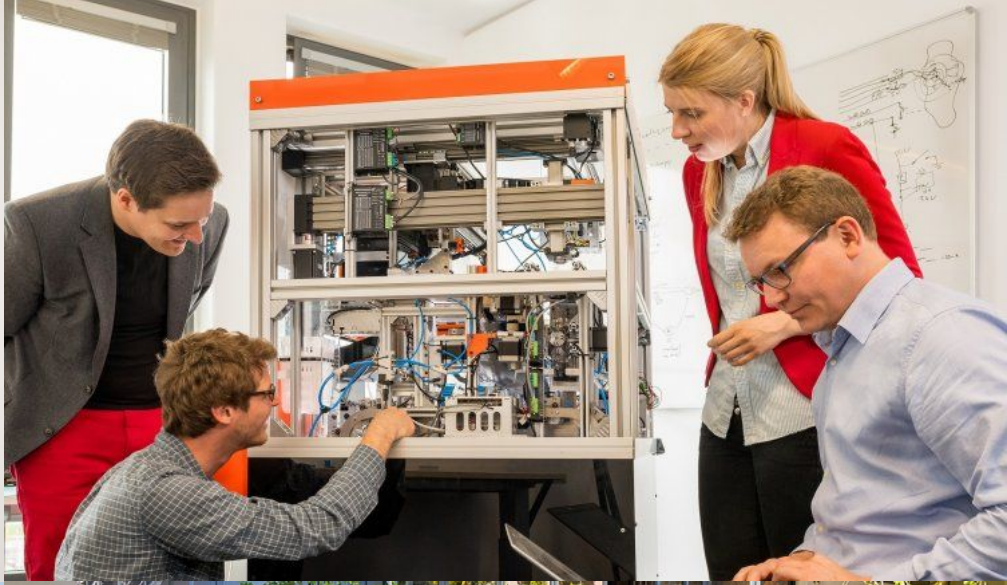
[devooops]: Cloud - native design - patterns

Maciej Lasyk

[devooops]: Cloud - native design - patterns

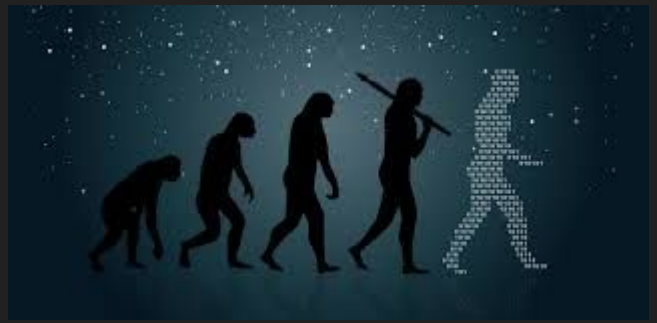
Maciej Lasyk


inveox
rethinking pathology



Chapter 01:

My evolution to “cloud native”



Chapter 02:

deployment models walkthrough

the given: business idea

Your app on bare metal / onprem

- Why?
- Are you a cloud provider?
- Are you a platform vendor?
- Do you sell IAAS software?
- ....
- Do it only when you have a good plan for the future and simple infra (KISS)!

Your app on cloud provider / IAAS

- That’s the way! But...
- Click, click in UI?
- IAAC: Terraform? Cloudformation? other?
- Learning curve!
- How do we even deploy?
- Cloud - agnostic? Technology lock-in?
- (always use simple, replaceable services)
- VPNs? Networking?
- Security policies? IAMs?
- GCP/AWS Organizations, projects, accounts?

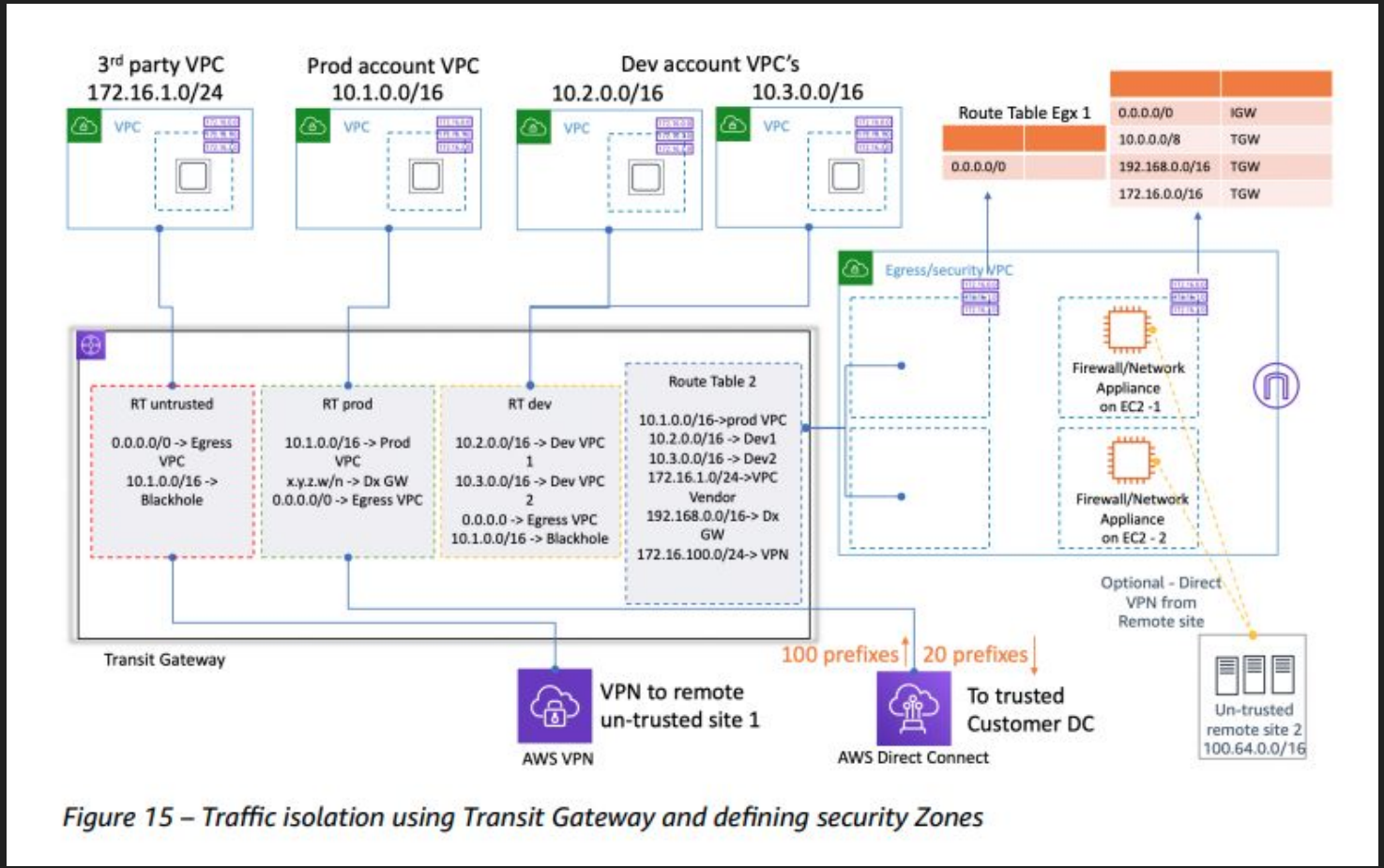
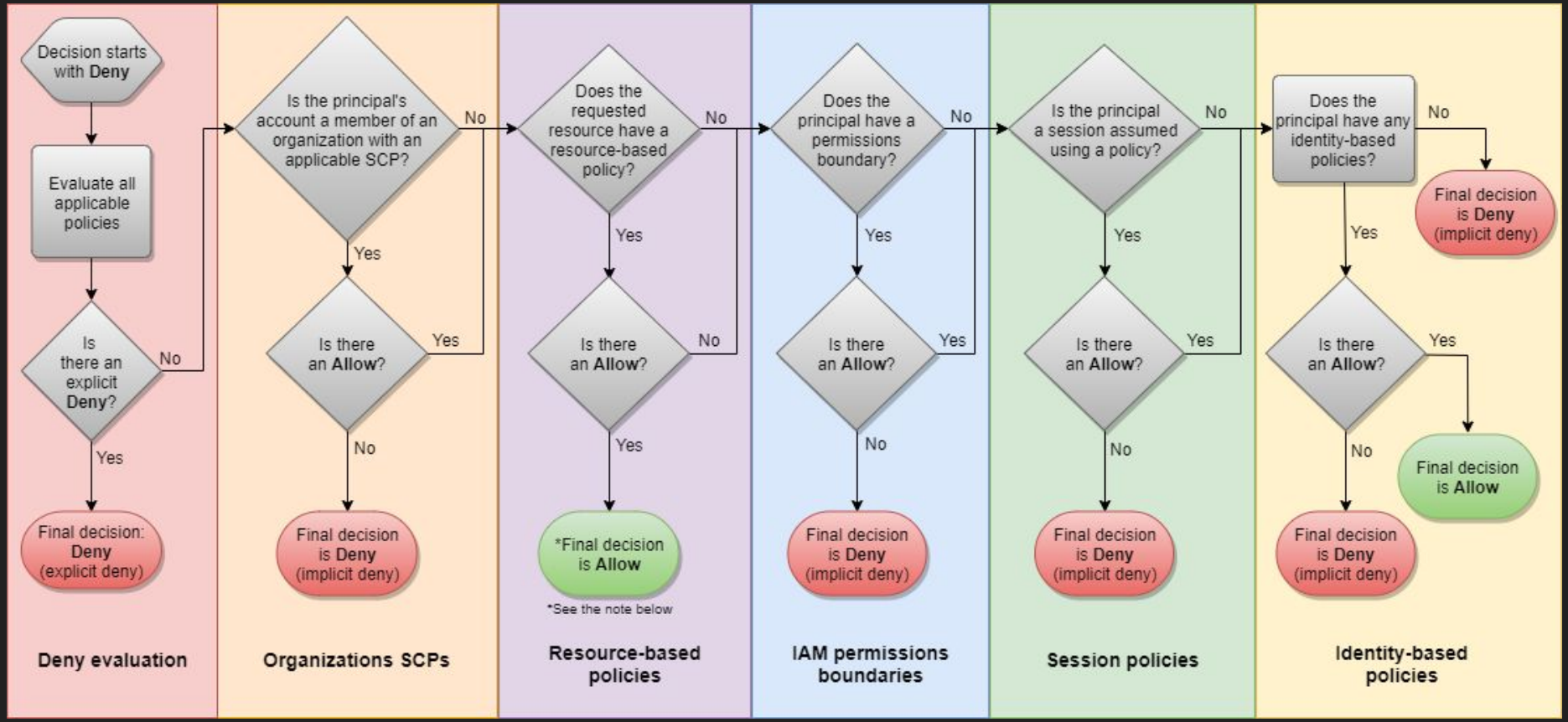
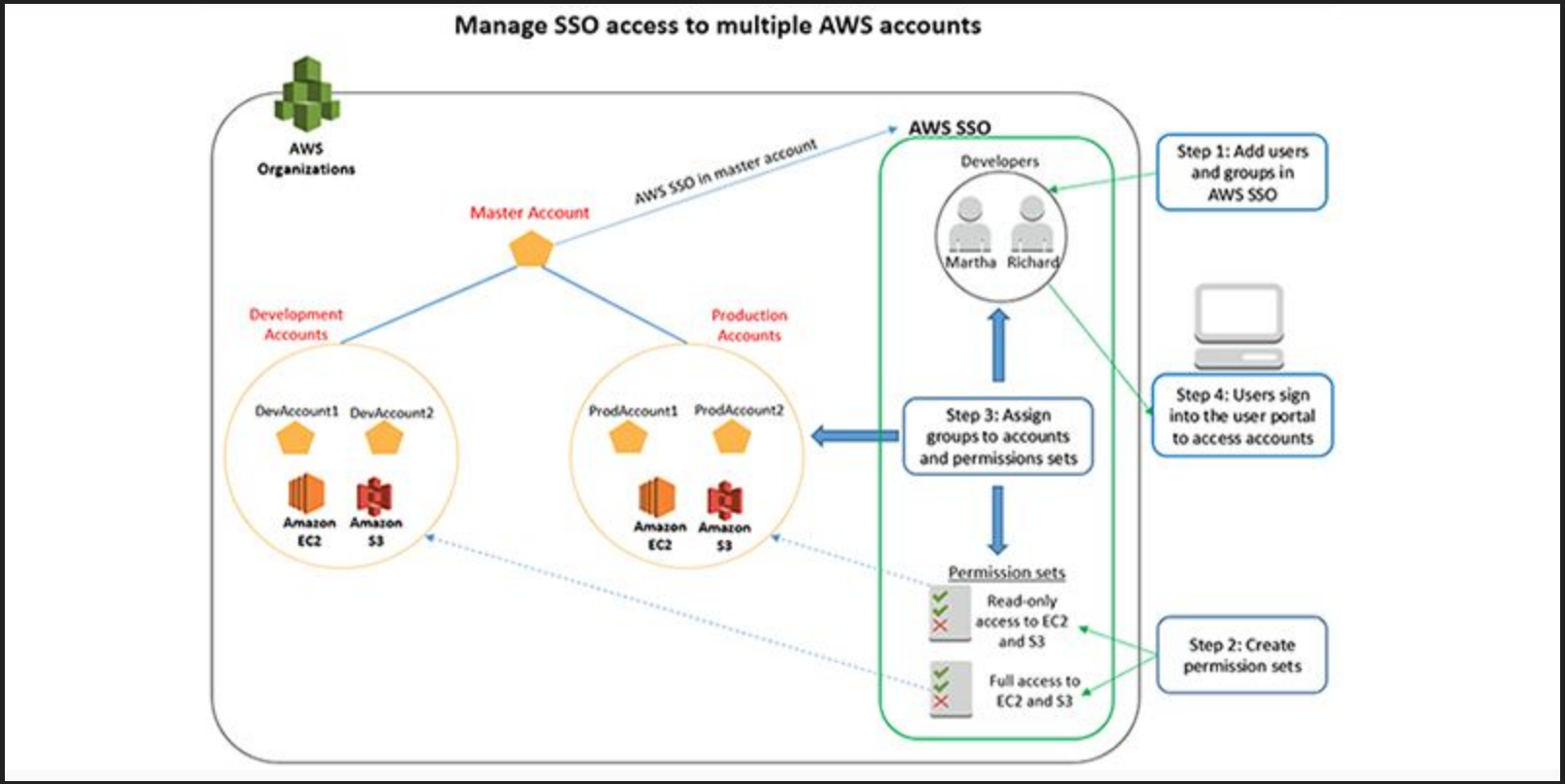


