

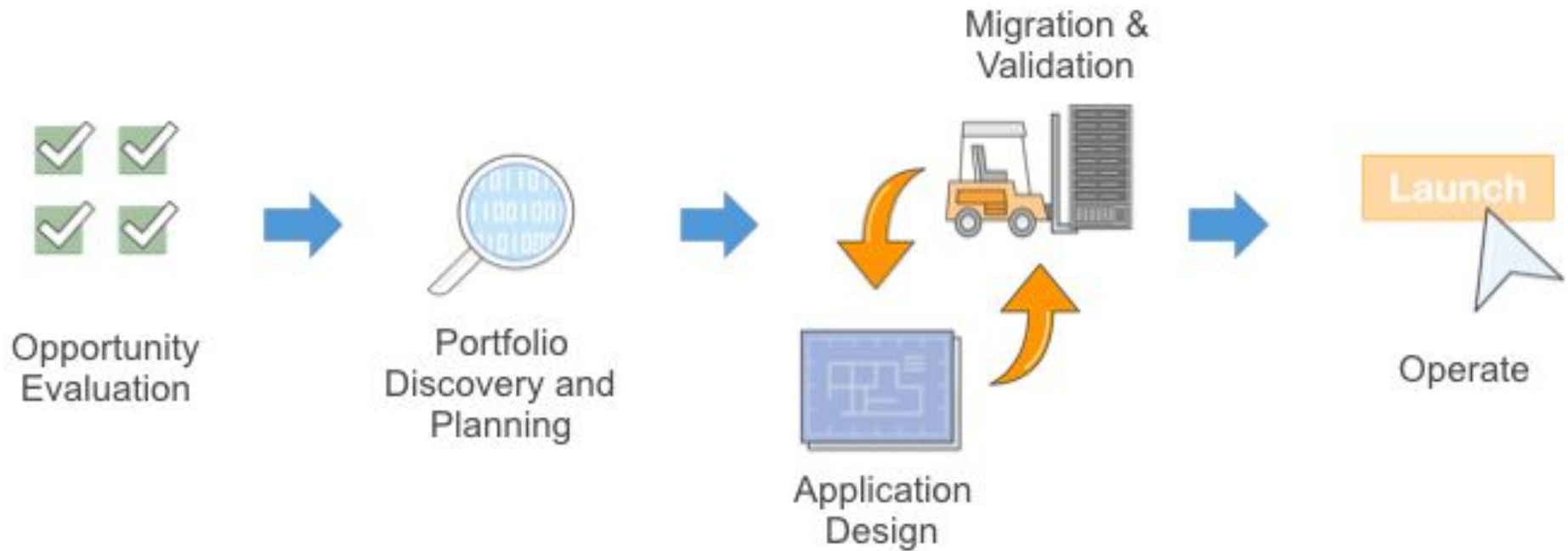


Chapter 12. Migrating to Cloud

Bilkent University | CS443 | 2021, Spring | Dr. Orçun Dayıbaşı

Introduction

- **Migration Process (5-phased)**



Opportunity Evaluation

- **Cloud Business Models**

- New applications born in the cloud but traditional products have to handle that transition
- In general, three types of online business models
 - B2C (Business-to-Customer): Ex. Amazon retail
 - B2B (Business-to-Business): Ex. AWS & Netflix
 - C2C (Customer-to-Customer): Ex. Amazon 2nd hand bookstore

- **Use of on-prem**

- If local computing is needed, e.g., when Internet connections are not reliable, then a “Hybrid Cloud” model is desirable with on-premise applications and data.
 - Consider: checkpointing, backup, disaster recovery, etc.

“**Checkpointing**” is a technique that provides fault tolerance for computing systems. It basically consists of saving a snapshot of the application's state, so that applications can restart from that point in case of failure. This is particularly important for the long running applications that are executed in the failure-prone computing systems.

Before Moving into the Cloud

- **Know your software licenses**

- Just because you have licensing that works for you in a traditional data center does not mean you can port those licenses into the cloud
- Different cost models
 - In general, you pay for resources by the CPU-hour in the cloud
 - Traditional software licenses are often based on the number of CPUs
- Example
 - You might have the following operating scenario:
 - From midnight to 9 a.m., run your application on **2 application servers** for redundancy's sake
 - From 9 a.m. to 5 p.m., launch **6 additional application servers** to support business-hour demand
 - For the evening hours through midnight, reduce the system down to **4 application servers**
 - Adding all that up, you pay for **110 hours** of computing time with **10 application server** licenses?

Before Moving into the Cloud

- **Know your TCO (Total Cost of Ownership)**

- Example

- Suppose your core infrastructure is:

- \$0.01/CPU-hour: one load balancer
- \$0.04/CPU-hour: two application servers
- \$0.08/CPU-hour: two database servers

- You would pay:

- Daily: $\$0.24 + \$1.92 + \$3.84 = \6
- Annual: \$2,190 (not including licensing fees, management tools, or labor)

- Use calculators of vendors (but it's not enough!)

