CS 4530 Software Engineering

Module 13: Principles and Patterns of Cloud Infrastructure

Jonathan Bell, Adeel Bhutta, Mitch Wand
Khoury College of Computer Sciences

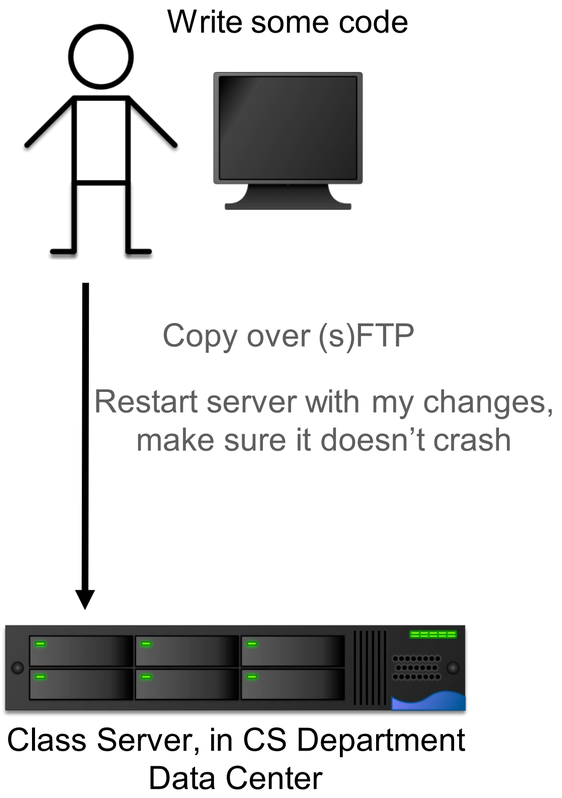
Learning Objectives for this Lesson

By the end of this lesson, you should be able to…

- Describe what “cloud” computing is
- Understand the role of virtual machines and containers in cloud computing
- Deploy a web app to the cloud

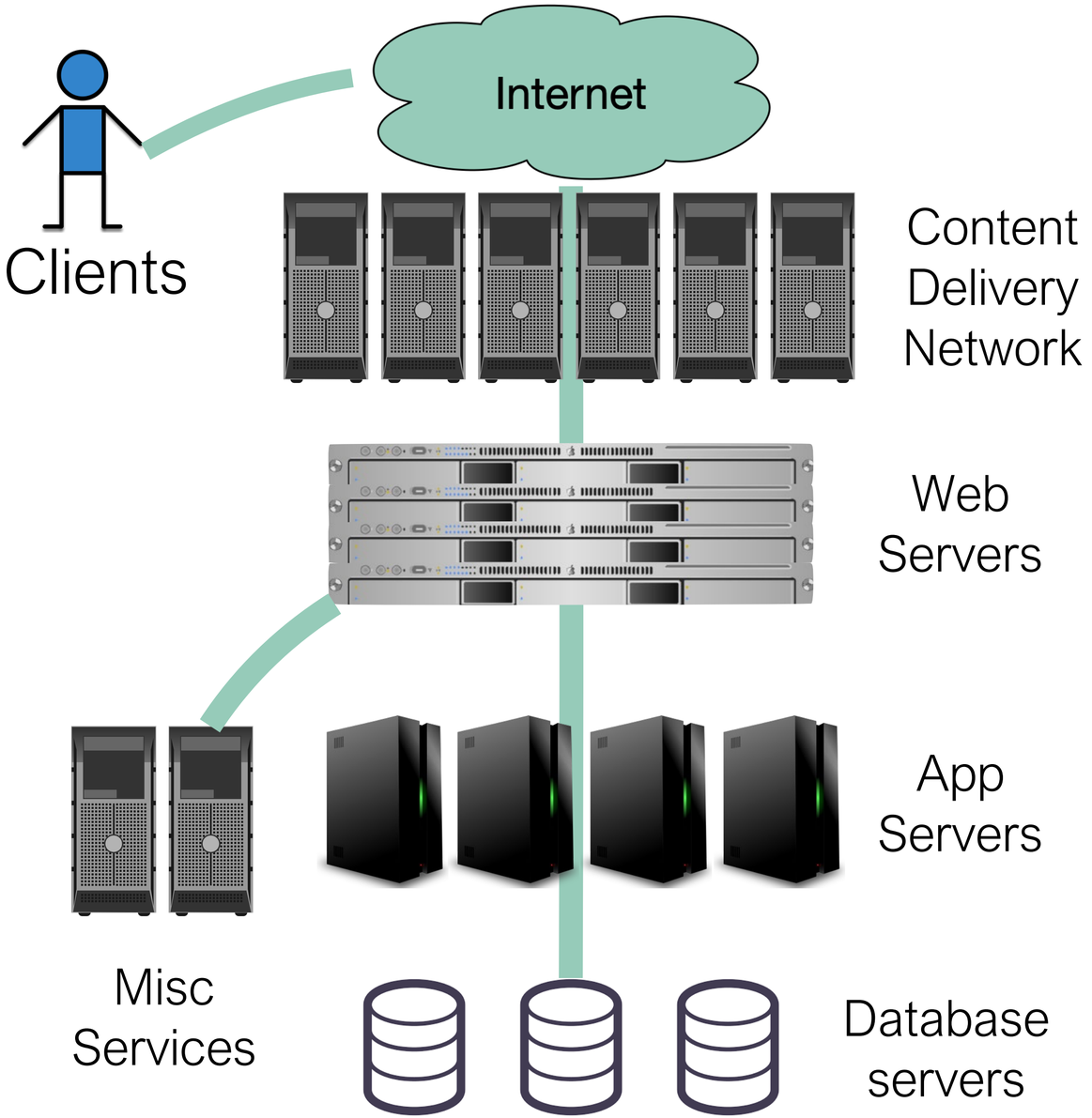
How to Deploy Web Applications?