Figure 15 – Traffic isolation using Transit Gateway and defining security Zones

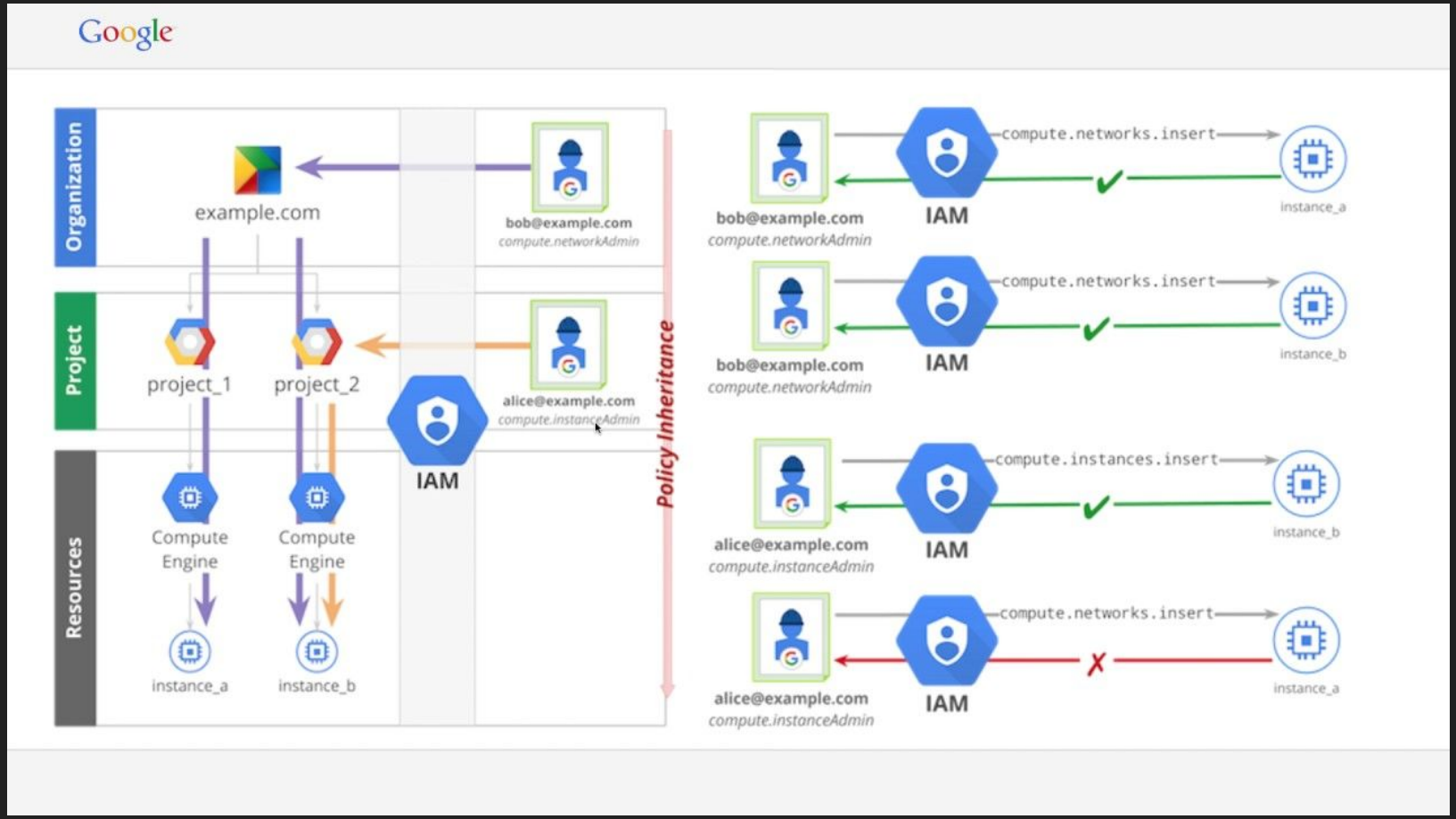
https://d1.awsstatic.com/whitepapers/building-a-scalable-and-secure-multi-vpc-aws-network-infrastructure.pdf