- <https://aws.amazon.com/tco-calculator/>

- <https://azure.microsoft.com/en-us/pricing/calculator/>



Before Moving into the Cloud

- **Know your location**

- Ex: Amazon EC2 location concepts



Before Moving into the Cloud

● Know your location

- Ex: Amazon EC2 location concepts
 - Amazon EC2 is hosted in multiple locations world-wide. These locations are composed of **Regions, Availability Zones, and Local Zones**.
 - Resources are tied to one of them (or global)
 - Each Region (separate/independent geographic area) has multiple, isolated locations known as Availability Zones.
 - **Local Zones** provide you the ability to place resources (such as compute and storage) in multiple locations closer to your end users.
 - Resources aren't replicated across Regions unless you specifically choose to do so.
 - Failures can occur that affect the availability of instances that are in the same location. If you host all of your instances in a single location that is affected by a failure, none of your instances would be available

Resource	Type
AWS account	Global
Key pairs	Global or Regional
Amazon EC2 resource identifiers	Regional
User-supplied resource names	Regional
AMIs	Regional
Elastic IP addresses	Regional
Security groups	Regional
EBS snapshots	Regional
EBS volumes	Availability Zone
Instances	Availability Zone

[\(src\)](#)

Before Moving into the Cloud

● Know your location

- Ex: Amazon EC2 location concepts
 - Service (175+) → resources → locations
 - AWS deploys services regionally (regions are not equal)
 - Each **Region** is logical group of DCs and it has at least two AZs
 - An AZ consists of one or more data centers that logically treated as a single entity. All AZs in a Region are interconnected with high-bandwidth, low-latency networking
 - **Availability Zones** give us the ability to produce applications that are more highly available, fault-tolerant, and scalable
 - Unlike AZ, a **Local Zone** is not inside a Region boundary and places closer to our end-users. In order to provide a low latency communication (single-digit millisecond latency)
 - <https://aws.amazon.com/about-aws/global-infrastructure/>
- You should select the right Region for your business
 - Latency (Distance to our customers)
 - Price (e.g. Tax laws or financial situations)
 - Legal restrictions (e.g. European GDPR)
 - Service availability (e.g. New services)

Cost & Billing Practices

- **Pay as you go**

- CAPEX-intensive → OPEX-intensive
- At the end of a month, or agreed upon usage period, a Cloud Service Provider (CSP) will send the bill to its users
 - If up-front menu choices are not made wisely, then the user may be in for a surprise, but little can be done retroactively to reduce costs

- **Ex: Amazon EC2**

- There are no min. fees, 4 ways to pay (+Saving plans)
 - **On-demand Instances:** Customers pay for computing capacity by the hour.
 - **Spot Instances:** Allow customers to **bid on spare** EC2 computing capacity
 - **Reserved Instances:** Reserved Instances provide customers with a significant discount (up to 75%) compared to the On-Demand instance pricing.
 - **Dedicated Instances/hosts:** A dedicated instance is a virtual machine assigned to a customer and hosted on a specific host.

Cost & Billing Practices

- **Ex: Amazon EC2**

On-Demand

Pay for compute capacity by the hour. No long-term commitments



Spiky workloads

Reserved

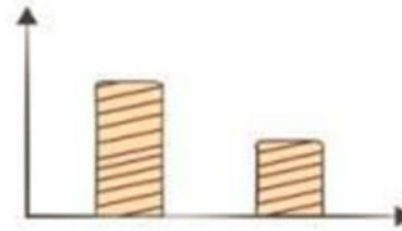
Pay upfront in exchange for hourly prices that are 50-75% lower than On-Demand



Committed utilization

Spot

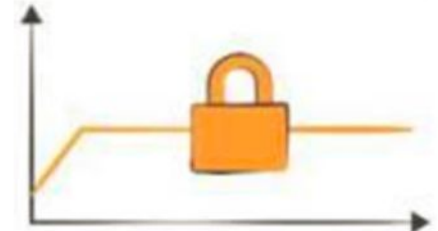
Bid for unused Amazon EC2 capacity



Time-insensitive workloads

Dedicated

Launch instances in VPC on dedicated customer hardware



Highly sensitive workloads

Getting Traffic

- **Locating your application in an elastic env.**

- Ex: Amazon Elastic IP (EIP)
 - When an instance is launched, AWS assigns it a public IP address. Every time the instance is stopped, this IP address is freed up.
 - An Elastic IP address is a static IP address designed for dynamic Cloud Computing.
 - An EIP is associated with customer's AWS account, not a particular instance.

