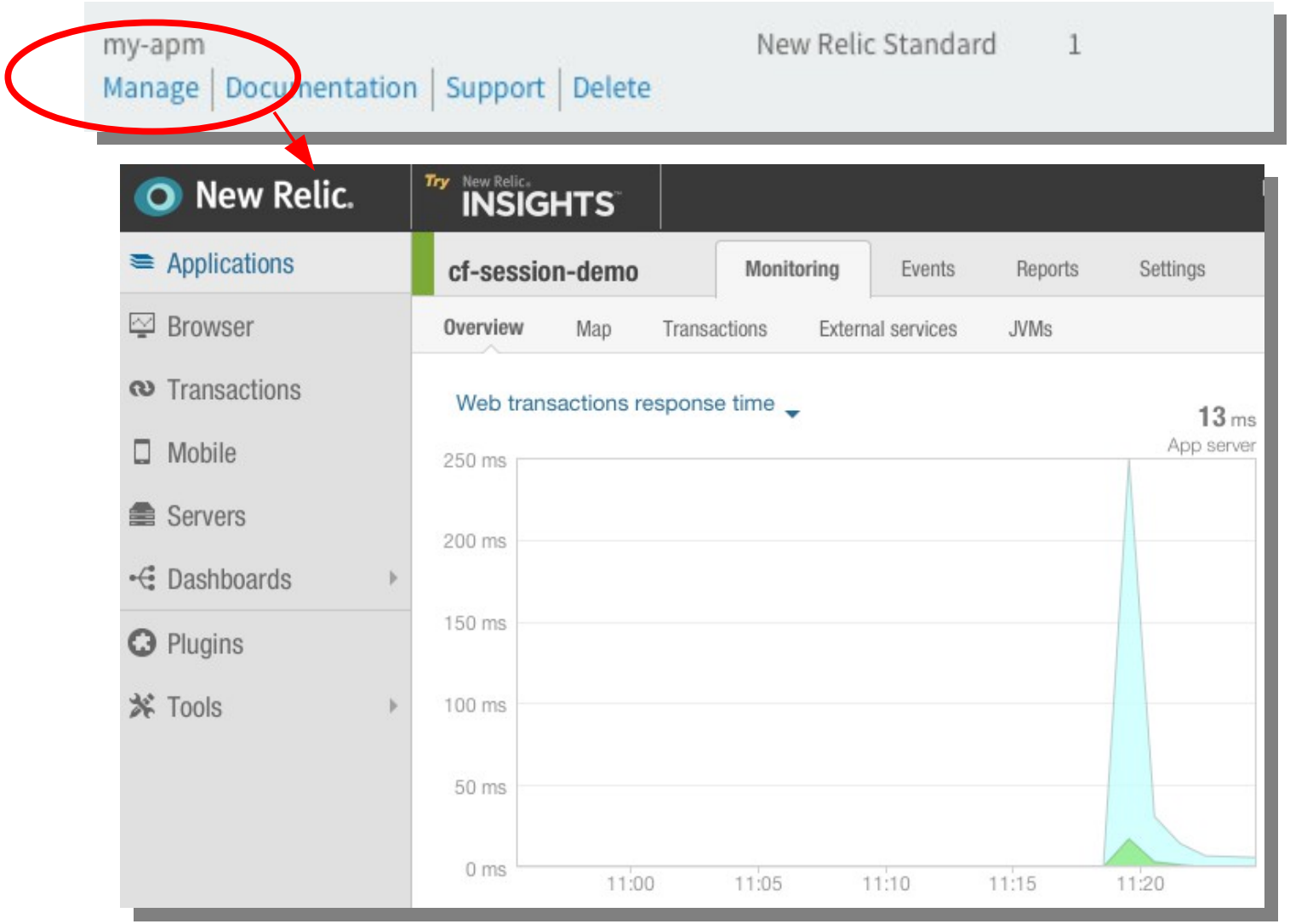
Create Service / Bind Application

- Use **cf** CLI or App Manager Console:




```
>$ cf bind-service some-app apm
```

Access via *Manage* Link in App Manager



Lab

Configuring Third-Party Application Performance Management Tools with Cloud Foundry

Pivotal™

Roadmap

- Log Management
- Application Performance Management
- **Autoscaling**
- Zero-Downtime Deployments

Pivotal™

Scaling Options

- CF allows horizontal scaling
  - Controlling the # of instances of an application running
  - All behind a common router (load balancer)
  - Controllable via the manifest, **cf** command line, or App Manager console

- All options require manual intervention

Pivotal™

AutoScaling

- CF can allow applications to be automatically scaled
  - “AutoScaling”

- System load can be used as a trigger in place of manual interaction.

- Autoscaling Service
  - Must be installed by administrator
  - Not available in PWS