- What we need:
  - A server that can run our application
  - A network that is configured to route requests from an address to that server
- Questions to think about:
  - What software do we need to run besides our application code?
  - Where does this server come from?
  - Who else gets to use this server?
  - Who maintains the server and software?



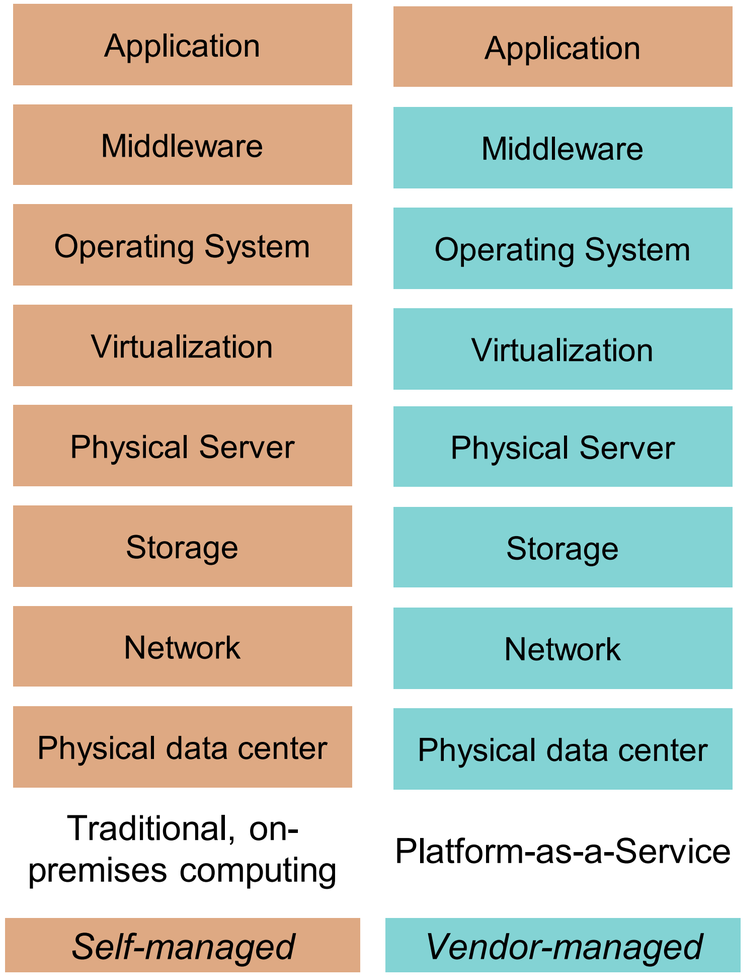
Many Applications Rely on Common Infrastructure