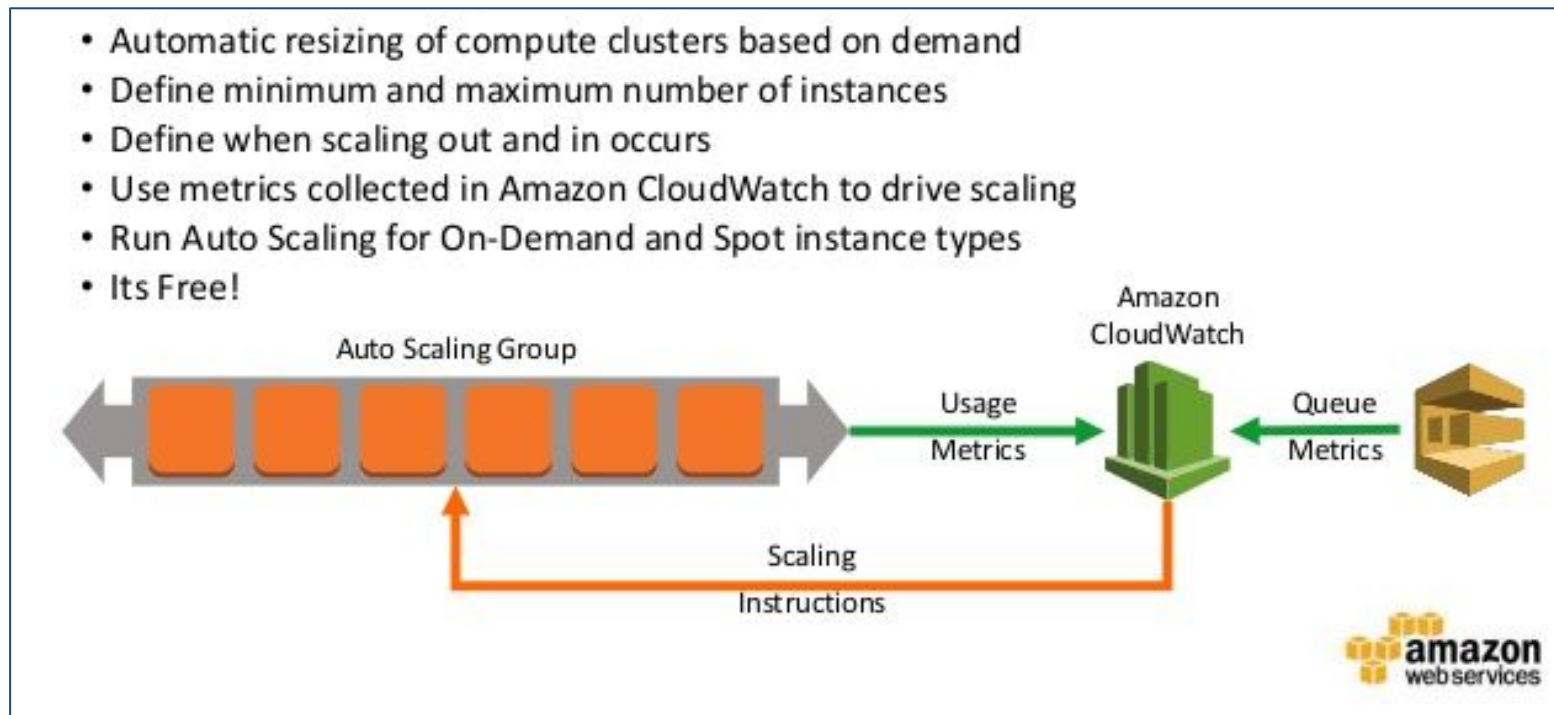
- **Routing the incoming traffic**

- Ex: Elastic Load Balancing (ELB)
 - ELB automatically distributes incoming application traffic across multiple Amazon EC2 instances (healthy ones)

Handling Load

● Auto-scaling

- Auto-scaling is the ability of a system to Scale In or Out depending on the workload
- Ex: Amazon auto-scaling
 - Auto scaling group (min, max, desired), launch configs, scaling plan



Portfolio Discovery & Planning

● Inspect

- What's in your environment, what are the interdependencies, what will you migrate first, and how will you migrate it?
- That list can outline a plan on migrating each of the applications/components in your portfolio and in what order
- There's less technical debt associated with something developed recently versus 20 years ago
 - modern app. → **rehosting** (aka “lift-and-shift”).
 - old mainframe app. → feature rationalization and **re-architecting**

Application Migration

- **6 Strategies (The 6 “R”s)**

- **1. Rehosting (lift-and-shift)**

- Most rehosting can be automated with tools (e.g. AWS VM Import/Export)
- Applications are easier to optimize/re-architect once they’re already running in the cloud
 - organization will have developed better skills
 - the hard part—migrating the application, data, and traffic—has already been done

- **2. Replatforming (lift-tinker-and-shift)**

- Here you might make a few cloud (or other) **optimizations** in order to achieve some tangible benefit, without changing the core architecture
- Ex: reduce the amount of time you spend managing database instances by migrating to a database-as-a-service platform or migrating your application to a fully managed platform

Application Migration

- **6 Strategies (The 6 “R”s)**

- **3. Repurchasing**

- Moving to another product (typically seen as a move to a SaaS platform)
 - Moving a CRM to Salesforce, an HR to Workday, etc.