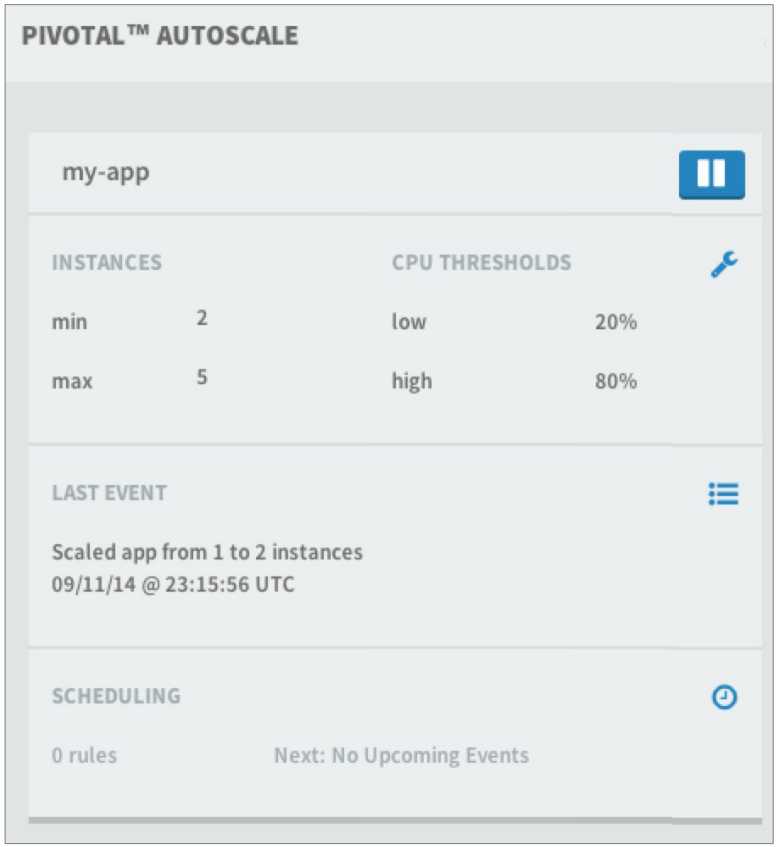
Pivotal™

AutoScaling Service – Steps

1) Create the service
   1) Select the desired plan

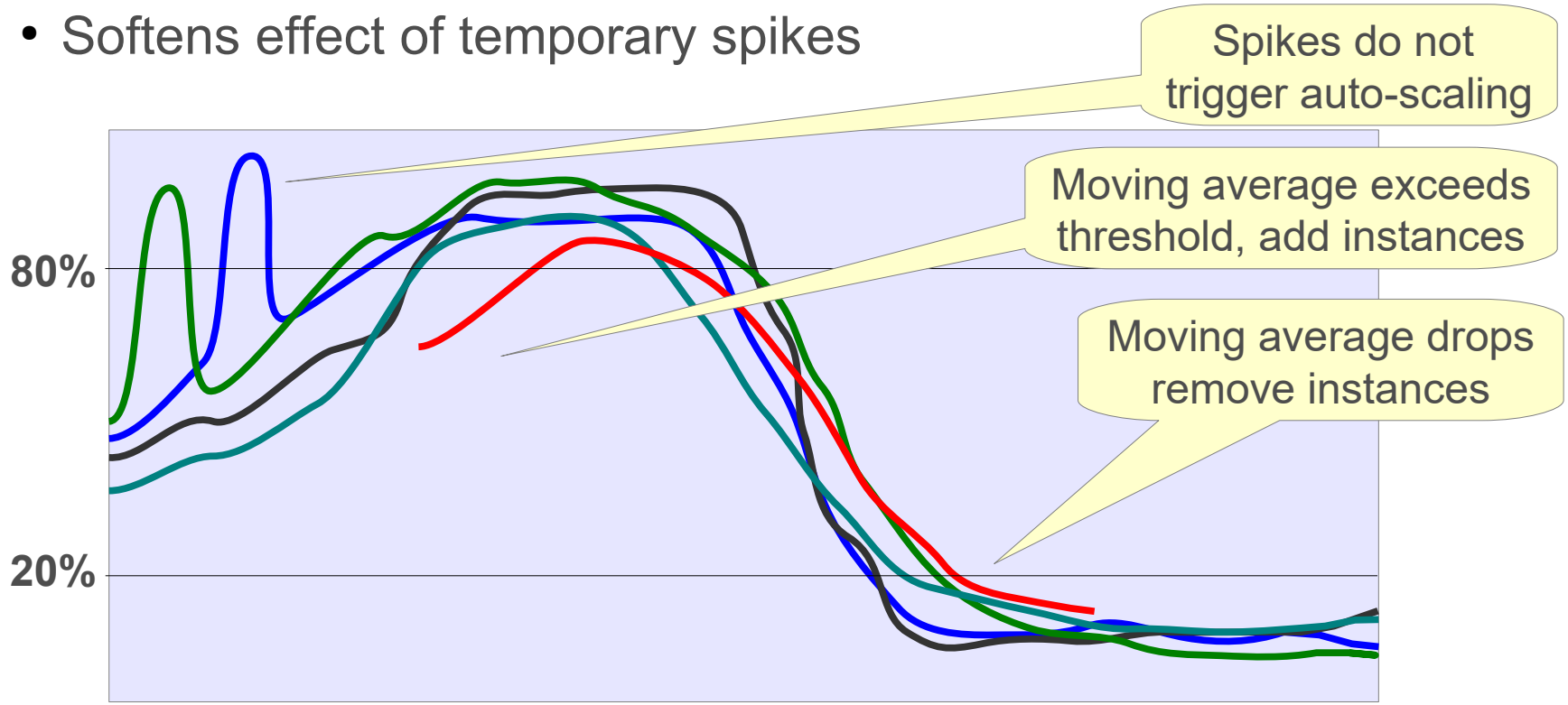
2) Bind to Application

3) Set desired scaling parameters
   1) Add instance whenever high threshold is reached
   2) Subtract instance whenever low threshold is reached

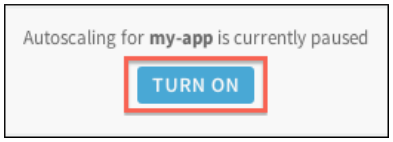


AutoScaling – Moving Average

- Scaling activity based on moving averages
  - Softens effect of temporary spikes

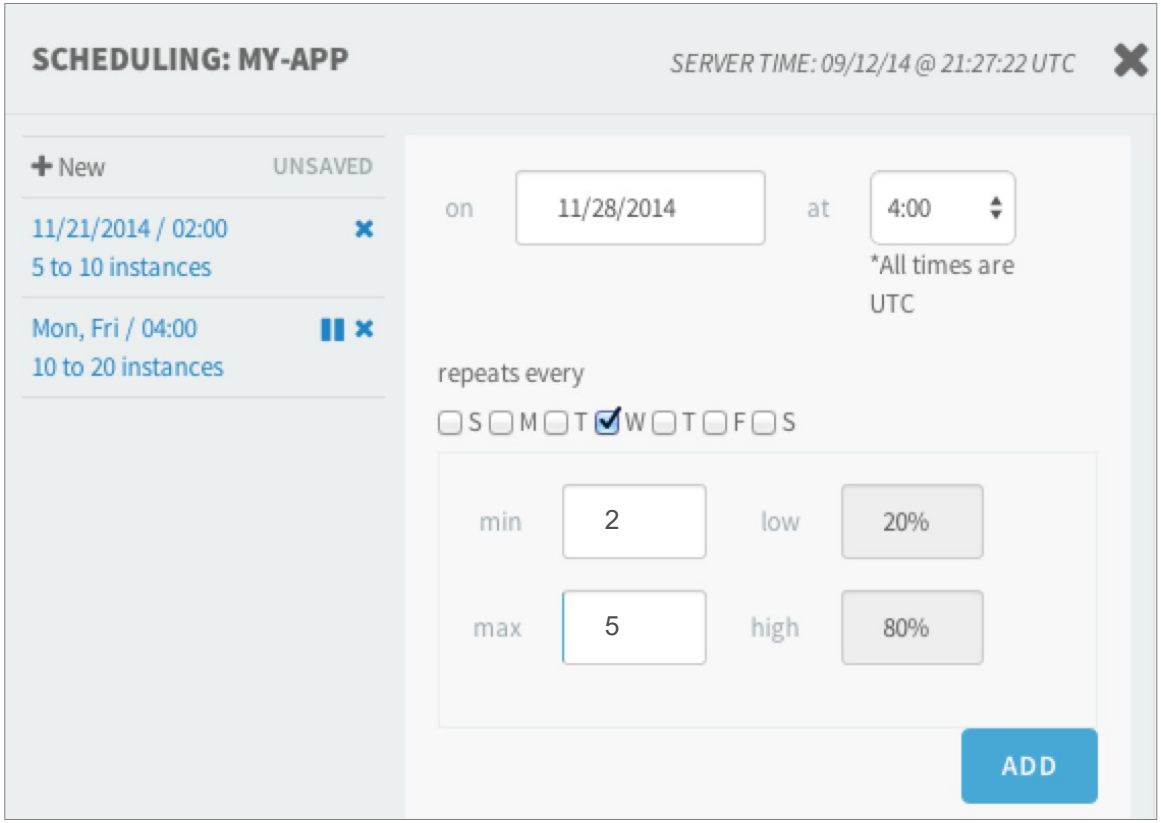


- Manual scaling disables AutoScaling
  - Re-enable:



AutoScaling – Scheduling

- Autoscaling events can be scheduled
- Changes autoscaling behavior on the given date / time.
- May be single event or recurring



Roadmap

- Log Management
- Application Performance Management
- Autoscaling
- **Zero-Downtime Deployments**

Pivotal™

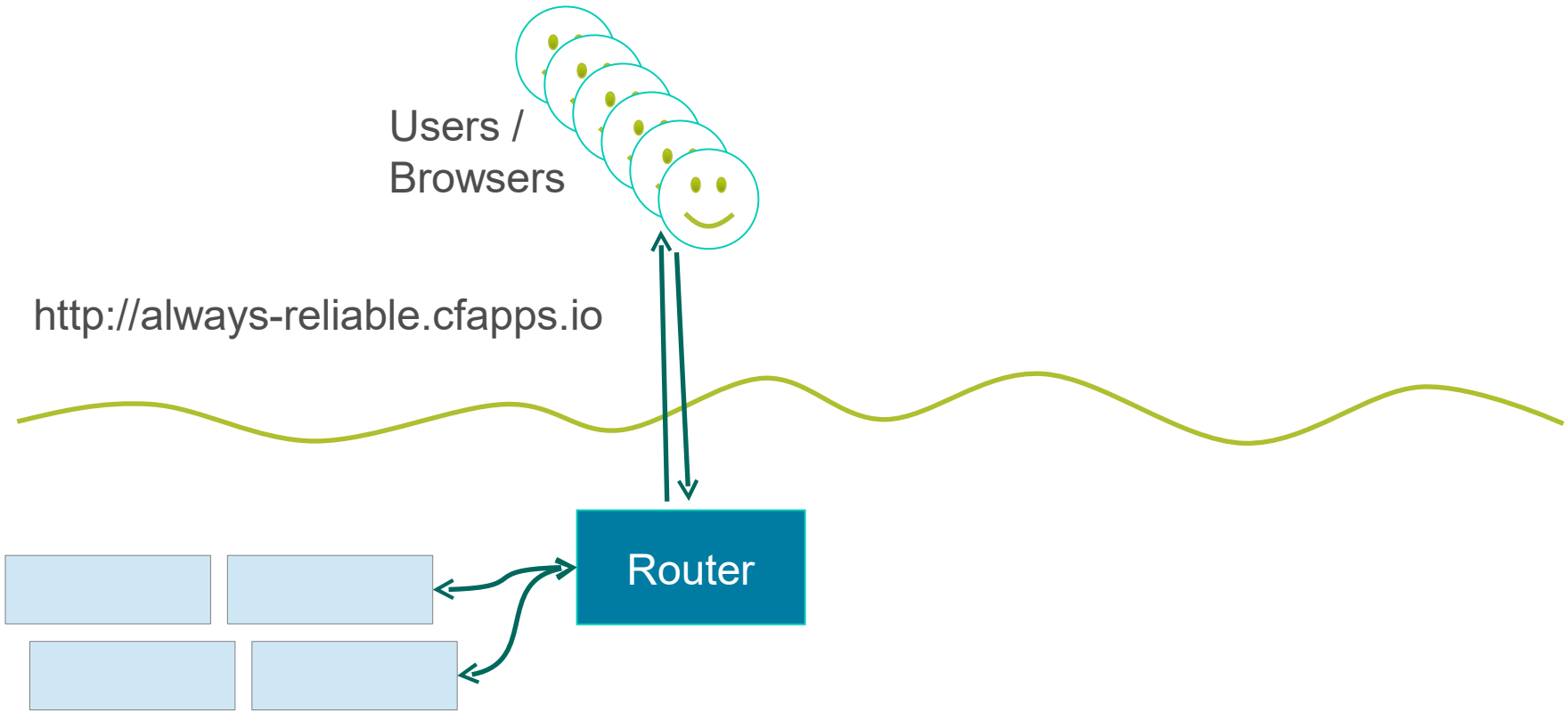
Blue / Green Deployments

- `cf push` causes CF to stop old instances, then start new
  - Bad news if you are a user
- Blue / Green Deployment eliminates user downtime
  - Also known as “zero-downtime” or “A / B” Deployment
  - Avoids *“Site Temporarily Down for Maintenance”*
- How it works:
  - Run 2 versions of an application (new / old)
    - NOT merely multiple instances.
  - Alter routes for applications to transfer traffic.
  - **Note:** Users can still experience session loss.

