Deploying R for production

2019-01-23

Stockholm R User Group

IBM

Holger Hellebro

IT Architect & Data Scientist
Cognitive & Analytics
IBM Global Business Services

https://www.linkedin.com/in/holgerhellebro
Twitter: @holken

Can we deploy R code?

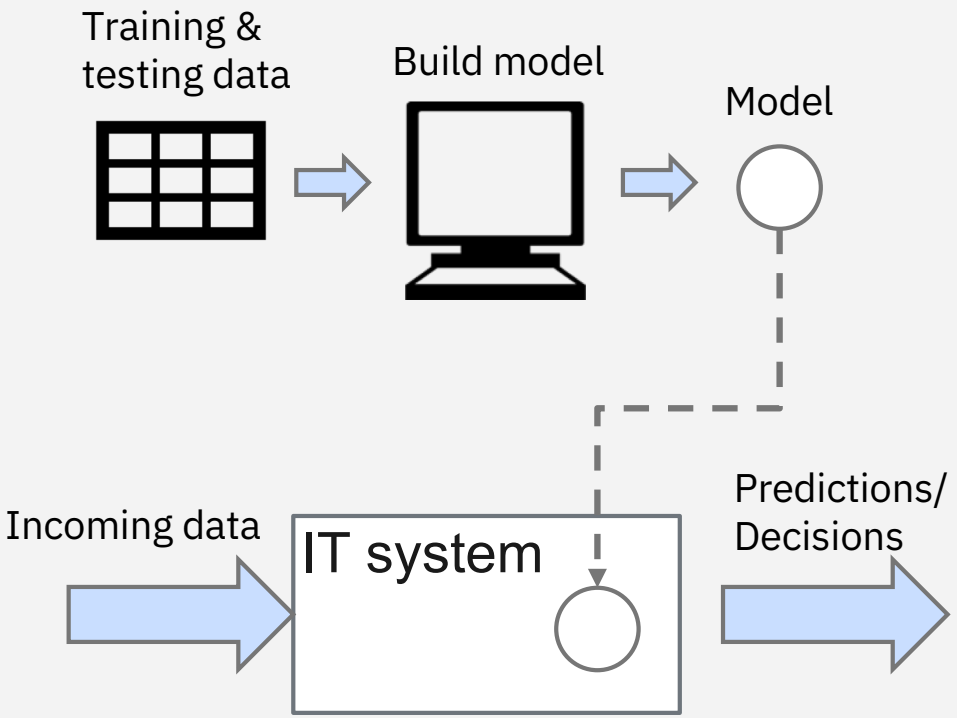
Why would we?

Will it scale?

How?

Security?

Why deploy R stuff?



Models for scoring

Visualizations

Data preparation

Feature extraction

Verify with more data

Deploy or rewrite?

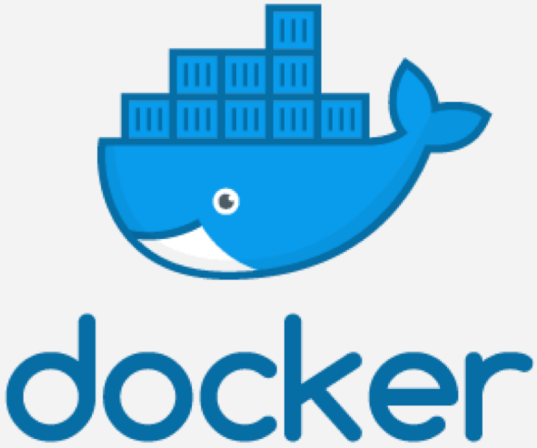
Why deploy?

- Reuse code
- Consistency
- Easier to maintain

Why rewrite?

- Minimize number of languages
- IT team skills
- Extreme performance / latency

Modern cloud technologies



Helps us package code and data.

Helps us run it on a cluster.