https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_evaluation-logic.html

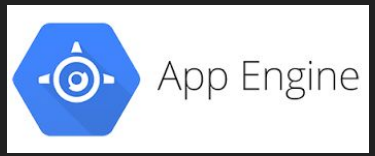


https://aws.amazon.com/blogs/security/tag/aws-organizations/

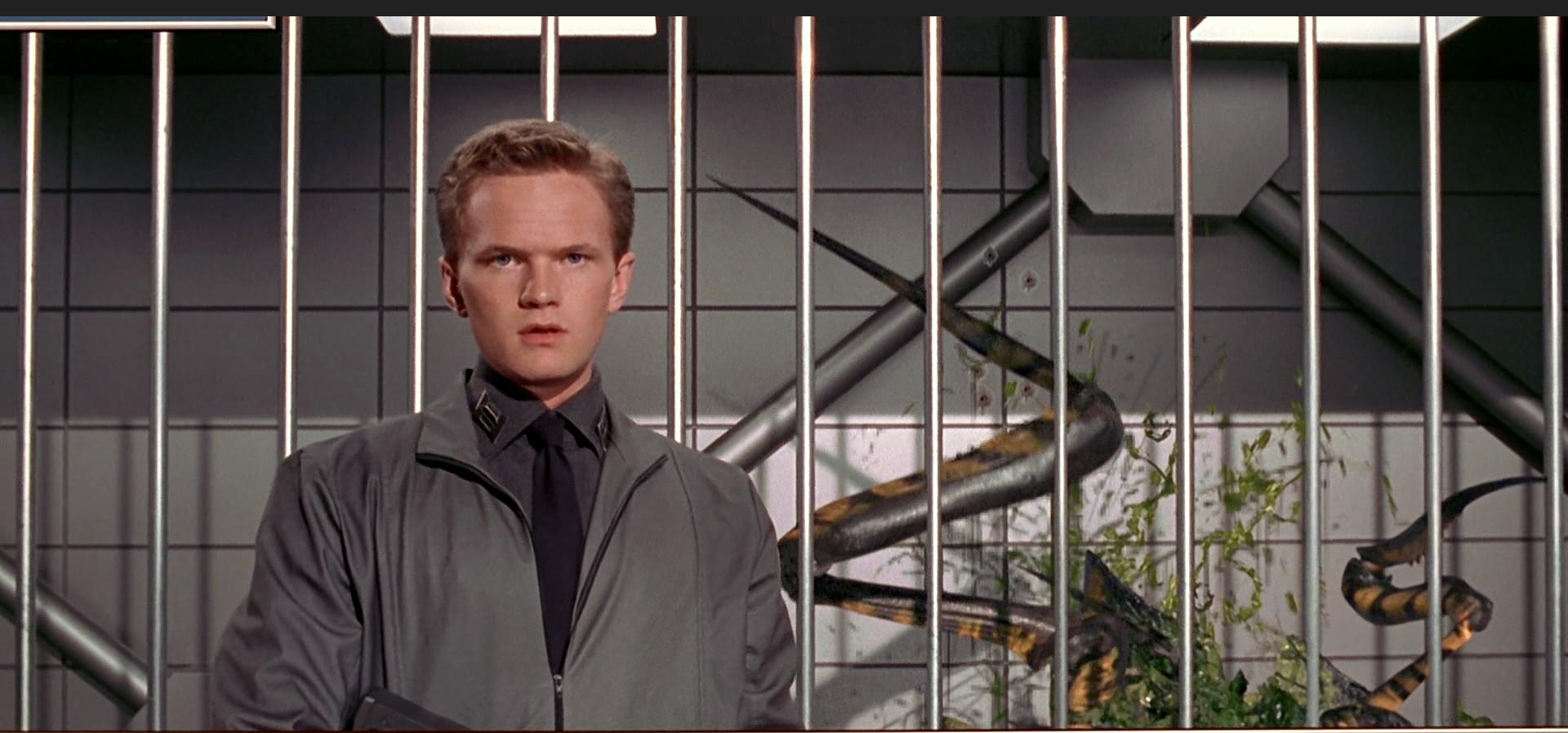


https://cloud.google.com/iam/docs/resource-hierarchy-access-control

PAAS to the rescue aka “I just want to deploy it”! aka “The evolution way”



- PAAS: pack your app, ship it, repeat
- When time is of the essence
- When productivity counts
- Even start w/monolith and learn your way
- Draft on paper / whiteboard
- Architecture workshops?
- Use monorepo?
- Github actions is easier than Jenkins(!)
- Datadog/New Relic/something + Pagerduty
- Monitor costs and performance(!)
- Learn SRE from the very beginning(!)


WOULD YOU LIKE TO KNOW MORE?

premature optimization

(aka: “when to get out of Heroku?)

- Is your productivity OK?
- Do you have spare cycles to work on your infra?
- Aren’t you just escaping tech - debt?
- Do you know exactly what you pay for?
- Can you scale up and side?
- “10 things against PAAS” meeting
- Do the math: How much time and ppl will it cost to migrate to IAAS/k8s?

Chapter 03:

Cloud Native

Wait, what?

Cloud native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. Containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach.

These techniques enable loosely coupled systems that are resilient, manageable, and observable. Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil.

The Cloud Native Computing Foundation seeks to drive adoption of this paradigm by fostering and sustaining an ecosystem of open source, vendor-neutral projects. We democratize state-of-the-art patterns to make these innovations accessible for everyone.

https://github.com/cncf/toc/blob/master/DEFINITION.md

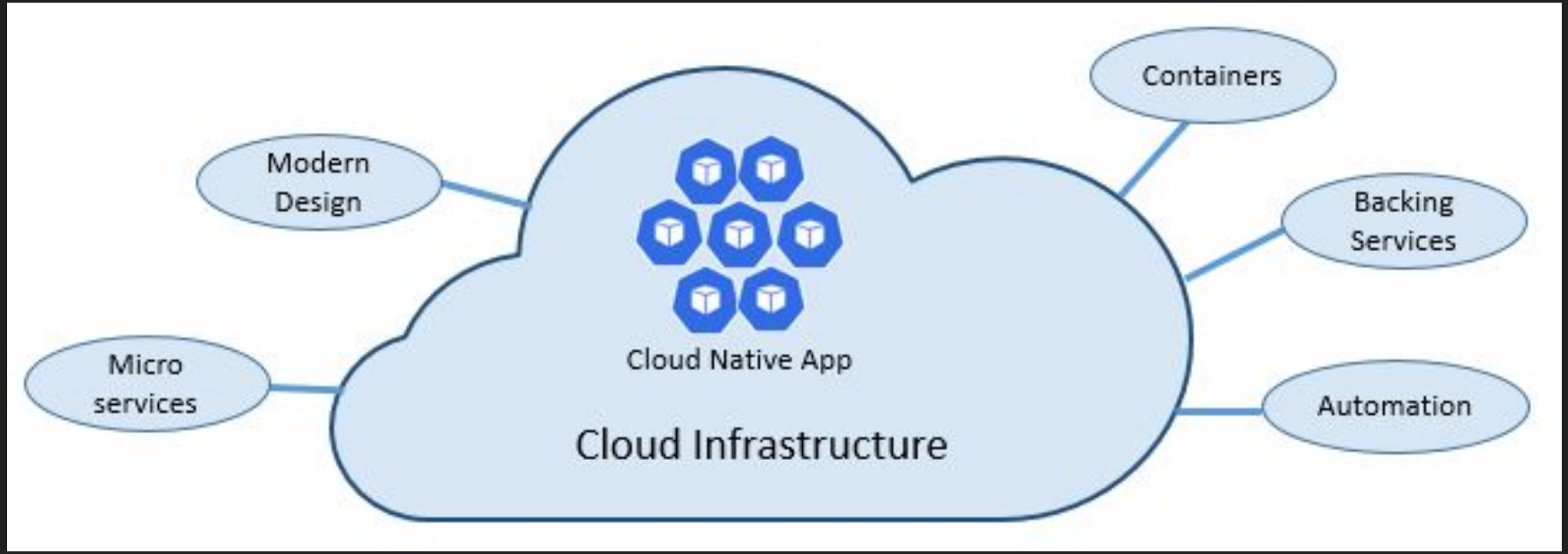
云原生技术有利于各组织在公有云、私有云和混合云等新型动态环境中，构建和运行可弹性扩展的应用。云原生的代表技术包括容器、服务网格、微服务、不可变基础设施和声明式API。

这些技术能够构建容错性好、易于管理和便于观察的松耦合系统。结合可靠的自动化手段，云原生技术使工程师能够轻松地对系统作出频繁和可预测的重大变更。

云原生计算基金会（CNCF）致力于培育和维护一个厂商中立的开源生态系统，来推广云原生技术。我们通过将最前沿的模式民主化，让这些创新为大众所用。

https://github.com/cncf/toc/blob/master/DEFINITION.md

Cloud native the hard way



https://docs.microsoft.com/en-us/dotnet/architecture/cloud-native/definition

Cloud native the hard way

The Twelve-Factor apps

1. Codebase
2. Dependencies
3. Config
4. Backing services
5. Build, release, run
6. Processes
7. Port binding
8. Concurrency
9. Disposability
10. Dev/prod parity
11. Logs
12. Admin processes

https://12factor.net/

Cloud native the hard way

Tough questions

1. Communication
2. Resiliency
3. Distributed Data
4. Identity

Cloud native the hard way

Tough questions #2

1. Backing services? Which queues? Which DBs?
2. Automation? Which one? Why?
3. CI/CD? Who will work on it?
4. Who will stay on-call? How will teams interact with each other during outage?
5. What do you know about reliability?
6. What are we missing here?

Cloud native the hard way

Tough questions #2

1. Backing services? Which queues? Which DBs?
2. Automation? Which one? Why?
3. CI/CD? Who will work on it?
4. Who will stay on-call? How will teams interact with each other during outage?
5. What do you know about reliability?
6. What are we missing here? Security of course <3

Let’s use Kubernetes!