Pivotal™

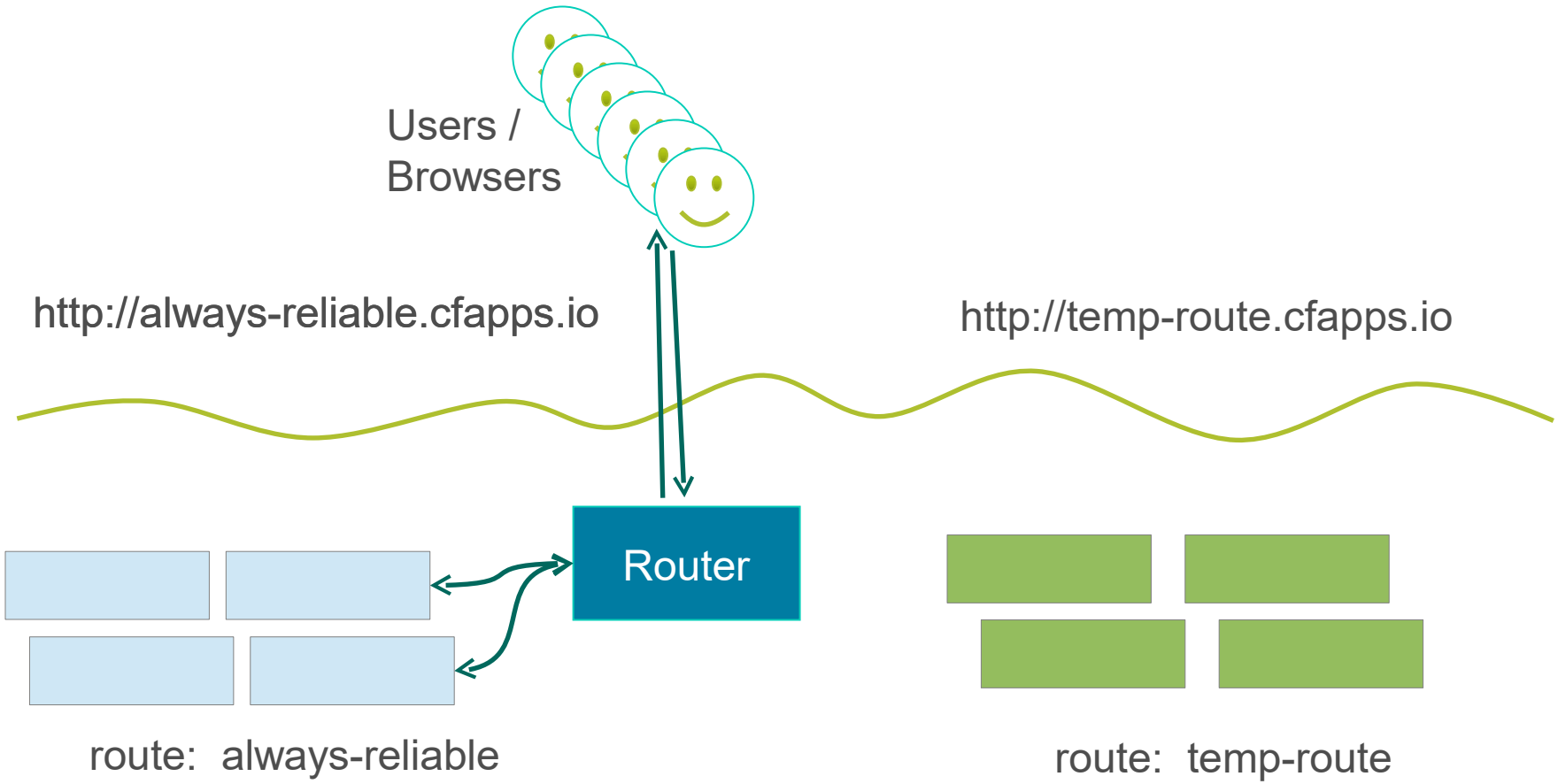
Blue Green Deployment – Existing App

```
cf push blue -p app.war -n always-reliable -i 4
```



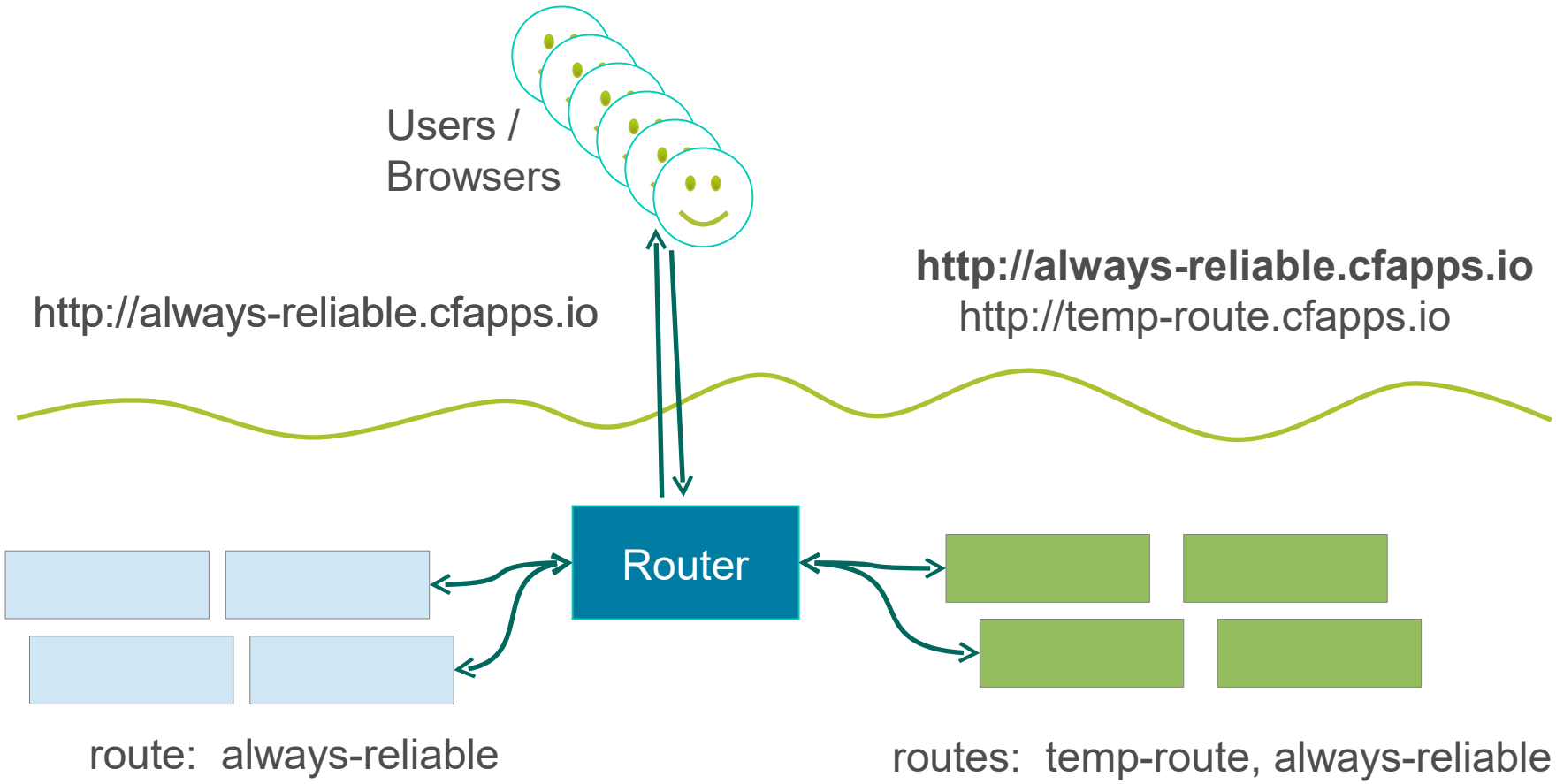
Blue Green Deployment – New Version

```
cf push green -p app.war -n temp-route -i 4
```