- **4. Refactoring / Re-architecting**

- Re-imagining how the application is architected and developed, typically using cloud-native features
 - This pattern tends to be **the most expensive**, but, if you have a good product-market fit, it **can also be the most beneficial**.

- **5. Retire**

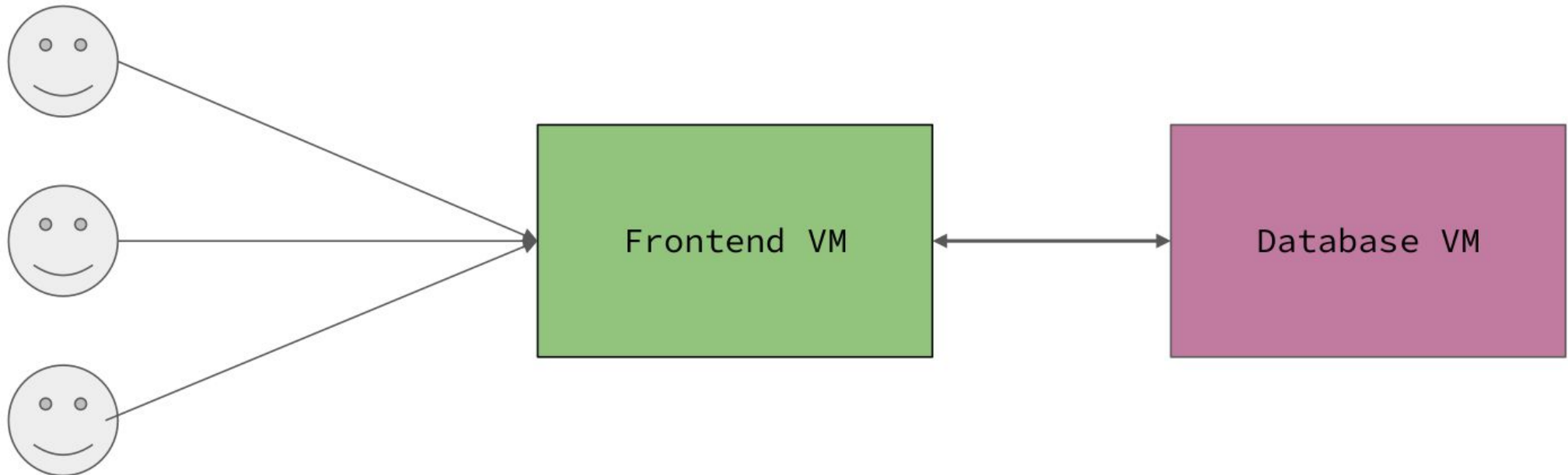
- Get rid of an IT portfolio is no longer useful (simply be turned off)

- **6. Retain (do nothing “for now”)**

- You should only migrate what makes sense for the business; and, as the gravity of your portfolio changes from on-premises to the cloud, you’ll probably have fewer reasons to retain