- Content delivery network: caches static content “at the edge” (e.g. cloudflare, Akamai)
- Web servers: Speak HTTP, serve static content, load balance between app servers (e.g. haproxy, traefik)
- App servers: Runs our application
- Misc services: Logging, monitoring, firewall
- Database servers: Persistent data



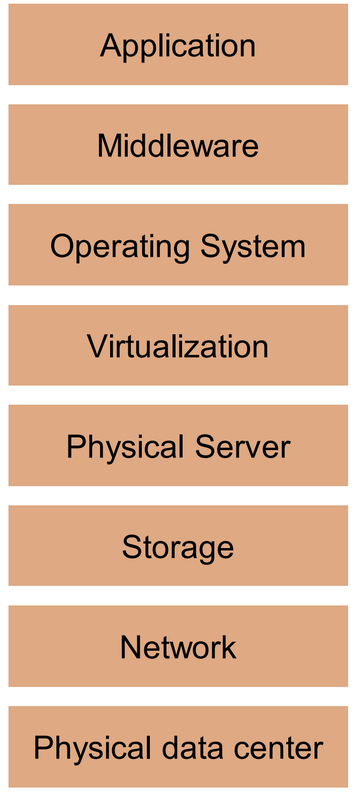
What is cloud infrastructure?

- Our apps run on a “tall stack” of dependencies
- Traditionally this full stack is self-managed
- Cloud providers offer a range of products that manage parts of that stack for us:
  - “Infrastructure as a service”
  - “Platform as a service”
  - “Software as a Service”



Cloud Infrastructure Creates Economies of Scale

- **At the physical level:**
  - Multiple customers' physical machines in the same data center
  - Save on physical costs (centralize power, cooling, security, maintenance)
- **At the physical server level:**
  - Multiple customers' virtual machines in the same physical machine
  - Save on resource costs (utilize marginal computing capacity)
- **At the application level:**
  - Multiple customer's applications hosted in same virtual machine
  - Save on resource overhead (eliminate redundant infrastructure like OS)



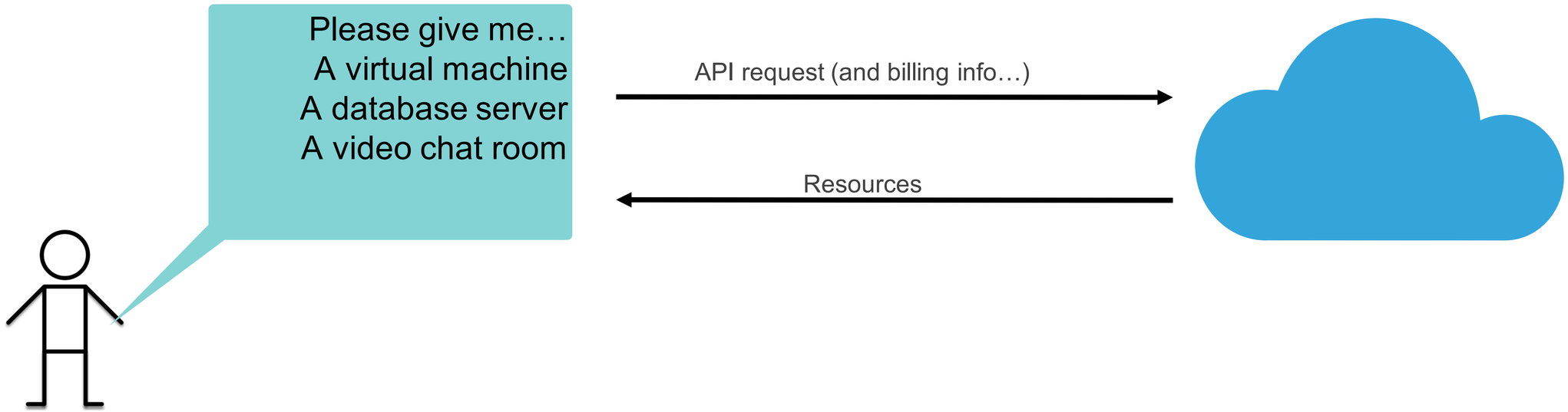
Multiple customers could share each of these tiers