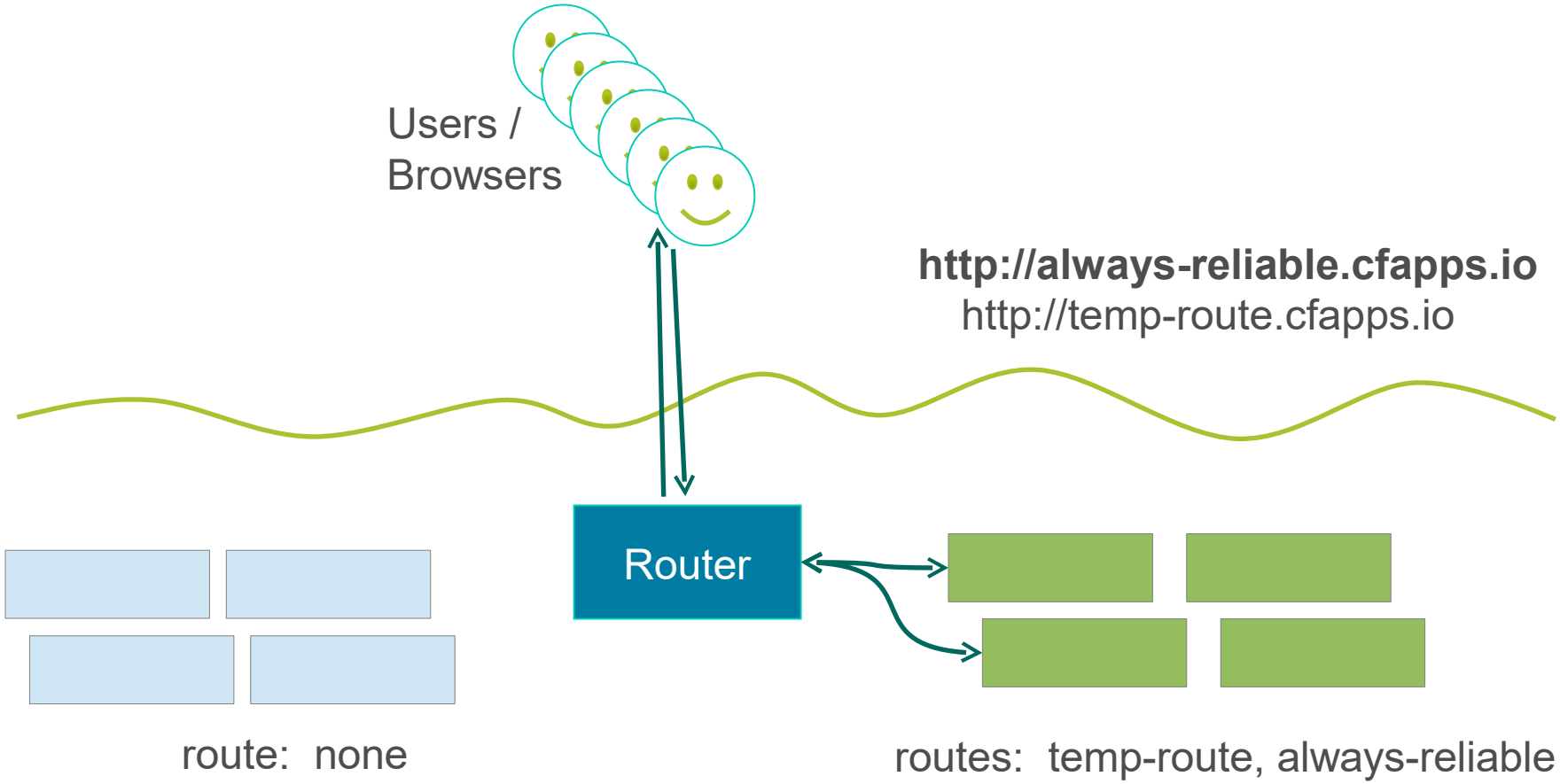
Blue Green Deployment – Duplicate Route

```
cf map-route green cfapps.io -n always-reliable
```



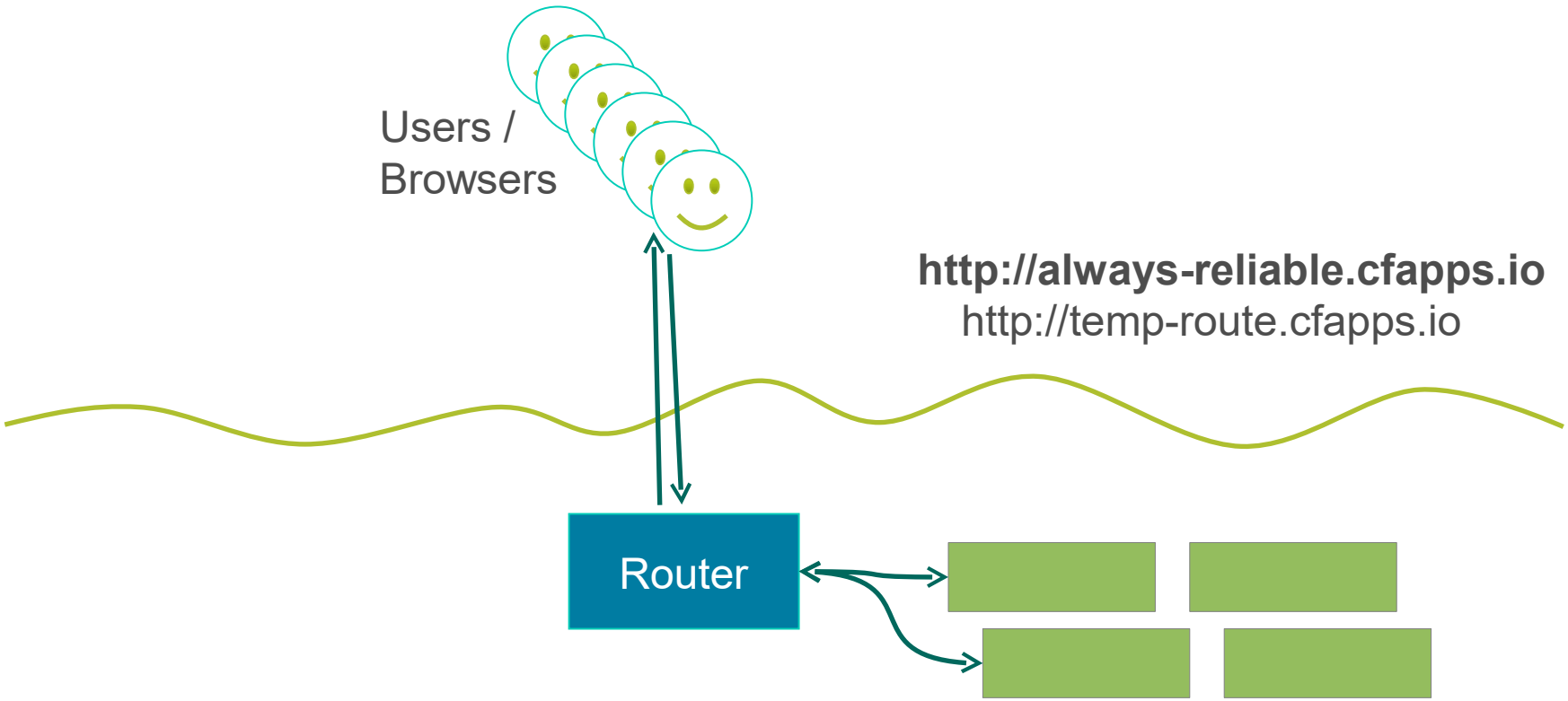
Blue Green Deployment – Disconnect Blue

```
cf unmap-route blue cfapps.io -n always-reliable
```