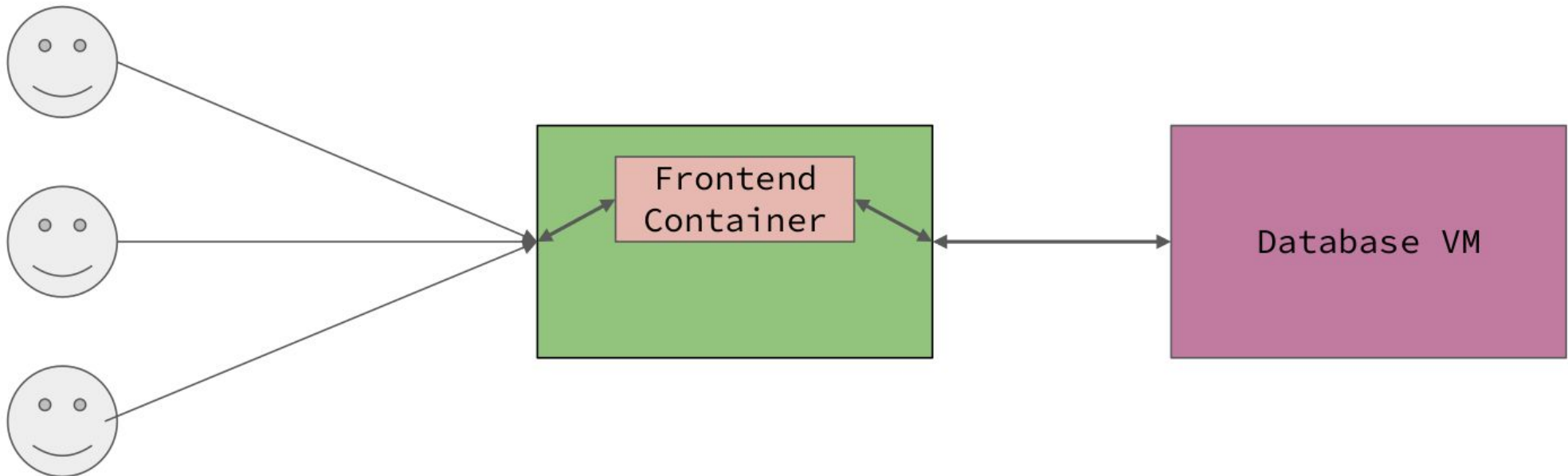
Application Migration

- **6 Strategies (The 6 “R”s)**
 - Ex: Refactor approach (v0)



Application Migration

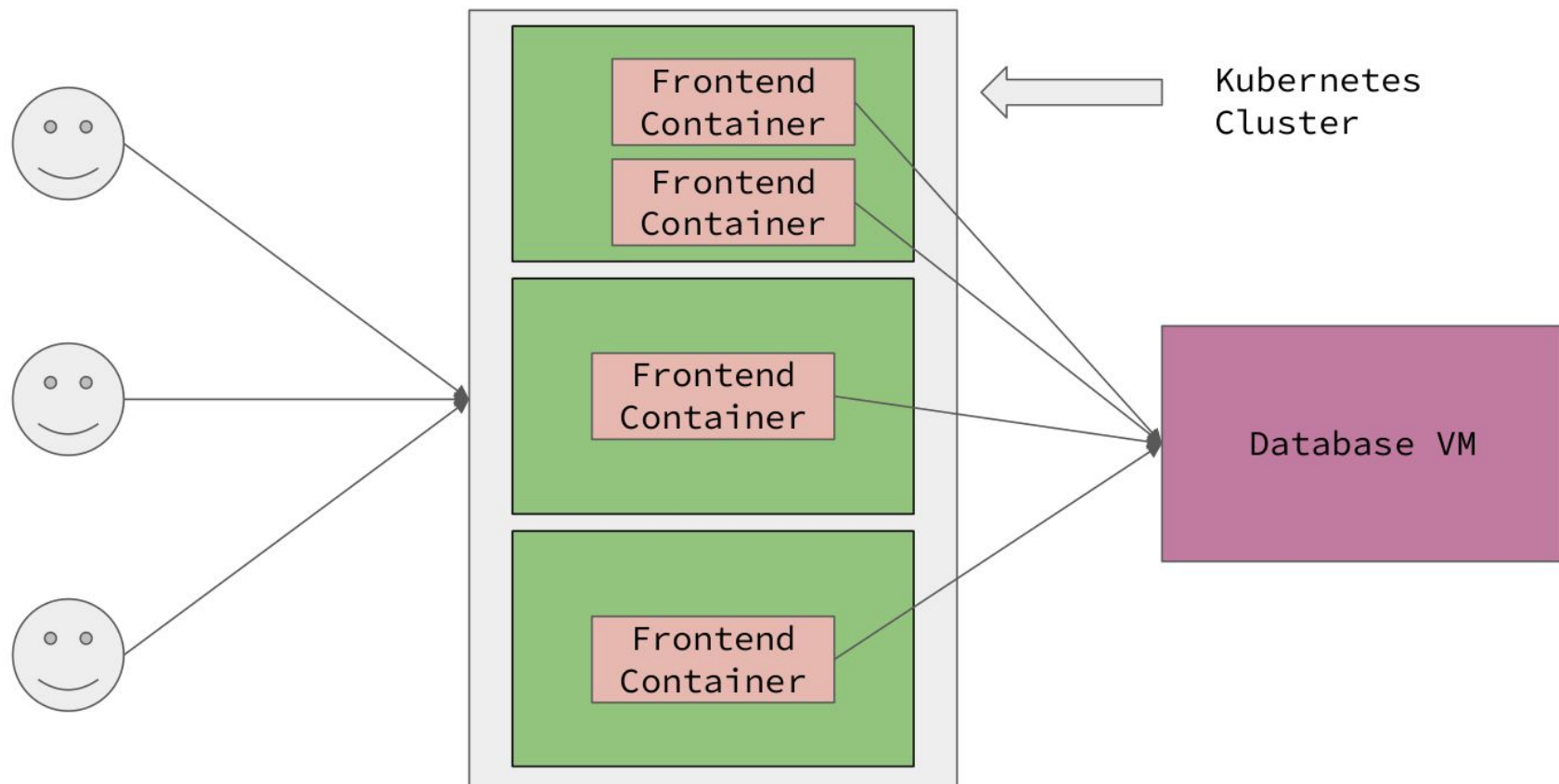
- **6 Strategies (The 6 “R”s)**
 - Ex: Refactor approach (v1)