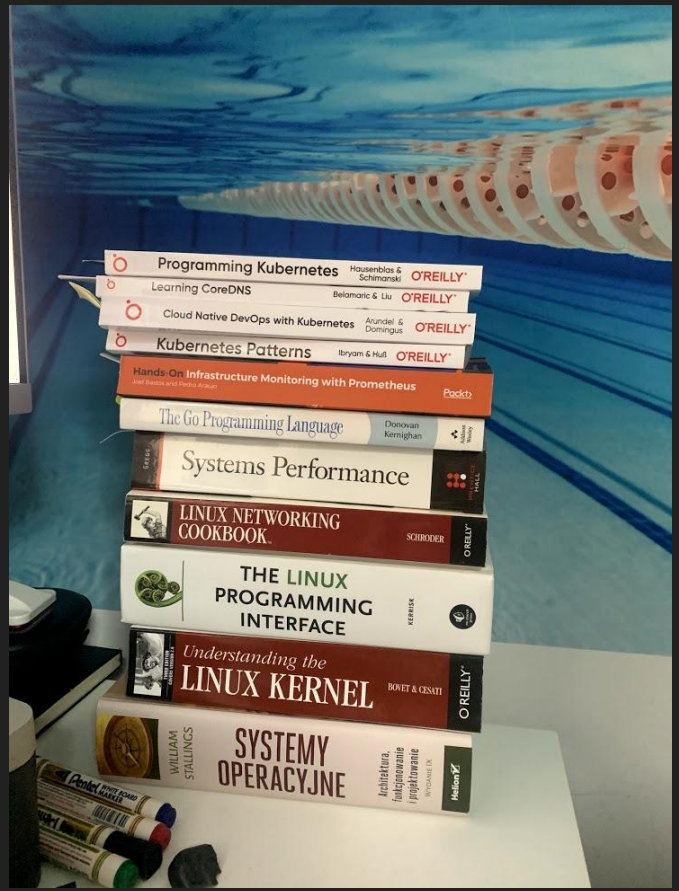
1. How big is your codebase? Does size_of_ur_code > k8s_yaml_size ?
2. Isn’t cloud VM autoscaling enough for you?
3. Do you have any infra experts in your team? And I mean “Experts”
4. Are there any Golang ppl in your company?

Let’s use Kubernetes!

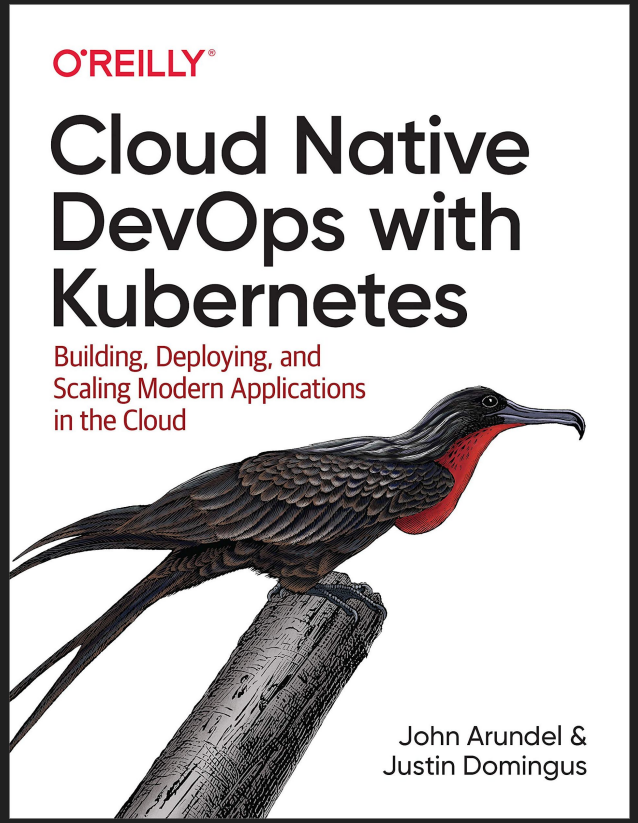
https://twitter.com/memenetes/status/1305612451443023876

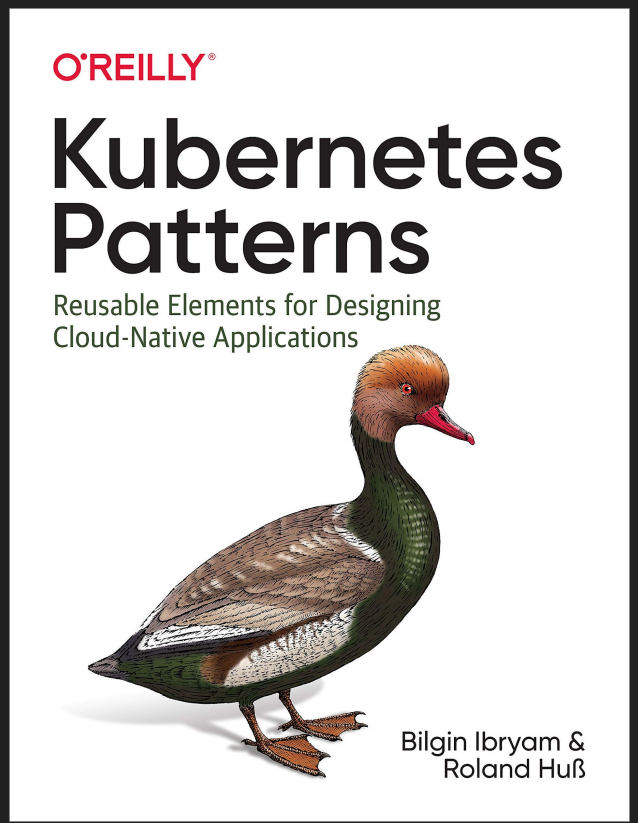
Let’s use Kubernetes!

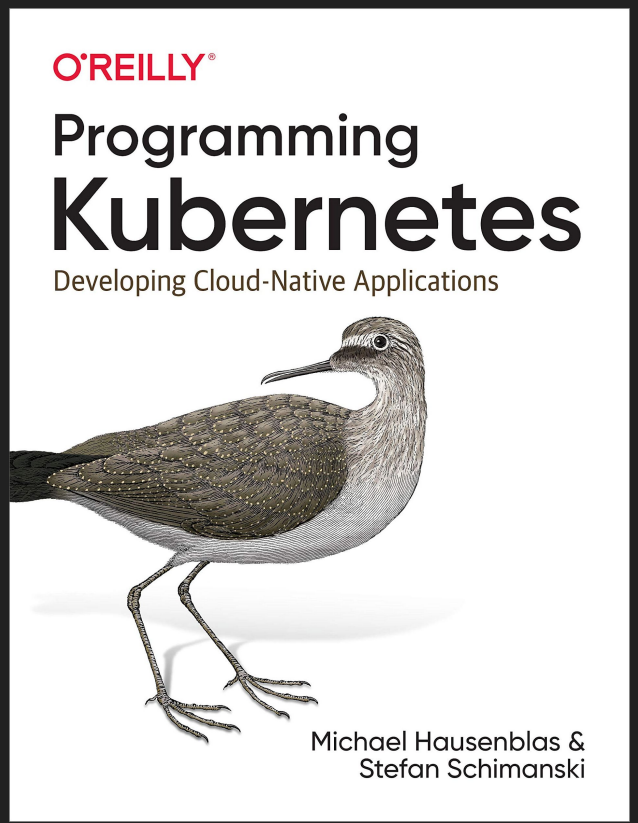
https://twitter.com/memenetes/status/1305612451443023876



Let’s learn Kubernetes!

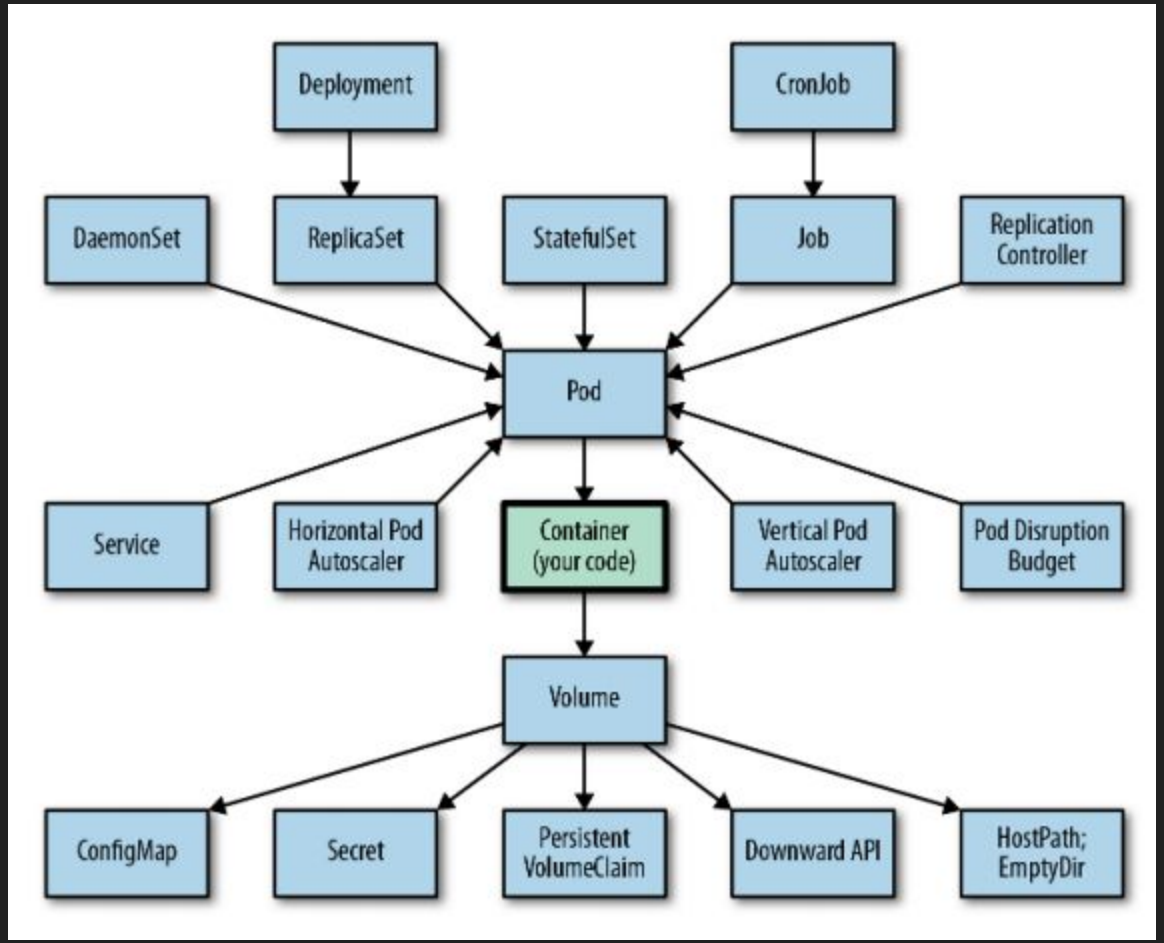






Chapter 04:

Cloud Native Design Patterns



Kubernetes patterns book

“common sense”

Problem:

How and where to deploy my app?