Docker images & containers

Means of packaging and running software

Independent of host system

Include all dependencies

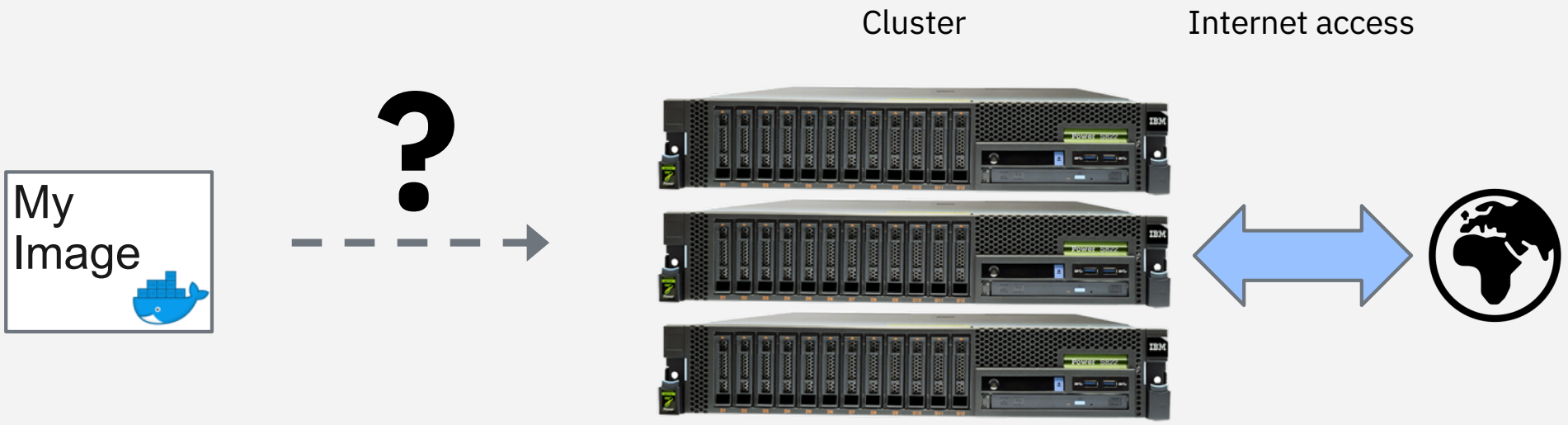
Light weight

"Dockerfile" tells docker how to build into an image

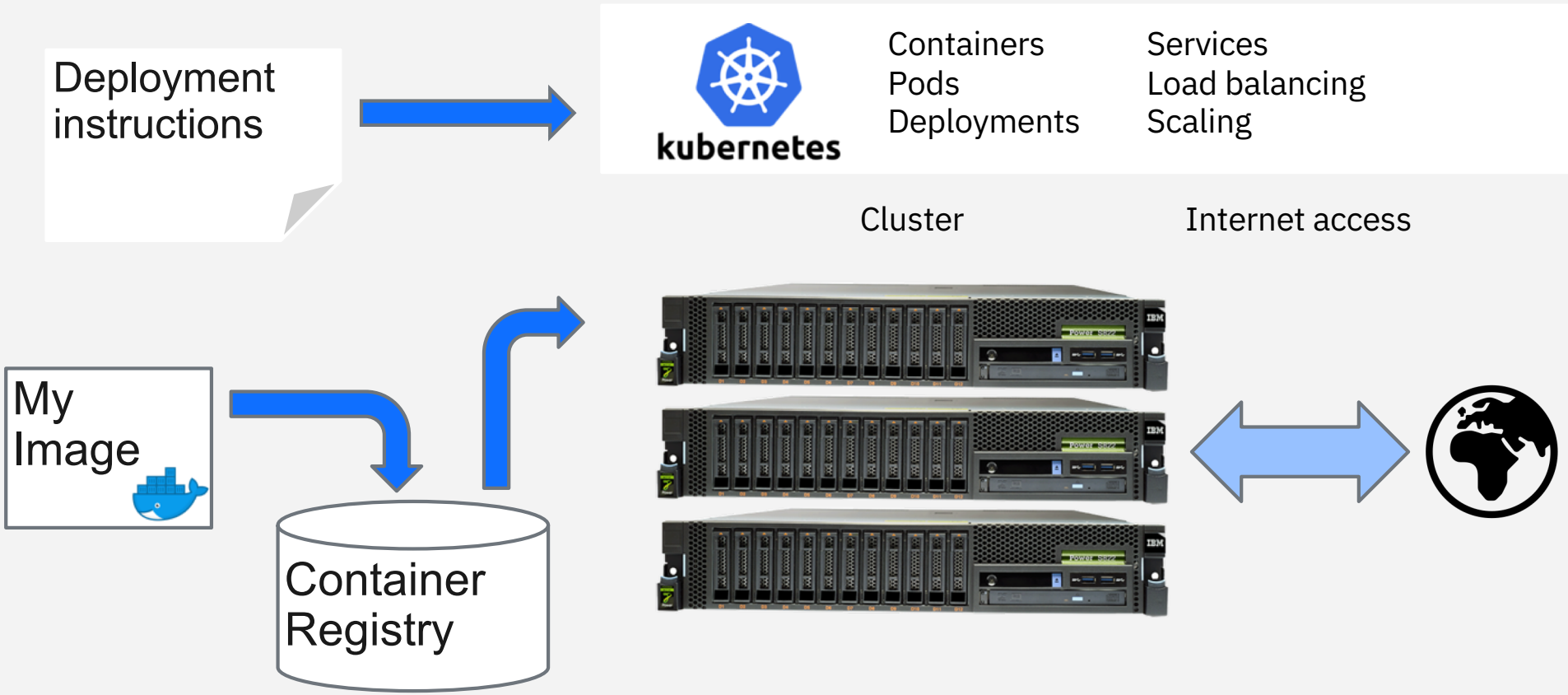
Images contain all dependencies, but not a full Operating System