Blue Green Deployment – Remove Blue

```
cf delete blue
```



routes: temp-route, always-reliable

Canary Deployments

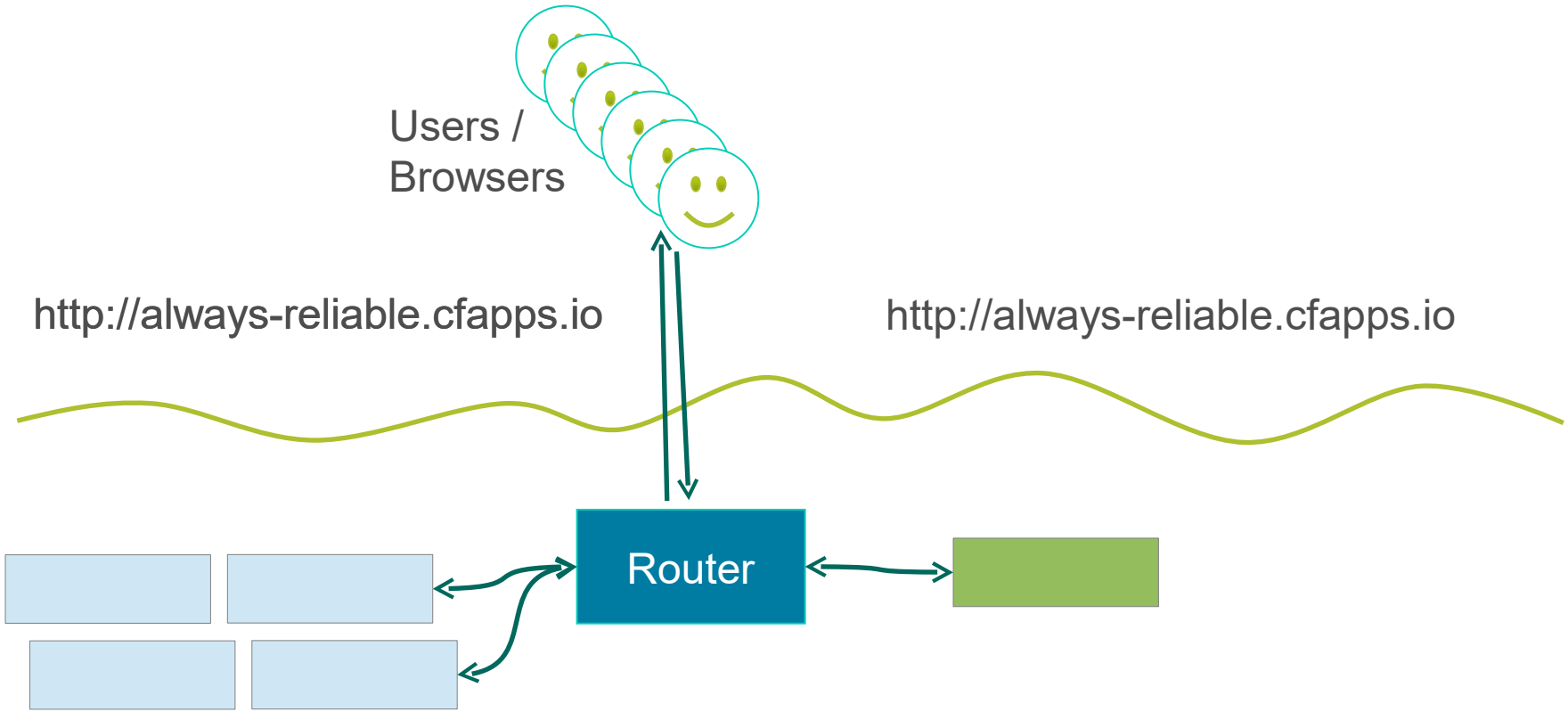
- Variation on the Blue/Green Deployment.
  - “Canary in a coal mine”

1. Start with many 'blue' instances
2. Start a single 'green' instance, route traffic to both
   - Green instance is the 'Canary'
3. Watch the Canary
   - If it behaves, scale 'green' up / scale 'blue' down.
4. Continue monitoring and scaling until zero blue instances