“common sense”

Problem:

How and where to deploy my app?

https://twitter.com/memenetes/status/1318296058305404928

“predictable demands”

Problem:

How much resources service needs and will get? and what if limits crossed?

“predictable demands”

Problem:

How much resources service needs and will get? and what if limits crossed?

Resource profiles:

- bare-metal: cgroups (OOM, throttle)
- paas: depends (restart app, throttle)
- iaas: VM size -> autoscaling (OOMs, VM will freeze)
- k8s: Pod QOS - best effort / burstable / guaranteed

“predictable demands”

Problem:

How much resources service needs and will get? and what if limits crossed?

Scheduler priorities:

- bare-metal: do you have any spare servers?
- paas / iaas: depends on availability in zone
- k8s: pod priority (move up in the queue, preempt/remove if no place)

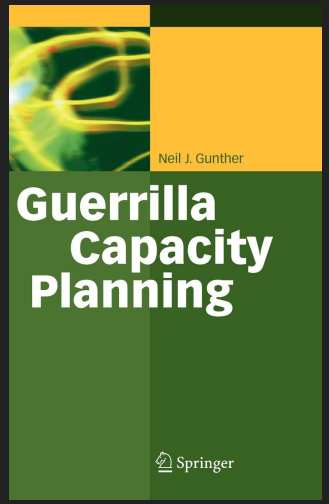
“predictable demands”

Problem:

How much resources service needs and will get? and what if limits crossed?

Conclusions?

- Capacity planning
- See Guerilla Capacity Planning book



“declarative deployment”

Problem:

What is the safe model of releasing your app/services changes?

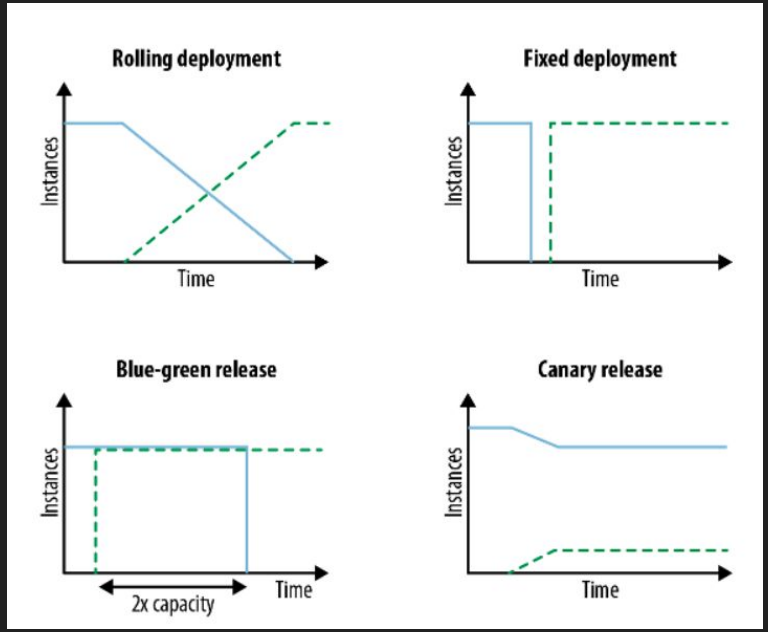
“declarative deployment”

Problem:

What is the safe model of releasing your app/services changes?

Deployment strategies:

- Rolling deployment
- Fixed deployment
- Blue / green
- Canary



Kubernetes patterns book

“declarative deployment”

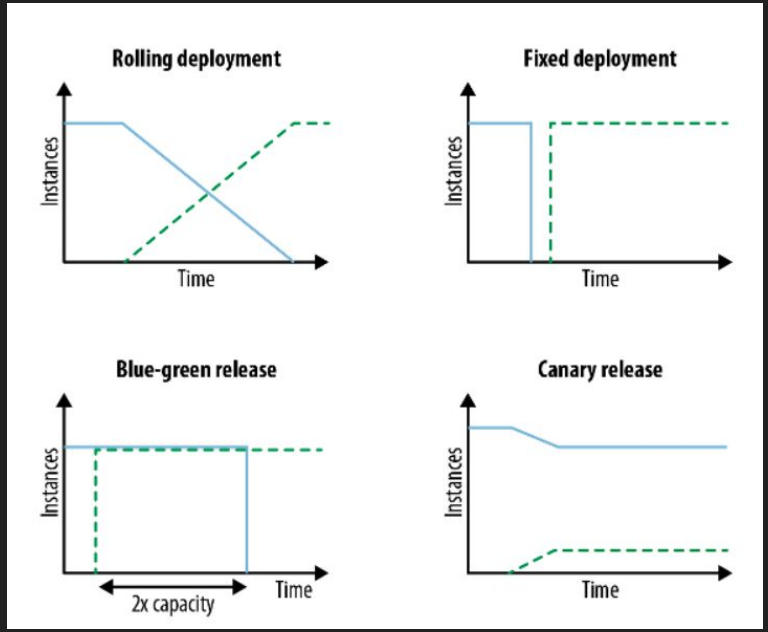
Problem:

What is the safe model of releasing your app/services changes?

Deployment strategies:

- Rolling deployment
- Fixed deployment
- Blue / green
- Canary

Those models are cloud-agnostic, but K8S has it built into Deployment definition



Kubernetes patterns book

“declarative deployment”

Problem:

What is the safe model of releasing your app/services changes?

Deployment strategies:

- Rolling deployment
- Fixed deployment
- Blue / green
- Canary

Those models are cloud-agnostic, but
K8S has some of it built into Deployment
Definition (but Blue/Green: see Istio)

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: random-generator
spec:
  replicas: 3
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
    selector:
      matchLabels:
        app: random-generator
```

Kubernetes patterns book

“healthchecking”

Problem:

How do we know service runs properly? What will happen if not?

“healthchecking”

Problem:

How do we know service runs properly? What will happen if not?

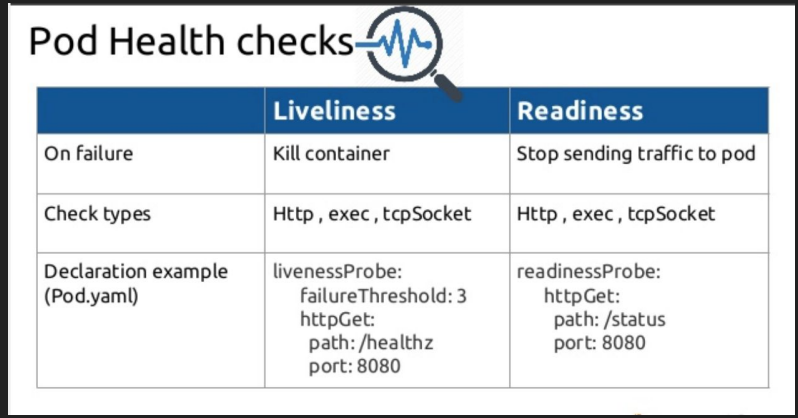
- Classical HTTP health checks

“healthchecking”

Problem:

How do we know service runs properly? What will happen if not?

- Classical HTTP health checks
- Liveness probes (k8s)
- Readiness probes (k8s)



	Liveliness	Readiness
On failure	Kill container	Stop sending traffic to pod
Check types	Http , exec , tcpSocket	Http , exec , tcpSocket
Declaration example (Pod.yaml)	livenessProbe: failureThreshold: 3 httpGet: path: /healthz port: 8080	readinessProbe: httpGet: path: /status port: 8080

https://www.weave.works/blog/resilient-apps-with-liveness-and-readiness-probes-in-kubernetes

“managed lifecycle”

Problem:

How can scheduler stop / start / kill / warmup application?

“managed lifecycle”

Problem:

How can scheduler stop / start / kill / warmup application?

- K8S:
  - SIGTERM
  - SIGKILL
  - Poststart hooks
  - Prestop hooks

“managed lifecycle”

Problem:

How can scheduler stop / start / kill / warmup application?

- K8S:
  - SIGTERM
  - SIGKILL
  - Poststart hooks
  - Prestop hooks
- Bare-metal / IAAS: Linux systemd can do a more specific job (but not a distributed one)
- PAAS: it’s up to provider

“placement”

Problem:

How are services placed across infrastructure?

“placement”

Problem:

How are services placed across infrastructure?

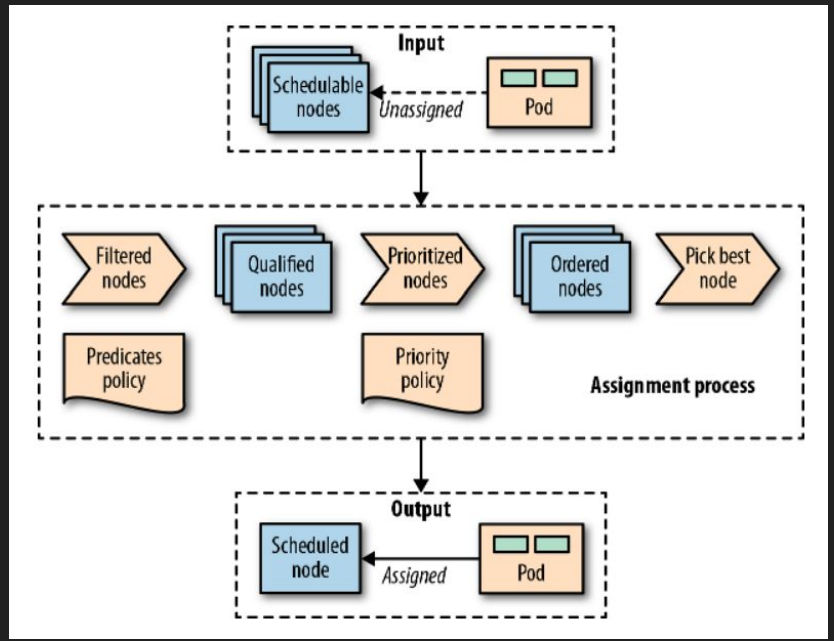
- Bare-metal: it’s up to us
- IAAS: use AZs / regions
- PAAS: most has it built-in HA, use AZs / regions

“placement”

Problem:

How are services placed across infrastructure?