Container images are uploaded to a **registry** to simplify deployment

Kubernetes



Kubernetes


Deployment instructions
kubernetes
Containers
Pods
Deployments
Services
Load balancing
Scaling
Cluster
Internet access
My Image
Container Registry

Deploying a Shiny app

Developing and building

app.R

```r
library(shiny)

ui <- fluidPage(
  titlePanel("My Shiny App"),
  mainPanel(
    actionButton("button", "Randomize"),
    plotOutput("distPlot")
  )
)

server <- function(input, output) {
  output$distPlot <- renderPlot({
    input$button
    hist(rnorm(1000, 0, 1), breaks = 30,
         col = 'darkgray', border = 'white')
  })
}

shinyApp(ui = ui, server = server)
```

Dockerfile

```
FROM rocker/shiny:latest
WORKDIR /srv/shiny-server/
COPY . .
EXPOSE 3838
CMD ["/usr/bin/shiny-server.sh"]
```

```
$ docker build . -t holken/myshiny

$ docker run -p 3838:3838 holken/myshiny

$ docker push holken/myshiny
```

Deploy

Deployment descriptor

- What container(s)
- Number of instances (replicas)

- Secrets
- Persistent storage
- Resource requirements & limits

…

deployment.yaml

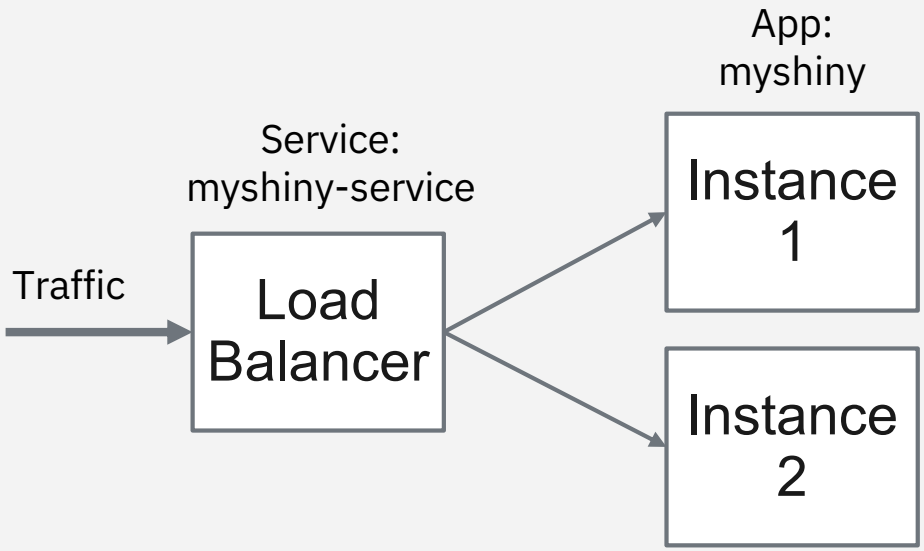

```
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: myshiny-deployment
spec:
  replicas: 2
  template:
    metadata:
      labels:
        app: myshiny
    spec:
      containers:
      - name: myshiny
        image:
docker.io/holken/myshiny:latest
```