Pivotal™

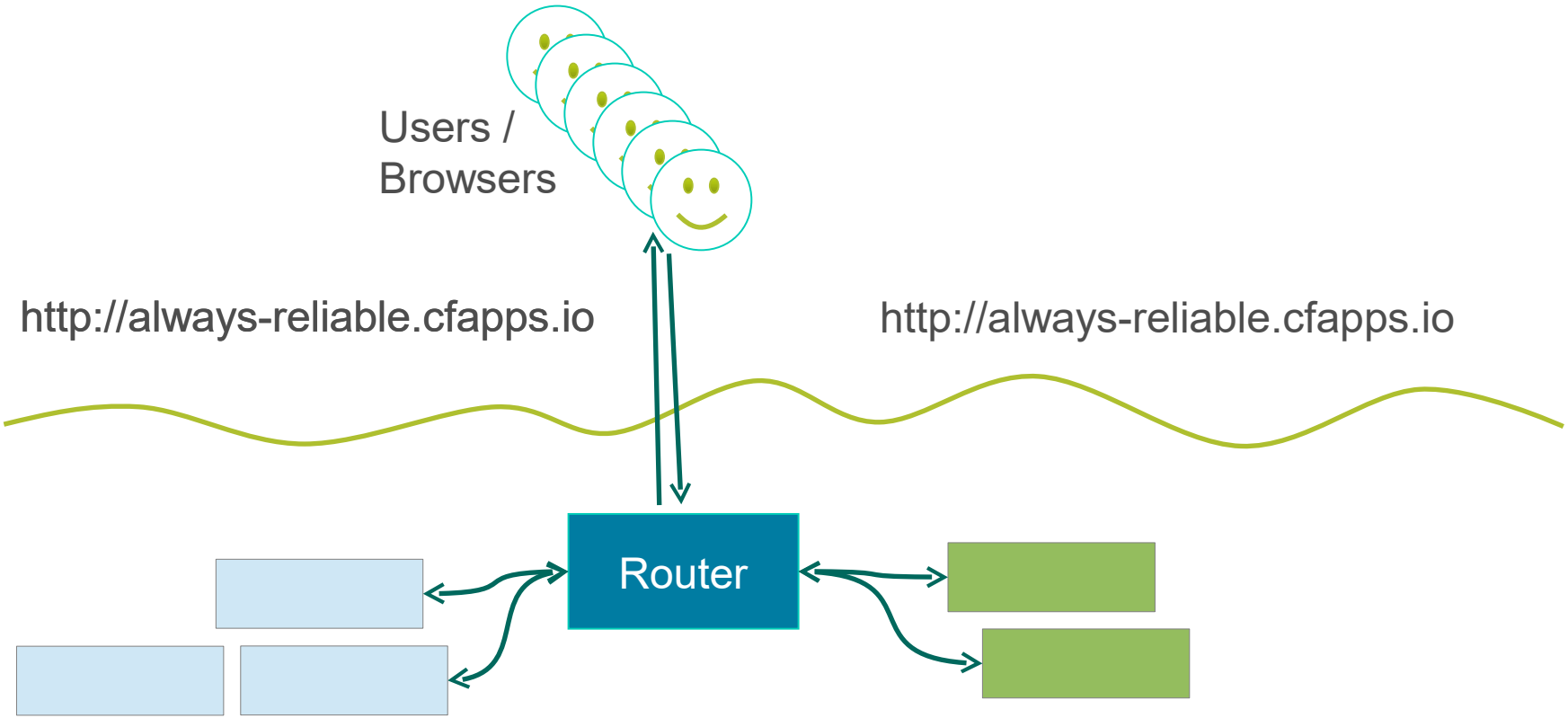
Canary Deployment – Push The Canary

```
cf push green -p app.war -n always-reliable -i 1
```



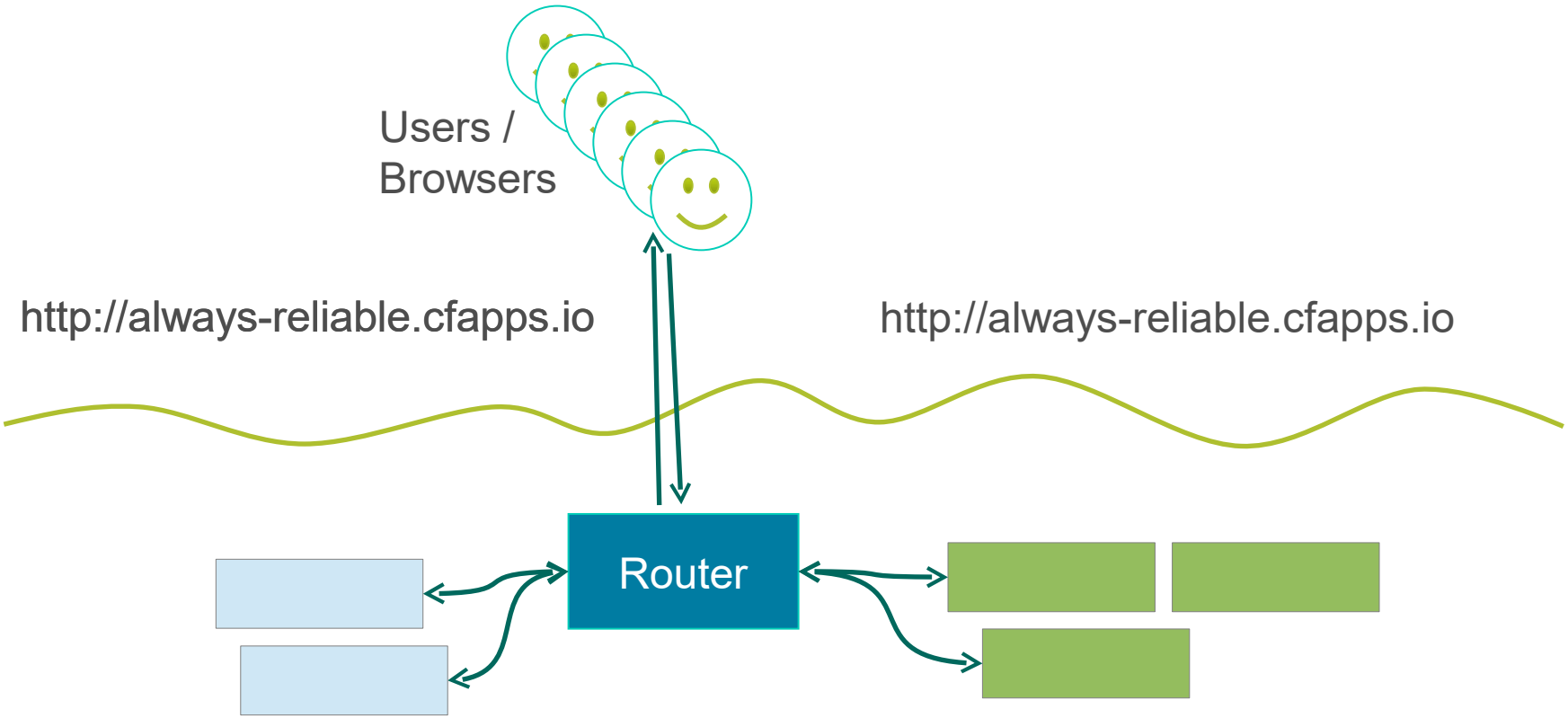
Canary Deployment – Scale Traffic

```
cf scale green -i 2
cf scale blue -i 3
```