```
{
    "kind" : "Policy",
    "apiVersion" : "v1",
    "predicates" : [                          ❶
        {"name" : "PodFitsHostPorts"},
        {"name" : "PodFitsResources"},
        {"name" : "NoDiskConflict"},
        {"name" : "NoVolumeZoneConflict"},
        {"name" : "MatchNodeSelector"},
        {"name" : "HostName"}
    ],
    "priorities" : [                          ❷
        {"name" : "LeastRequestedPriority", "weight" : 2},
        {"name" : "BalancedResourceAllocation", "weight" : 1},
        {"name" : "ServiceSpreadingPriority", "weight" : 2},
        {"name" : "EqualPriority", "weight" : 1}
    ]
}
```



Kubernetes patterns book

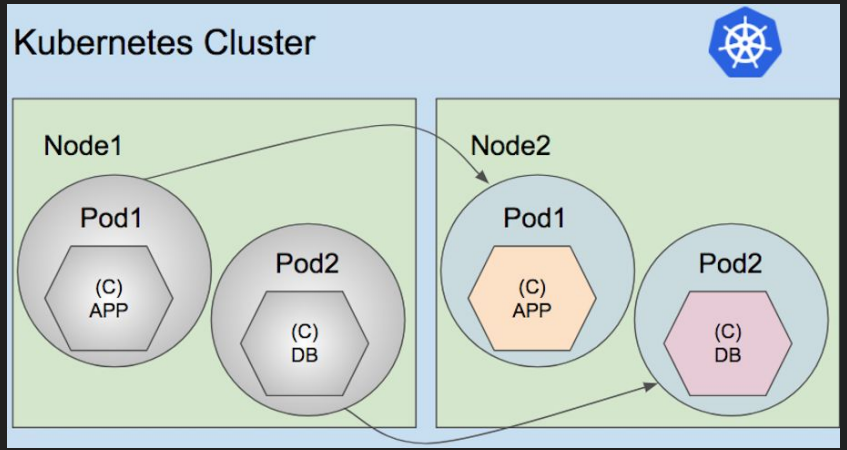
“placement”

Problem:

How are services placed across infrastructure?

K8S:

- Node affinity (labels)
- Node anti-affinity (labels)
- Pod affinity (labels)
- Pod anti-affinity (labels)
- Taints / tolerations (node definition)
- Custom scheduler



https://cloudify.co/blog/migrating-pods-containerized-applications-nodes-kubernetes-cluster-using-cloudify/

Cloud Native Design Patterns

Behavioral patterns

“batch job”

Problem:

How can we schedule a batch job in a cloud-native way?

“batch job”

Problem:

How can we schedule a batch job in a cloud-native way?

- Bare-metal: run a script (not a distributed thing)
- IAAS: run oneshot script/VM or use some cloud scheduler like AWS Batch
- PAAS: use some cloud scheduler like AWS Batch

“batch job”

Problem:

How can we schedule a batch job in a cloud-native way?

K8S:

```yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: random-generator
spec:
  completions: 5
  parallelism: 2
  template:
    metadata:
      name: random-generator
    spec:
      restartPolicy: OnFailure
```

Kubernetes patterns book

“periodic job”

Problem:

How can we schedule a periodic job?

“periodic job”

Problem:

How can we schedule a periodic job?

- Bare-metal: systemd-timer, cronjob
- IAAS: batch features (events, lambdas/functions, K8S)
- PAAS: crons

“periodic job”

Problem:

How can we schedule a periodic job?

K8S:

```yaml
apiVersion: batch/v1beta1
kind: CronJob
metadata:
  name: random-generator
spec:
  # Every three minutes
  schedule: "*/3 * * * *"
```

Kubernetes patterns book

“daemon service”

Problem:

How can we run a daemon?

“daemon service”

Problem:

How can we run a daemon?

- Bare metal: systemd simple service (or fork); thus not distributed
- IAAS: depends; probably also systemd service in VM
- PAAS: e.g. via extensions (Beanstalk ebextensions)

“daemon service”

Problem:

How can we run a daemon?

K8S:

```yaml
apiVersion: extensions/v1beta1
kind: DaemonSet
metadata:
  name: random-refresher
spec:
  selector:
    matchLabels:
      app: random-refresher
  template:
    metadata:
      labels:
        app: random-refresher
    spec:
      nodeSelector:
        feature: hw-rng
```

Kubernetes patterns book

“singleton service”

Problem:

How can we make sure exactly one instance of service is running?

“singleton service”

Problem:

How can we make sure exactly one instance of service is running?

Bare-metal/IAAS:

- Out-application locking: 3rd party service / scheduler
- In-application locking

“singleton service”

Problem:

How can we make sure exactly one instance of service is running?

Bare-metal/IAAS:

- Out-application locking: 3rd party service / scheduler
- In-application locking

K8S:

- Out-application locking: exactly 1 pod replica
- In-application locking: many replicas, app takes care of the rest

“stateful service”

Problem:

How can we run a stateful service?

“stateful service”

Problem:

How can we run a stateful service?

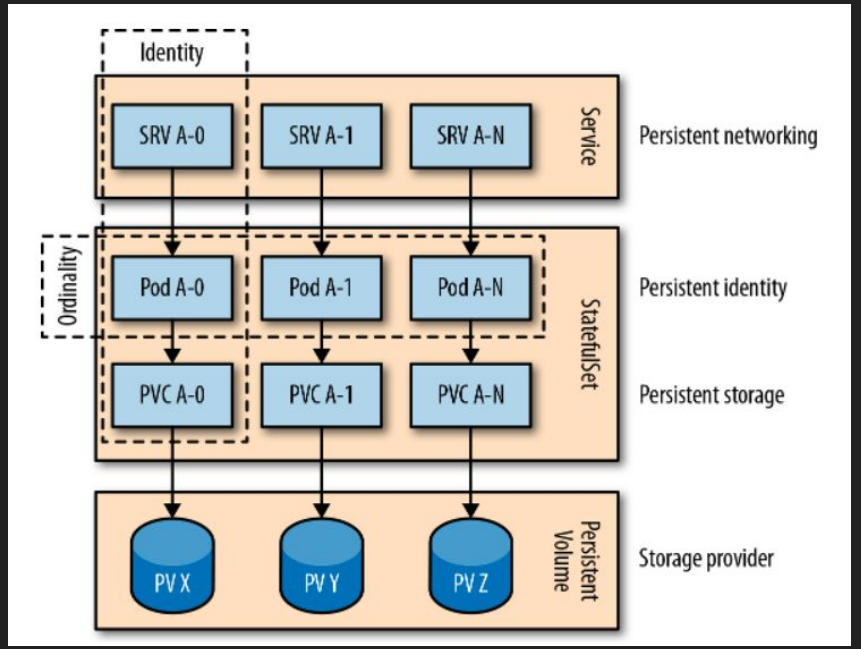
- Bare-metal: automation
- IAAS: automation
- PAAS: configuration

“stateful service”

Problem:

How can we run a stateful service?

- K8S: StatefulSet
  - Storage
  - Networking
  - Identity
  - Ordinality



Kubernetes patterns book

“stateful service”

Problem:

How can we run a stateful service?

- K8S: StatefulSet
  - Storage
  - Networking
  - Identity
  - Ordinality