Networking

kubernetes

Isolated by default

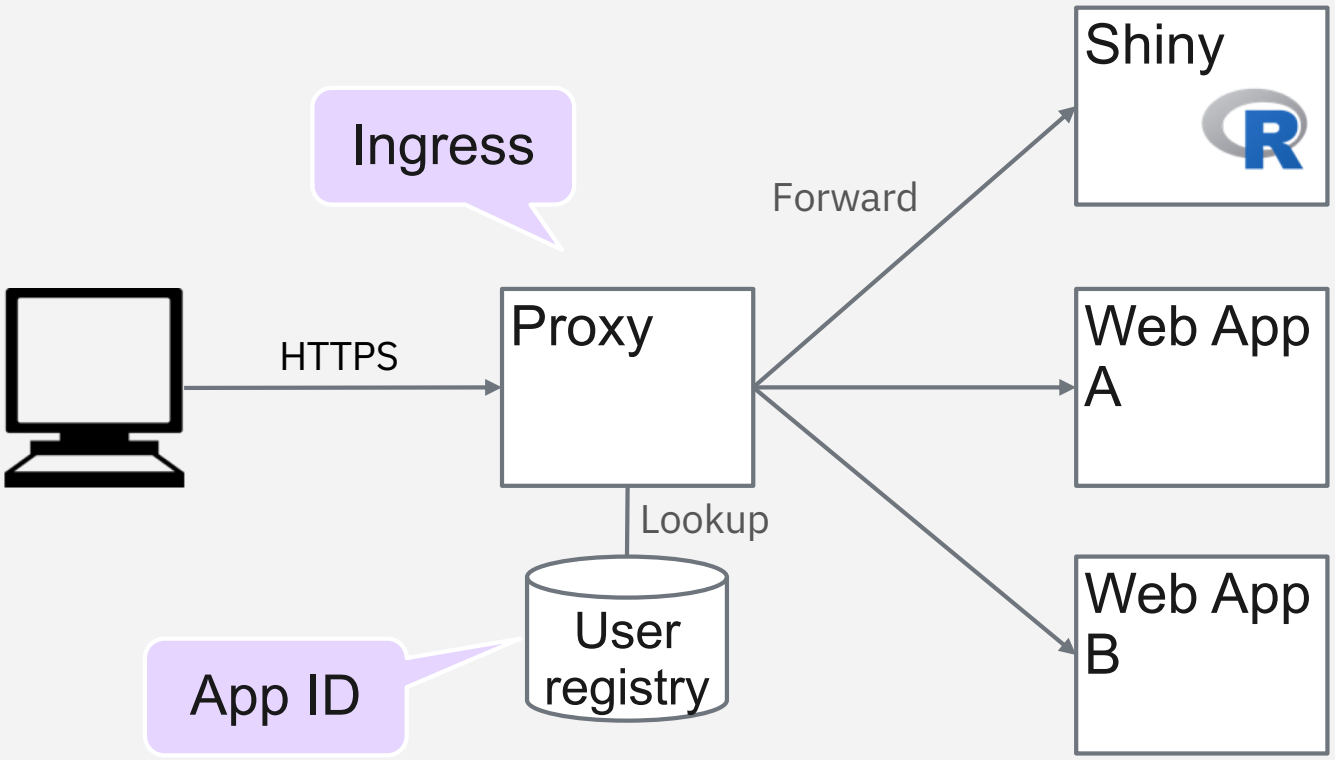
“Service” defines networking



service.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: myshiny-service
spec:
  ports:
  - port: 80
    targetPort: 3838
    protocol: TCP
  type: LoadBalancer
  selector:
    app: myshiny
```

Making it secure with Ingress and App ID



IBM Cloud Docs / IBM Cloud Kubernetes Service

Exposing apps with Ingress

Last Updated: 2019-01-21 | Edit in GitHub

App ID

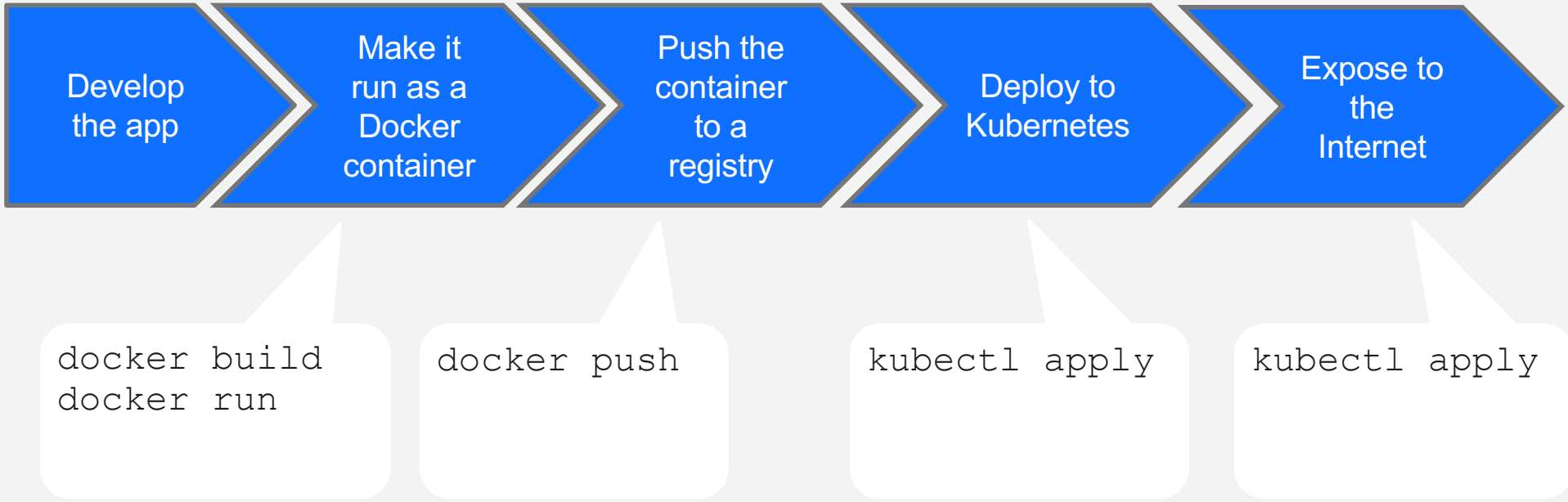
Lite • IBM

User Authentication and User Profiles for your apps.

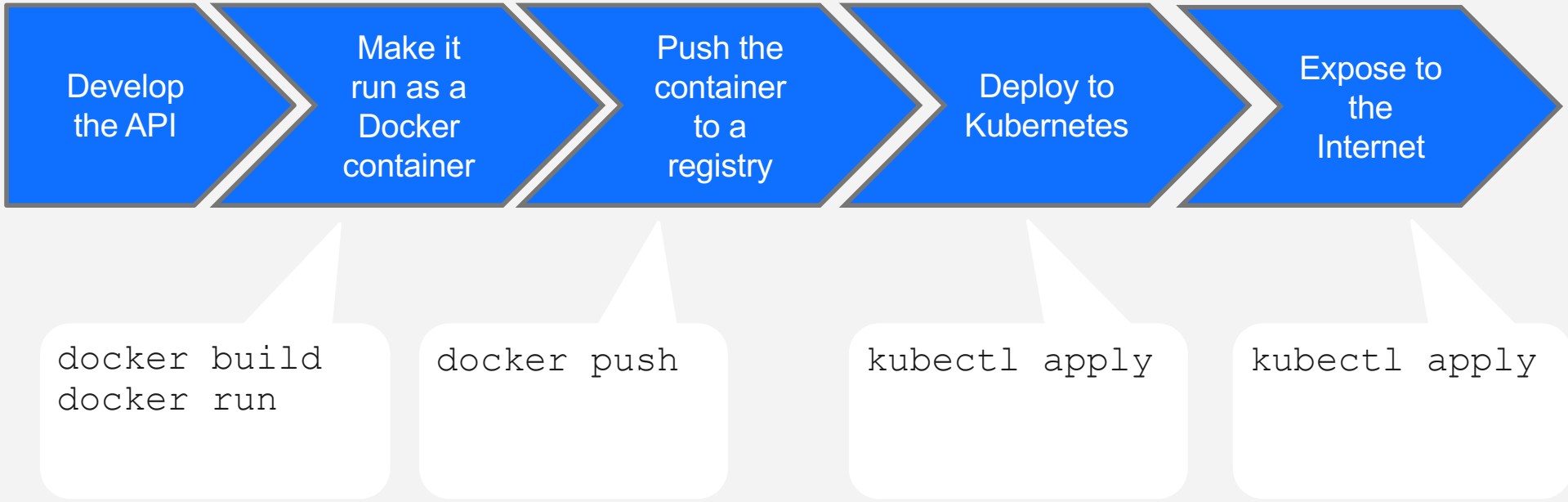
https://console.bluemix.net/docs/containers/cs_ingress.html#ingress
https://console.bluemix.net/docs/containers/cs_annotations.html#appid-auth

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: my-test-ingress
  annotations:
    ingress.bluemix.net/rewrite-path: "serviceName=myshiny-service rewrite=/"
    ingress.bluemix.net/appid-auth: "bindSecret=binding-appid-test namespace=default
requestType=web serviceName=myshiny-service,web-service"
spec:
   tls:
   - hosts:
     - growsmarter-test.eu-de.containers.appdomain.cloud
     secretName: growsmarter-test
   rules:
   - host: growsmarter-test.eu-de.containers.appdomain.cloud
     http:
       paths:
       - path: /shiny/
         backend:
           serviceName: myshiny-service
           servicePort: 3838
       - path: /
         backend:
           serviceName: web-service
           servicePort: 80
```