Cloud Infrastructure Scales Elastically

- “Traditional” computing infrastructure requires capital investment
  - “Scaling up” means buying more hardware, or maintaining excess capacity for when scale is needed
  - “Scaling down” means selling hardware, or powering it off
- Cloud computing scales elastically:
  - “Scaling up” means allocating more shared resources
  - “Scaling down” means releasing resources into a pool
  - Billed on consumption (usually per-second, per-minute or per-hour)

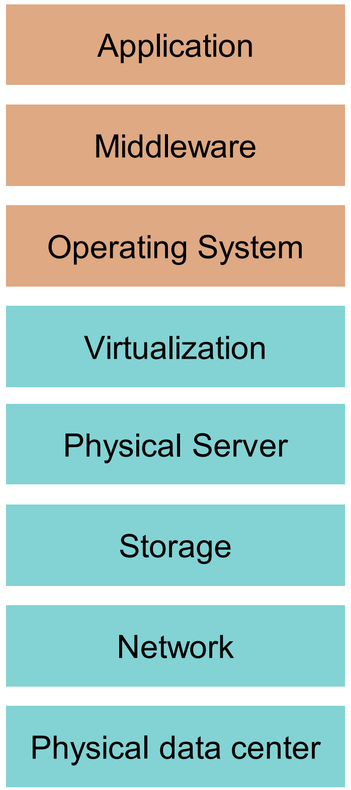
Cloud Infrastructure is On-Demand Access to Resources

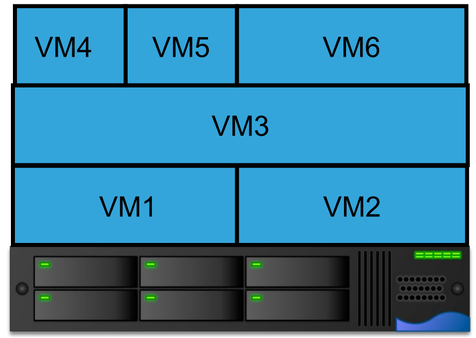
- Vendor provides a service catalog of "**X as a service**" abstractions
- Web interface or API allows us to provision resources on-demand



Infrastructure as a Service: Virtual Machines

- Rely on *virtualization* to slice a single large server into many smaller machines
- OS provides resource limits and quality guarantees per-VM
- Each VM runs its **own operating system**
- Examples: Amazon EC2, Google Compute Engine, Azure VMs





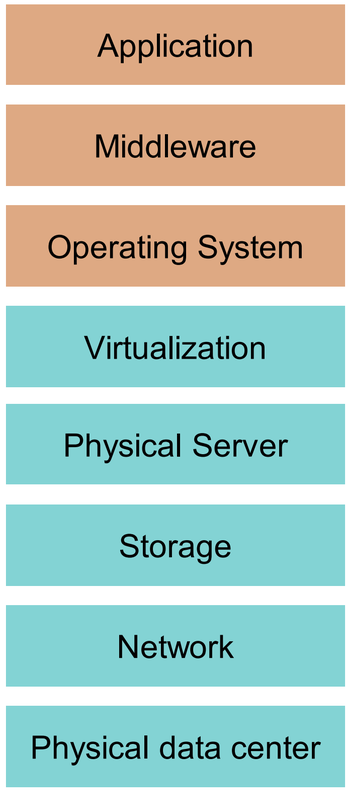
A single server in the cloud

IaaS

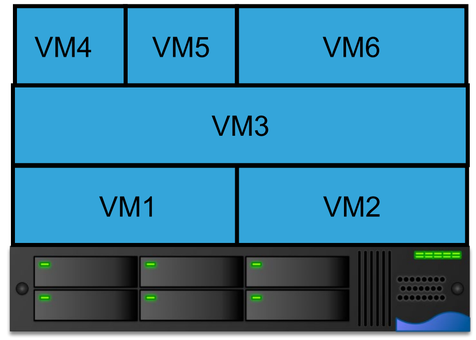
Self-managed | *Vendor-managed*

Virtual Machines are a Core Abstraction

- **Multi-Tenancy**: Multiple customers sharing same physical machine, oblivious to each other
- Decouples the entire application from physical machine: virtualization service can provide “live migration”
- Faster to provision and release than physical machines (max ~10 minutes vs ~hours/days)