Application Migration

- **6 Strategies (The 6 “R”s)**

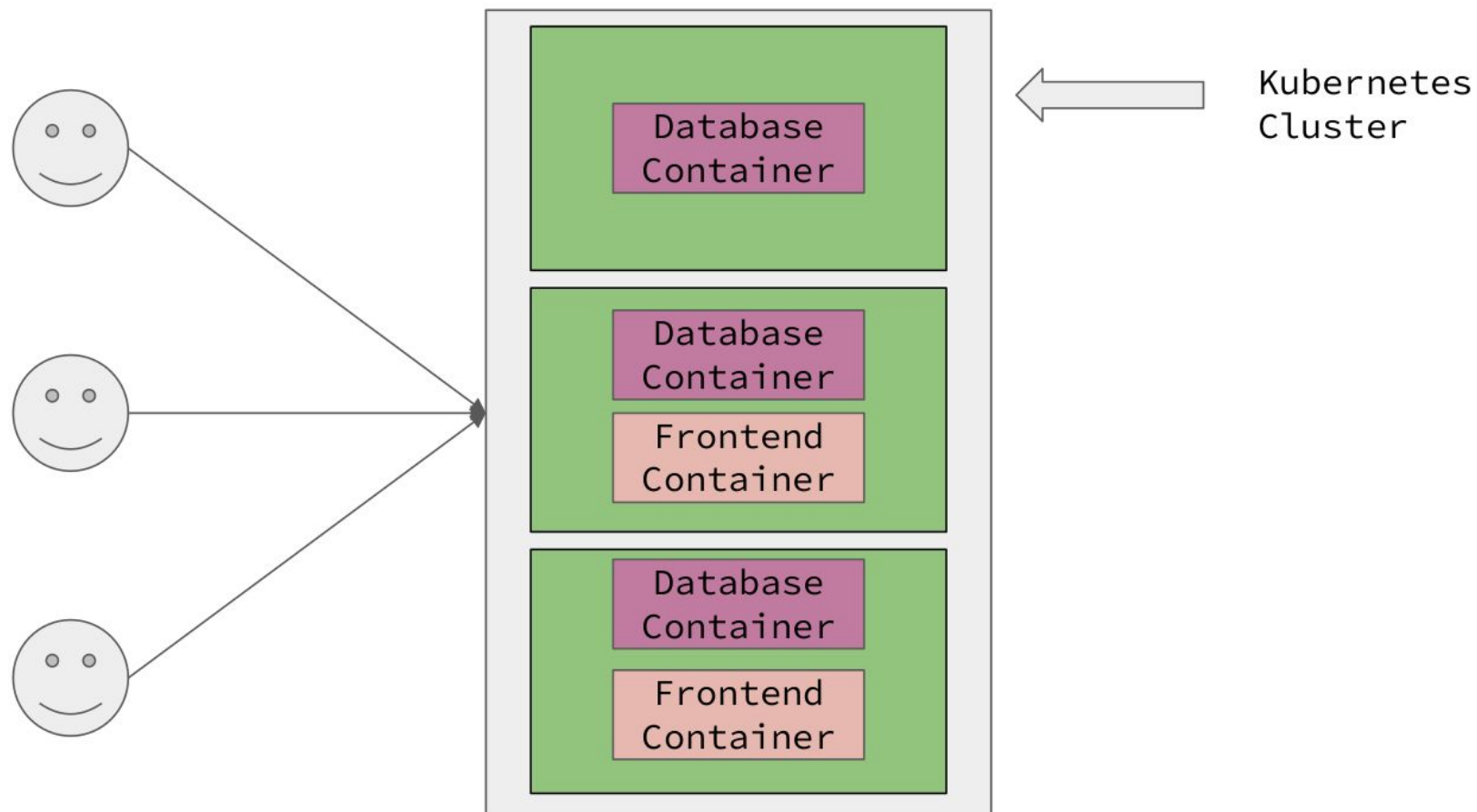
- Ex: Refactor approach (v2)