```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: rg
spec:
  serviceName: random-generator
  replicas: 2
  selector:
    matchLabels:
      app: random-generator
```

Kubernetes patterns book

“service discovery”

Problem:

How can we send a request to a particular service?

“service discovery”

Problem:

How can we address sending a request to a particular service?

- Bare-metal: service-discovery software + LB (Consul, Haproxy..)
- IAAS: cloud-provided service discovery, LB target groups
- PAAS: depends, e.g. Heroku private spaces + DNS

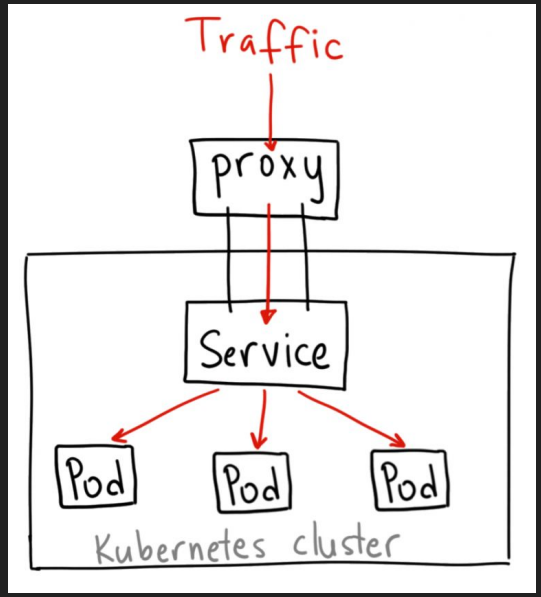
“service discovery”

Problem:

How can we address sending a request to a particular service?

K8S:

- Internal: ClusterIp (w/DNS name)
- Internal: NodePort
- External: LoadBalancer
- External / Internal: Ingress
- External: Metallb in L2/BGP/ARP



https://rtfm.co.ua/en/kubernetes-clusterip-vs-nodeport-vs-loadbalancer-services-and-ingress-an-overview-with-examples/

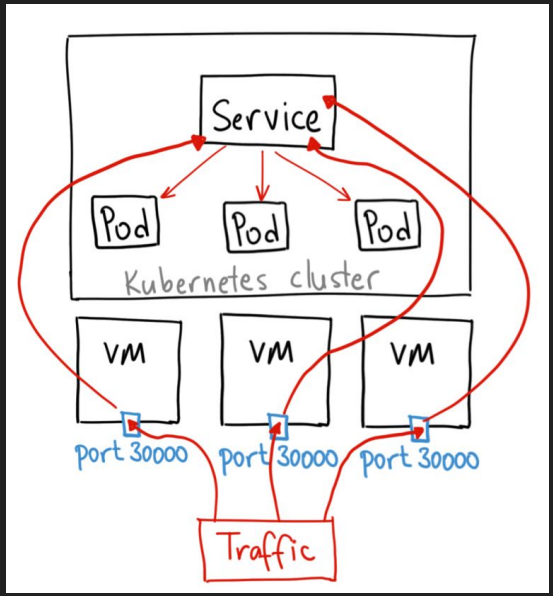
“service discovery”

Problem:

How can we address sending a request to a particular service?

K8S:

- Internal: ClusterIp (w/DNS name)
- Internal: NodePort
- External: LoadBalancer
- External / Internal: Ingress
- External: Metallb in L2/BGP/ARP



https://rtfm.co.ua/en/kubernetes-clusterip-vs-nodeport-vs-loadbalancer-services-and-ingress-an-overview-with-examples/

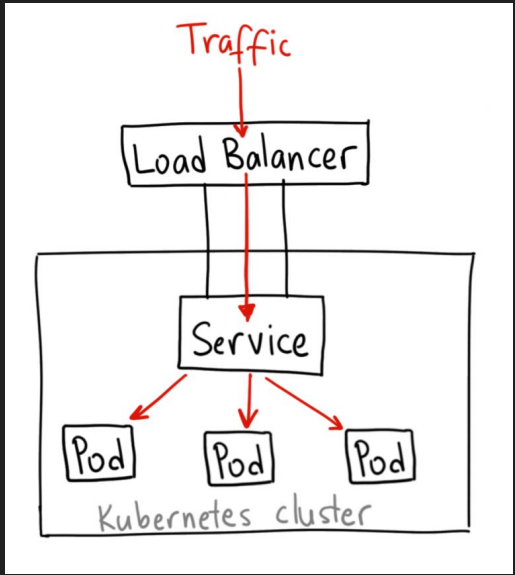
“service discovery”

Problem:

How can we address sending a request to a particular service?

K8S:

- Internal: ClusterIp (w/DNS name)
- Internal: NodePort
- External: LoadBalancer
- External / Internal: Ingress
- External: Metallb in L2/BGP/ARP



https://rtfm.co.ua/en/kubernetes-clusterip-vs-nodeport-vs-loadbalancer-services-and-ingress-an-overview-with-examples/

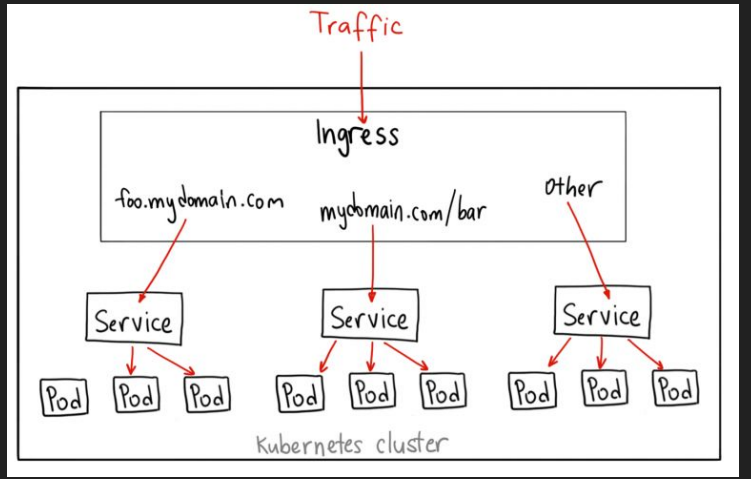
“service discovery”

Problem:

How can we address sending a request to a particular service?

K8S:

- Internal: ClusterIp (w/DNS name)
- Internal: NodePort
- External: LoadBalancer
- External / Internal: Ingress
- External: Metallb in L2/BGP/ARP



https://rtfm.co.ua/en/kubernetes-clusterip-vs-nodeport-vs-loadbalancer-services-and-ingress-an-overview-with-examples/

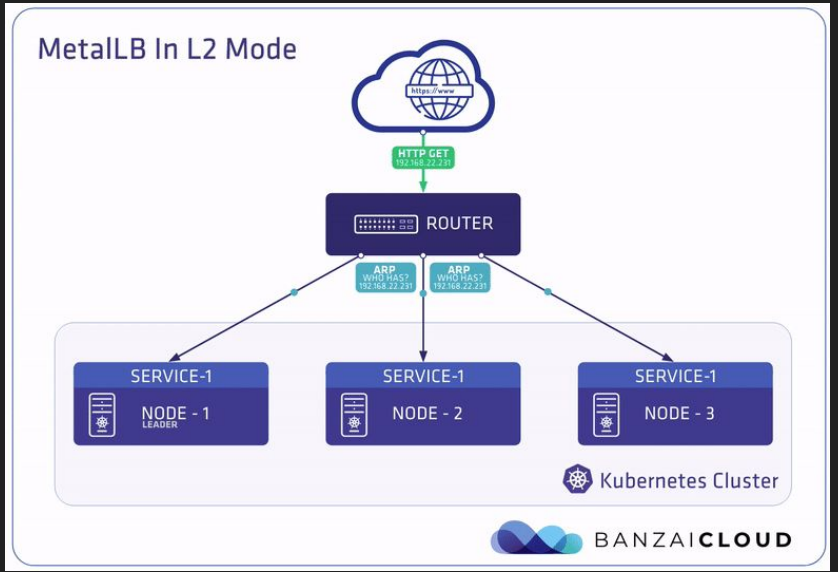
“service discovery”

Problem:

How can we address sending a request to a particular service?

K8S:

- Internal: ClusterIp (w/DNS name)
- Internal: NodePort
- External: LoadBalancer
- External / Internal: Ingress
- External: Metallb in L2/BGP/ARP



https://banzaicloud.com/blog/load-balancing-on-prem/

“self awareness”

Problem:

How can service get environment - specific variables (IP, hostname etc)?

“self awareness”

Problem:

How can service get environment - specific variables (IP, hostname etc)?

- Bare-metal: OS function, e.g. socket.gethostname()
- IAAS: cloud endpoint (e.g. AWS http://169.254.169.254/latest/meta-data/)
- PAAS: look in environment variables

“self awareness”

Problem:

How can service get environment - specific variables (IP, hostname etc)?

K8S:

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: random-generator
spec:
  containers:
  - image: k8spatterns/random-gener
    name: random-generator
    env:
    - name: POD_IP
      valueFrom:
        fieldRef:
          fieldPath: status.podIP
```

Kubernetes patterns book

https://kubernetes.io/docs/tasks/inject-data-application/environment-variable-expose-pod-information/

Cloud Native Design Patterns

Structural patterns

“initialization”

Problem:

How can service have all prerequisites fulfilled before starting up?

“initialization”

Problem:

How can service have all prerequisites fulfilled before starting up?

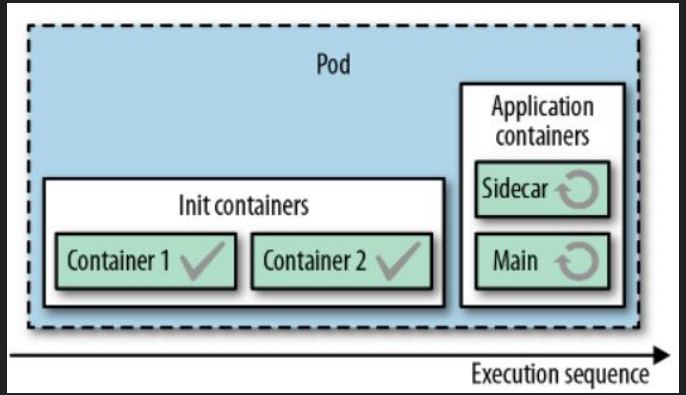
- Bare-metal: e.g. systemd PreStart
- IAAS: VM startup script (e.g. EC2 user data)
- PAAS: use extensions (e.g. Beanstalk ebextensions)

“initialization”

Problem:

How can service have all prerequisites fulfilled before starting up?

- Bare-metal: e.g. systemd PreStart
- IAAS: VM startup script (e.g. EC2 user data)
- PAAS: use extensions (e.g. Beanstalk ebextensions)
- K8S: an Init Container



Kubernetes patterns book

“sidecar”

Problem:

How to extend service functionality?

“sidecar”

Problem:

How to extend service functionality?

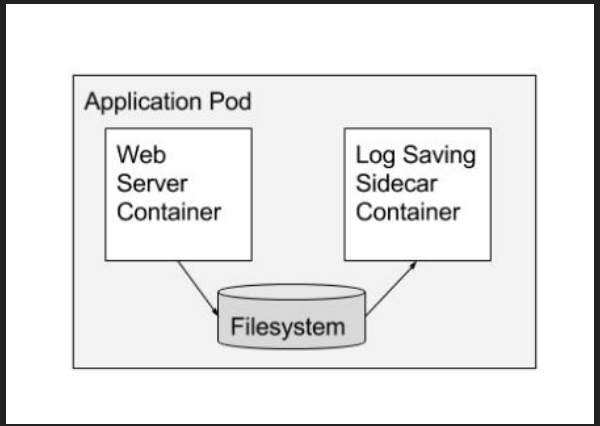
- Bare-metal: run new process in machine
- IAAS: also run a new process inside of VM
- PAAS: use extensions (e.g. Beanstalk ebextensions / services dependencies)

“sidecar”

Problem:

How to extend service functionality?

- Bare-metal: run new process in machine
- IAAS: also run a new process inside of VM
- PAAS: use extensions (e.g. Beanstalk ebextensions / services dependencies)
- K8S: a sidecar container



“adapter”

Problem:

How to work in heterogeneous environment?