IaaS

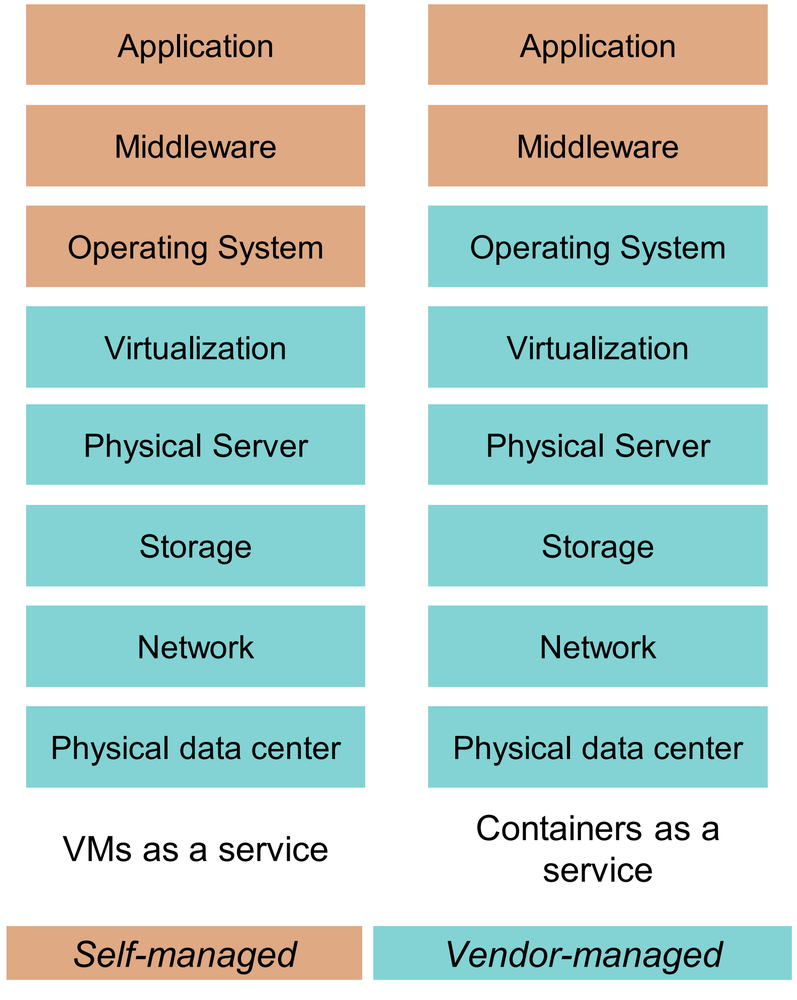


A single server in the cloud

Self-managed | *Vendor-managed*

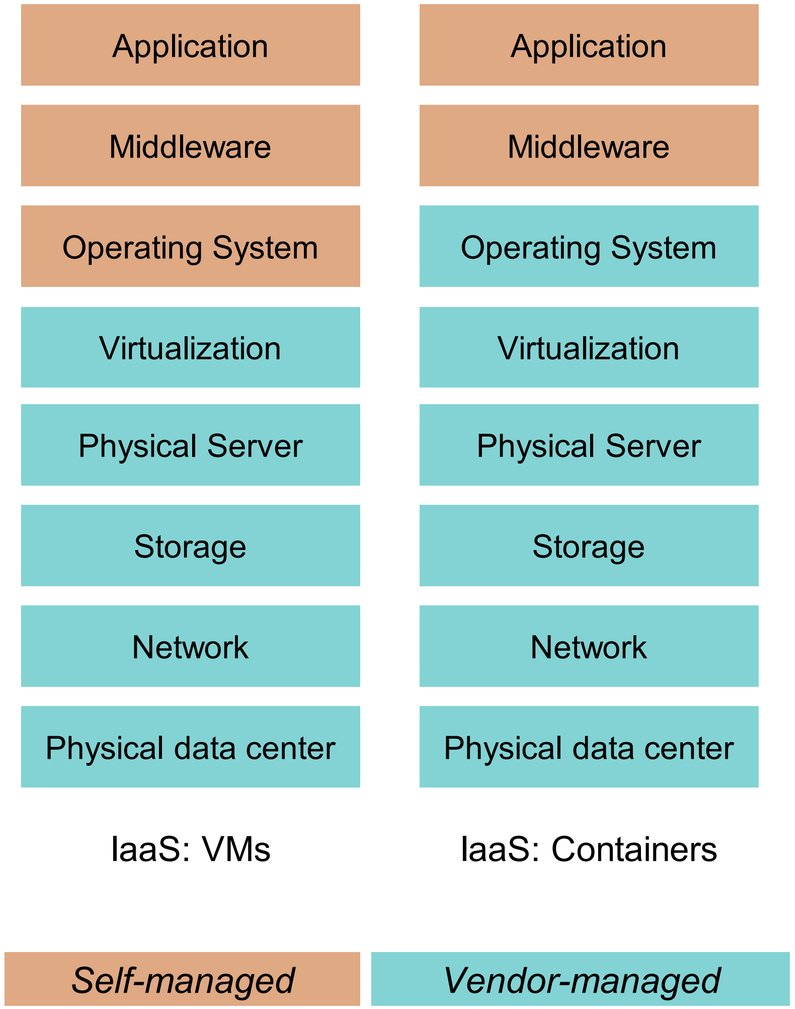
Virtual Machines to Containers

- Each VM contains a full operating system
- **What if** each application could run in the same (overall) operating system? Why have multiple copies?