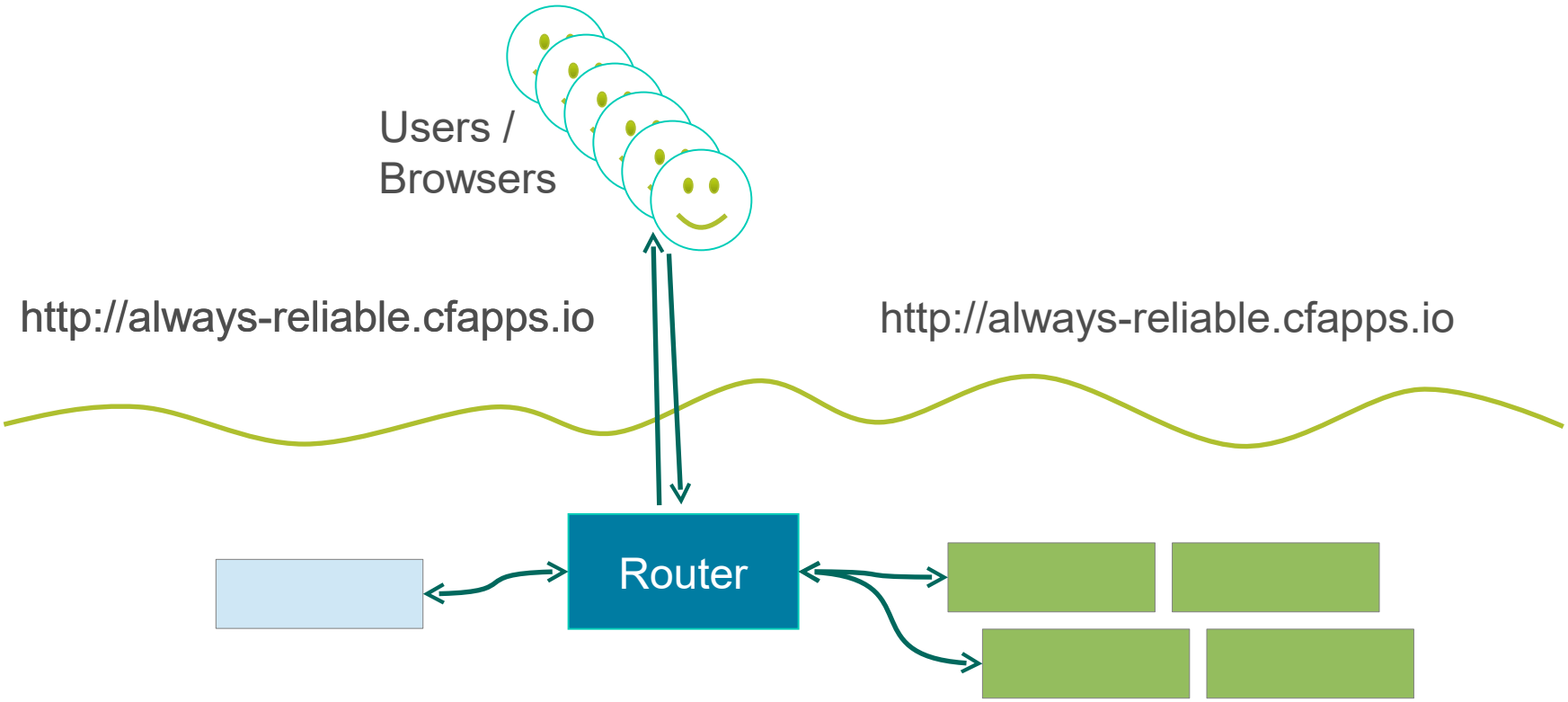
Canary Deployment – Scale Traffic

```
cf scale green -i 3
cf scale blue -i 2
```



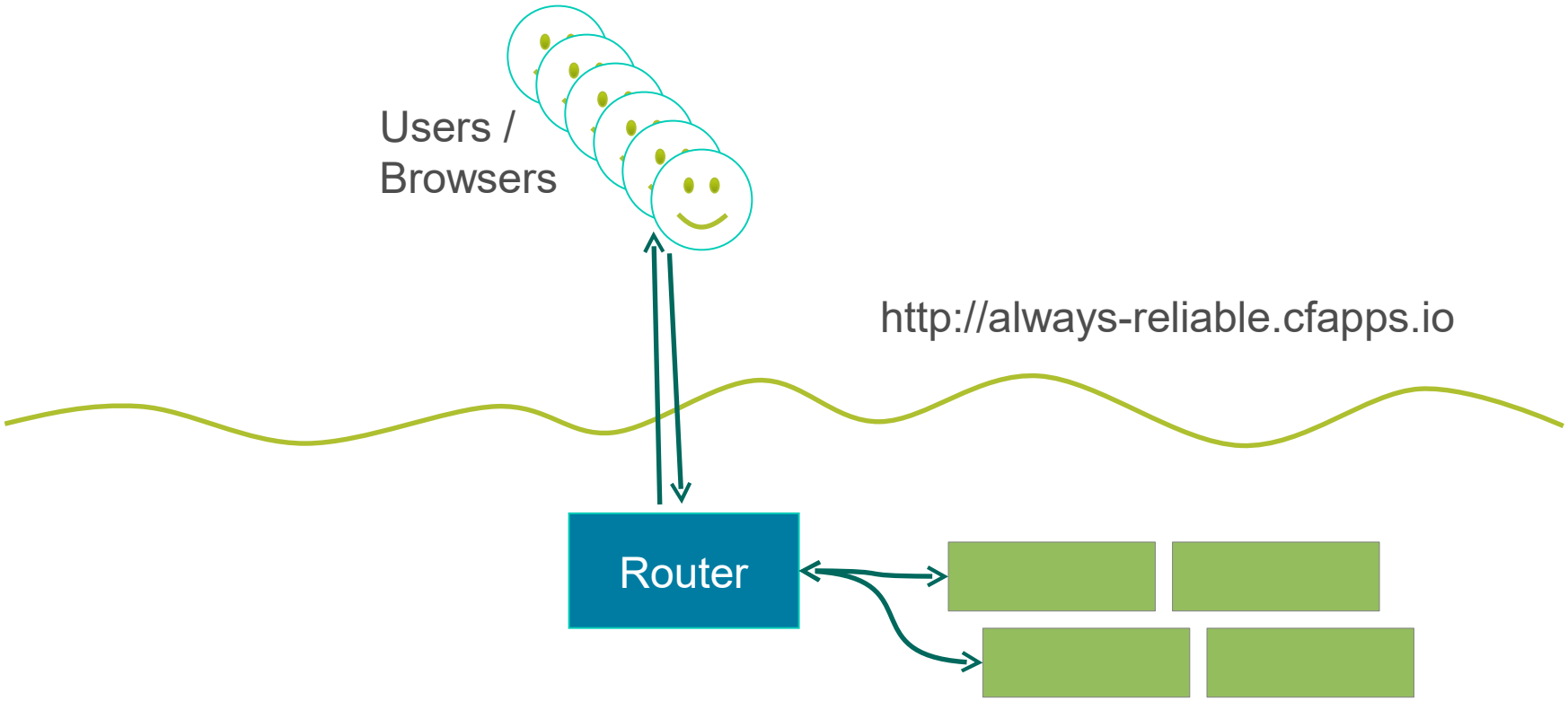
Canary Deployment – Scale Traffic

```
cf scale green -i 4
cf scale blue -i 1
```



Canary Deployment – Scale Traffic

```
cf delete blue
```



Lab

Blue / Green Deployment

Pivotal™

Summary

- After completing this lesson, you should have learned:
  - How to integrate with third-party log manager
  - How to integrate with APM services
  - How to employ App Autoscaling
  - How to deploy with zero downtime.

Pivotal™