Application Migration

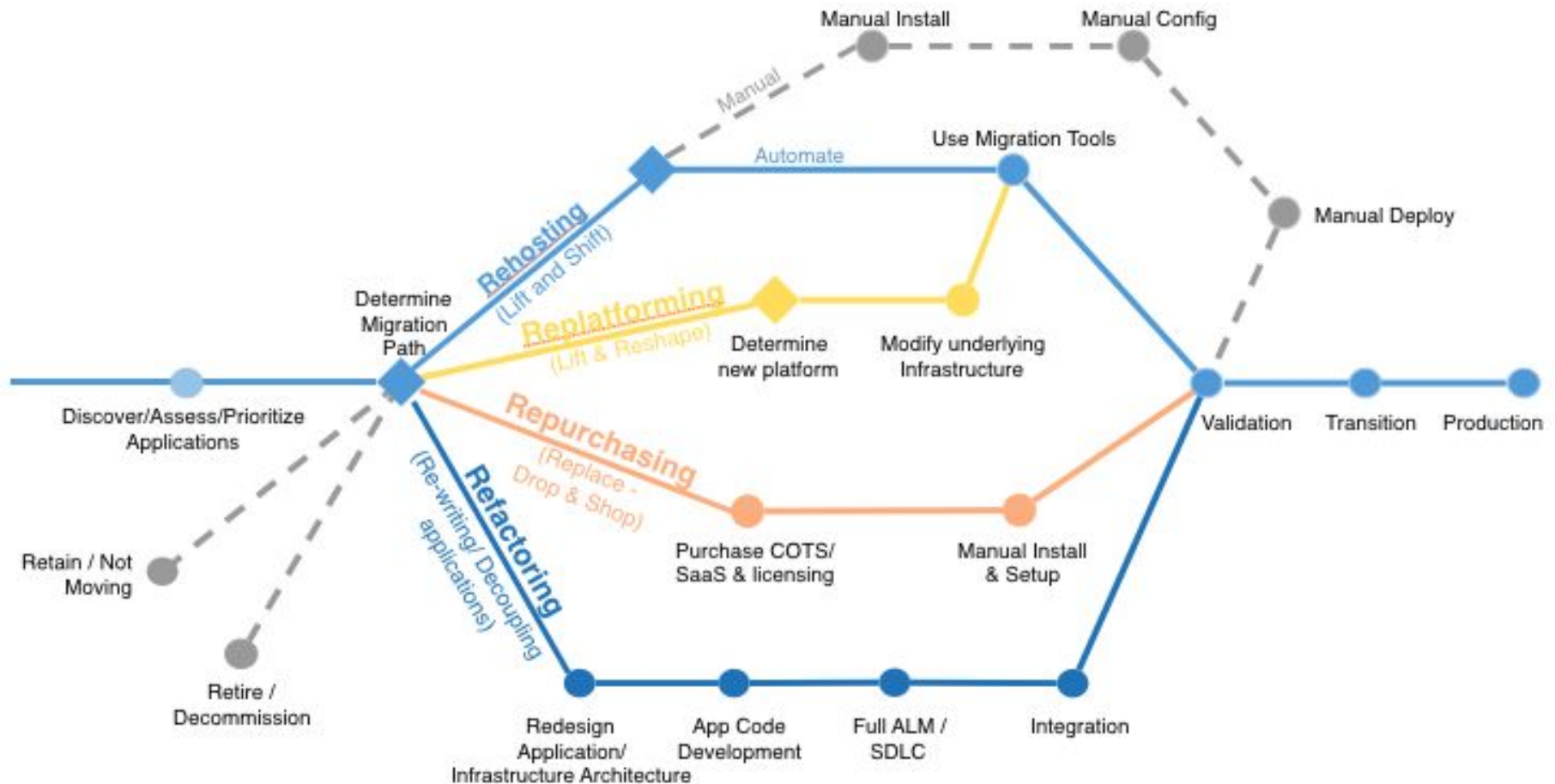
- **6 Strategies (The 6 “R”s)**

- Ex: Refactor approach (v3)



Application Migration

- 6 Strategies (The 6 “R”s)