Deploying a Shiny app



Deploying an API

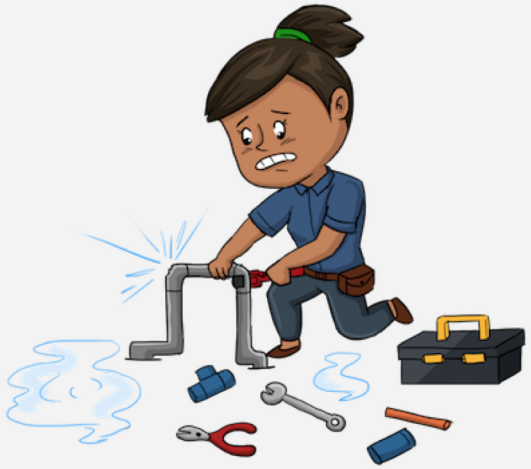


Building APIs with plumber

Step 1. Building the model

create-model.R

```
library(ggplot2) ## for dataset
library(readr) ## for write_rds
data(diamonds)

model <- lm(price ~ carat, data = diamonds)

write_rds(model, "lm-model.rds")
```



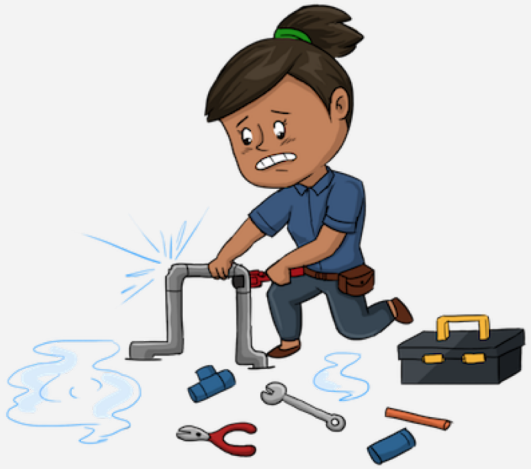
https://www.rplumber.io/

Building APIs with plumber

Step 2. Do scoring

score-lm.R

```
# read model
model <- readRDS("lm-model.rds")

#* Score using linear model
#* @param carat The carat value of the diamond
#* @get /score-lm
function(carat) {
  list(price = predict(model,
     newdata = data.frame(
       carat = as.numeric(carat))))
}
```



https://www.rplumber.io/

Deploying

Base image

Default port 8000

Can customize

Build model in Dockerfile or separately?

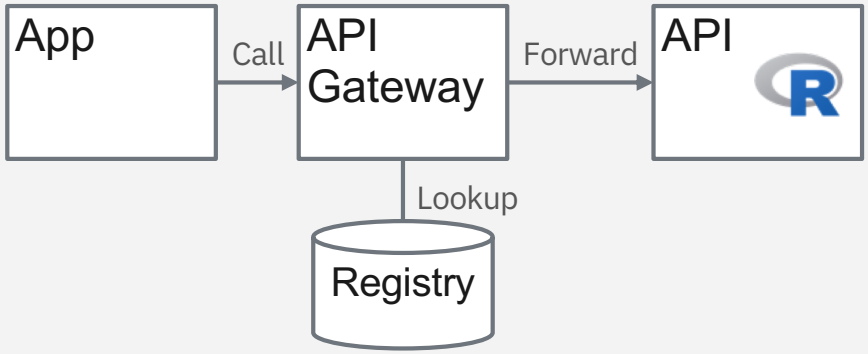
```
FROM trestletech/plumber
WORKDIR /R/
COPY . .
EXPOSE 8000

RUN ["install2.r", "ggplot2", "readr"]
RUN ["R", "-f", "/R/create-model.R"]

CMD ["/R/score-lm.R"]
```


```
$ docker build . -t score-lm

$ docker run -p 8000:8000 score-lm

$ curl http://localhost:8000/score-lm?carat=0.5
{"price":[1621.8522]}
```