“adapter”

Problem:

How to work in heterogeneous environment?

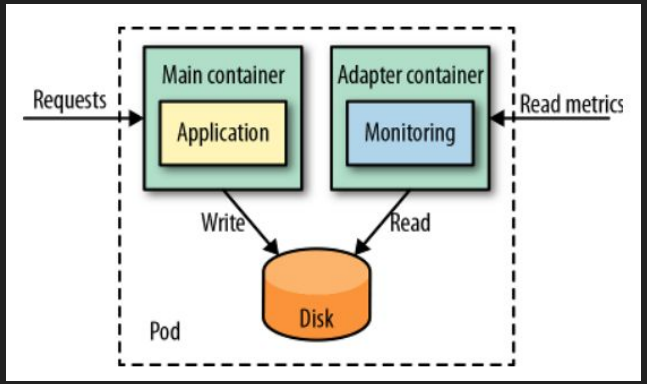
- Bare-metal: an adapter/translation service
- IAAS: an adapter/translation service inside of VM
- PAAS: should be already unified in PAAS monitoring service (e.g. via CloudWatch metrics)

“adapter”

Problem:

How to work in heterogeneous environment?

- Bare-metal: an adapter/translation service
- IAAS: an adapter/translation service inside of VM
- PAAS: should be already unified in PAAS monitoring service (e.g. via CloudWatch metrics)
- K8S: an adapter container



Kubernetes patterns book

“ambassador”

Problem:

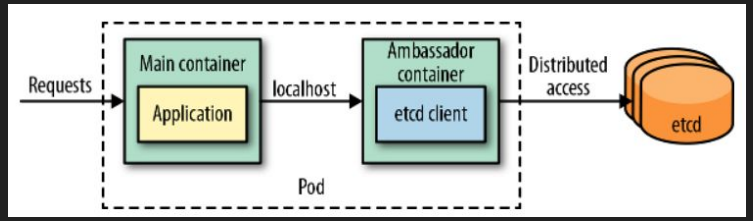
How to decouple access to an external service?

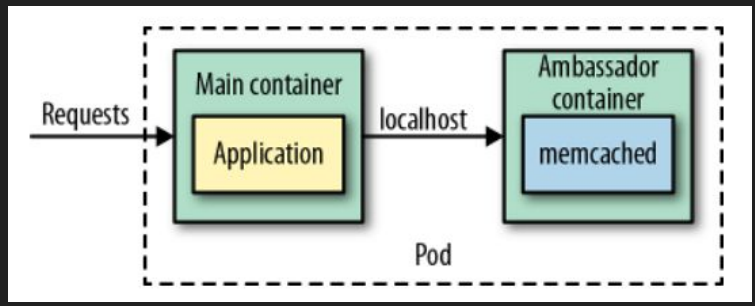
“ambassador”

Problem:

How to decouple access to an external service?

- K8S: an ambassador container





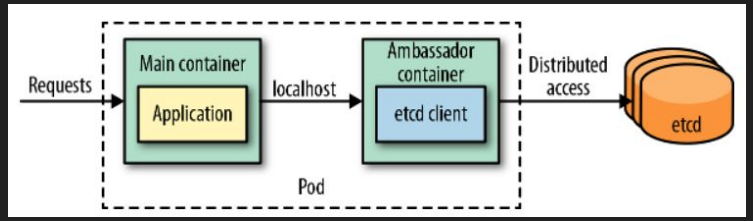
Kubernetes patterns book

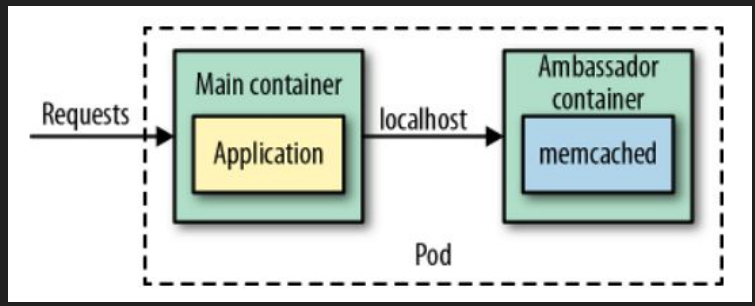
“ambassador”

Problem:

How to decouple access to an external service?

- K8S: an ambassador container
- Bare-metal: a proxy/adapter service
- IAAS: a proxy/adapter service
- PAAS: it might be tricky, probably additional service





Kubernetes patterns book

Cloud Native Design Patterns

Configuration patterns

“Environment configuration”

Problem:

How to set environment configurations?

“Environment configuration”

Problem:

How to set environment configurations?

- Bare-metal: base image, automation + env vars (watch out for non-login, non-interactive sessions!)
- IAAS: base image, user-data, automation + env vars
- PAAS: use extensions (e.g. Beanstalk ebextensions), or config definitions (appEngine > app.yml)

“Environment configuration”

Problem:

How to set environment configurations?

- Bare-metal: base image, automation + env vars (watch out for non-login, non-interactive sessions!)
- IAAS: base image, user-data, automation + env vars
- PAAS: use extensions (e.g. Beanstalk ebextensions), or config definitions (appEngine > app.yml)
- K8S:
  - Docker image env vars (ENV)
  - Use k8s Deployment parameters

```
spec:
  containers:
  - image: k8spatterns/random-generator:1.0
    name: random-generator
    env:
    - name: LOG_FILE
      value: /tmp/random.log
```

Kubernetes patterns book

“Complex configuration”

Problem:

How to set complex configurations, like app config files?

“Complex configuration”

Problem:

How to set complex configurations, like app config files?

- Bare-metal: automation + templates
- IAAS: automation + templates
- PAAS: probably not really needed; use extensions (e.g. Beanstalk ebextensions and templating magic)

“Complex configuration”

Problem:

How to set complex configurations, like app config files?

- Bare-metal: automation + templates
- IAAS: automation + templates
- PAAS: probably not really needed; use extensions (e.g. Beanstalk ebextensions and templating magic)
- K8S: ConfigMaps

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: random-generator-config
data:
  PATTERN: Configuration Resource
  application.properties: |
    # Random Generator config
    log.file=/tmp/generator.log
    server.port=7070
  EXTRA_OPTIONS: "high-secure,native"
  SEED: "432576345"
```

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: random-generator
spec:
  containers:
  - image: k8spatterns/random-generat
    name: random-generator
    volumeMounts:
    - name: config-volume
      mountPath: /config
  volumes:
  - name: config-volume
    configMap:
      name: random-generator-config
```

“immutable configuration”

Problem:

How to set immutable configurations?

“immutable configuration”

Problem:

How to set immutable configurations?

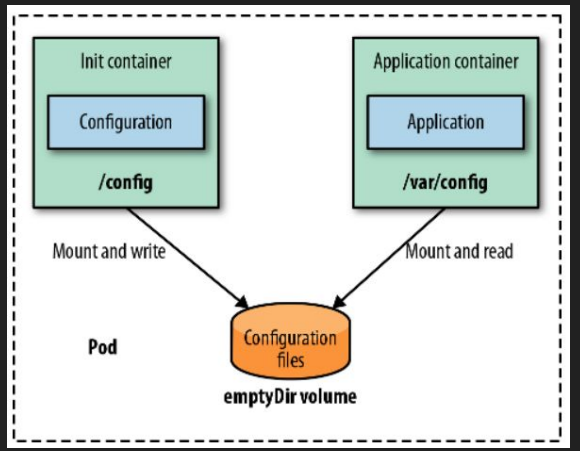
- Bare-metal: tricky; chattr +i; read-only config FS
- IAAS: as above or immutable VMs
- PAAS: usually provided by provider as deployment method (Beanstalk Rolling update type == Immutable)

“immutable configuration”

Problem:

How to set immutable configurations?

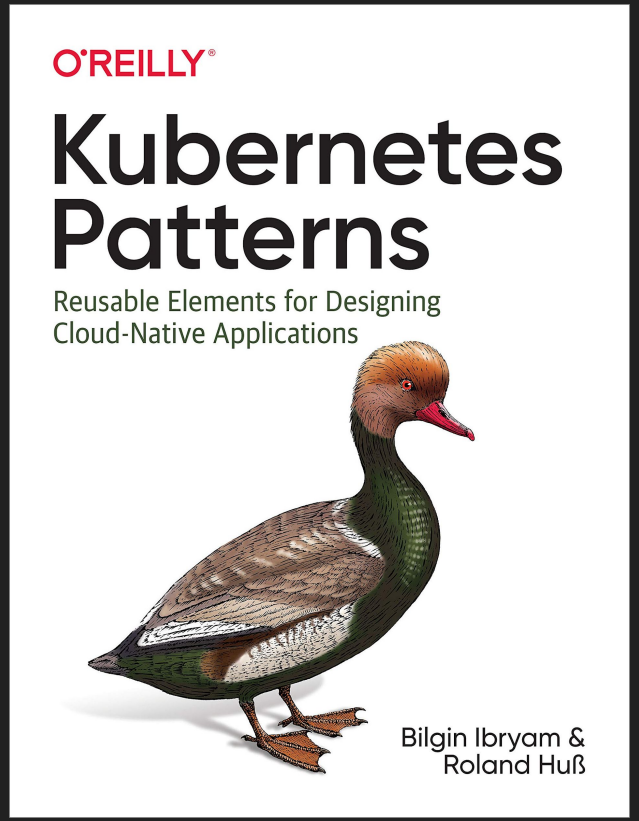
- Bare-metal: tricky; chattr +i; read-only config FS
- IAAS: as above or immutable VMs
- PAAS: usually provided by provider as deployment method (Beanstalk Rolling update type == Immutable)
- K8S:
  - Put configuration into Docker container (during build time)
  - Use init containers to prepare configs and mount as RO in service



Kubernetes patterns book

Cloud Native Design Patterns

Advanced patterns



- Controller
- Operator
- Elastic scale
- Image builder

[devooops]: Cloud - native design - patterns

Maciej Lasyk