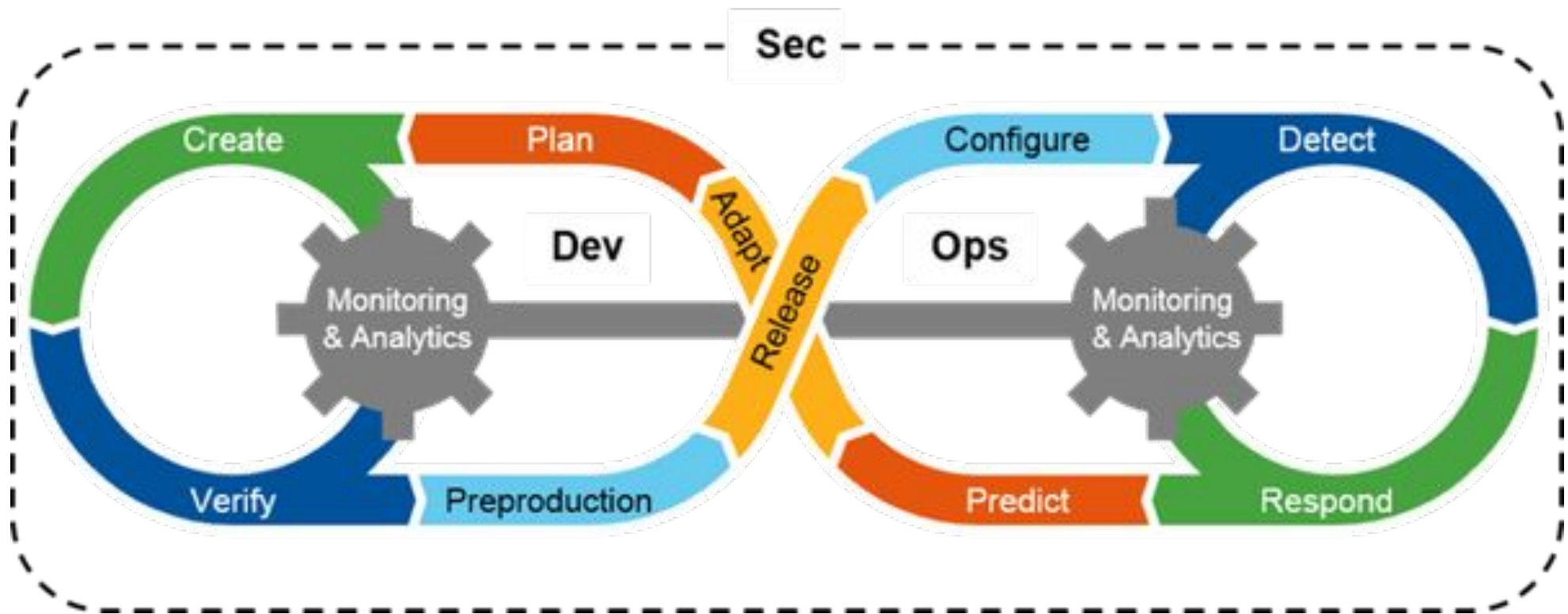
Operation

- **Modern Operating Model**

- Iterate on your new foundation, turn off old systems, and constantly iterate toward a modern operating model.
- Adopt a **DevOps** culture
- Operating model is an evergreen set of people, process, and technology that constantly improves as you migrate more applications
 - Your operating model will **continually improve** and become more sophisticated
- DevSecOps?
 - DevOps prioritizes delivery speed, DevSecOps shifts **security** to the left

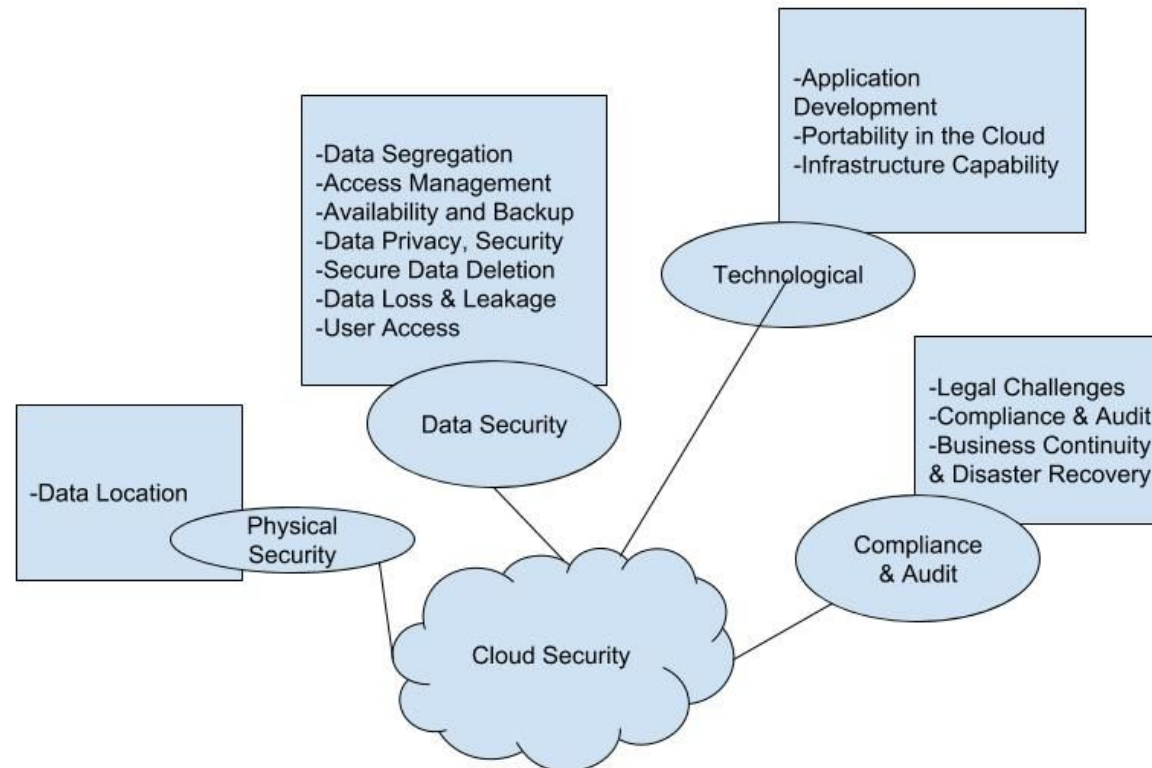
Security & Privacy

- **DevSecOps**
 - Integrate security into DevOps



Security & Privacy

- **Security**
 - Protecting the system as a whole
- **Privacy**
 - Protecting the users' data





Q/A