https://hub.docker.com/r/trestletech/plumber/

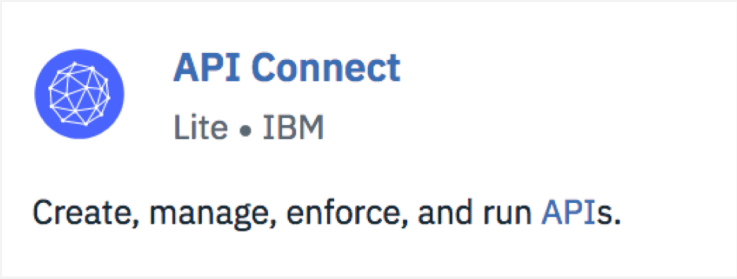
Securing APIs

1. Protect by network
2. Simple API key
3. API Gateway



Considerations:

- Public-facing APIs
- Brute force attacks
- Denial of service attacks

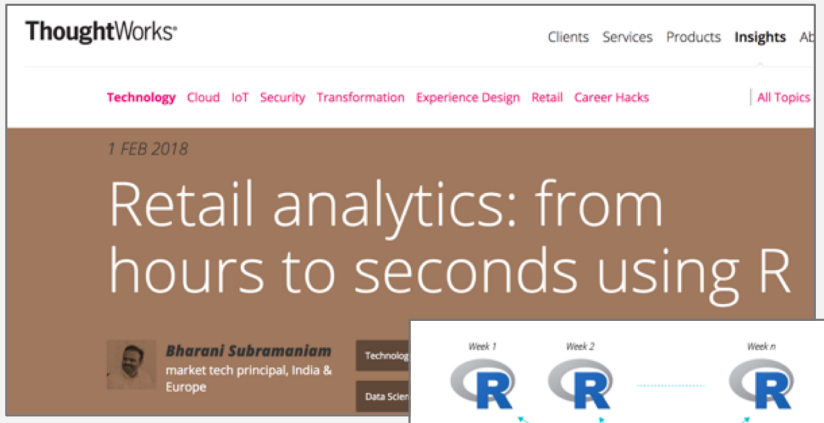


Is R slow?

Compared to what?

What is too slow?

Several things we can do



Joe Cheng, CTO RStudio:
"*R can be slow, and is single-threaded –*
But it's probably less slow than you think"

Timing the Same Algorithm in R, Python, and C++

While developing the `RcppDynProg` `R` package I took a little extra time to port the core algorithm from `C++` to both `R` and `Python`.

https://www.thoughtworks.com/insights/blog/retail-analytics-hours-seconds-using-r
http://www.win-vector.com/blog/2019/01/timing-of-the-same-algorithm-in-r-python-and-c/

Write efficient code	Leverage multi-core	Scale with multiple instances	Integrate other technologies
Vectorize!	parallel foreach doParallel	Kubernetes – load-balanced pods	
data.table	ranger	Node scaling	
Profile to find bottlenecks	caret	Map - Reduce	
RCpp	future / promises		
	ipc		

Resources

My tutorial: Deploying Shiny Apps on IBM Cloud
https://github.com/holken1/deploying-r-on-cloud/tree/master/shiny-on-ibm-cloud

- Free IBM Cloud cluster
- IBM Container Registry

My sample code:
https://github.com/holken1/deploying-r-on-cloud

IBM Cloud: https://cloud.ibm.com

RStudio: Load testing Shiny Applications
https://rstudio.github.io/shinyloadtest/

Joe Cheng: Shiny in Production (17 Jan)
https://speakerdeck.com/jcheng5/shiny-in-production

Colin Fay: An Introduction to Docker for R Users
https://colinfay.me/docker-r-reproducibility/

Yes, we can deploy R code.

It’s easy!

Secure!

Scalable!

IBM