- Advantages to smaller apps:
  - Faster to copy (and hence provision)
  - Consume less storage at rest



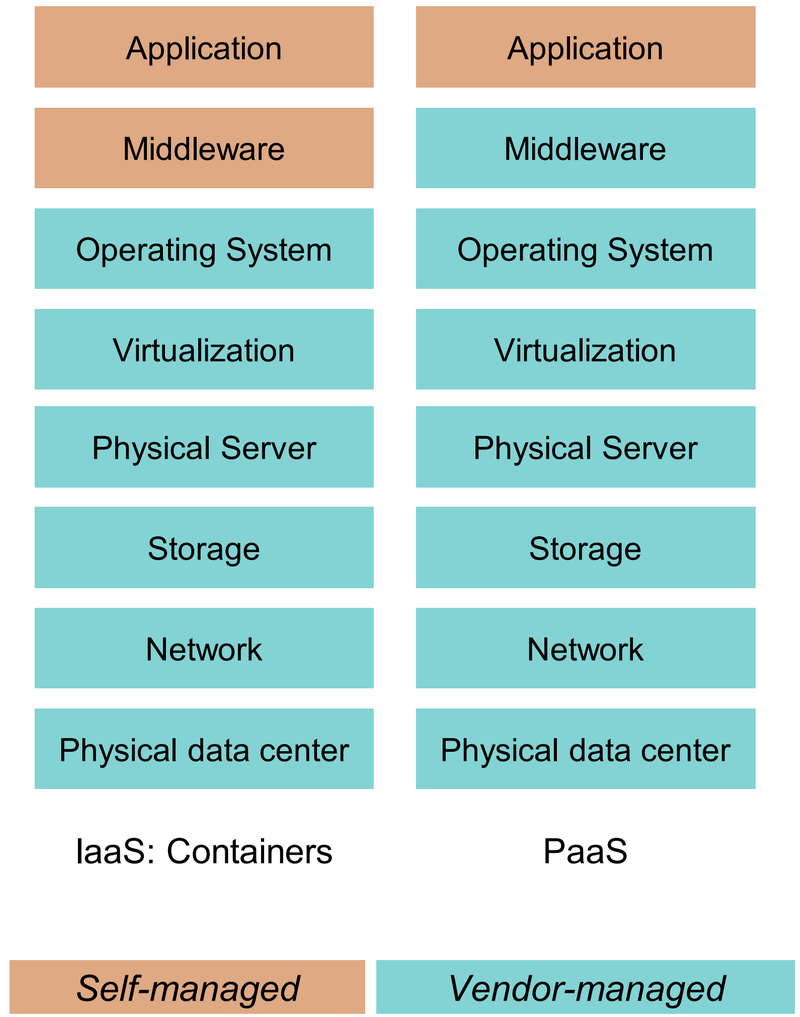
Infrastructure as a Service: Containers

- Each application is encapsulated in a “lightweight container,” includes:
  - System libraries (e.g. glibc)
  - External dependencies (e.g. nodejs)
- “Lightweight” in that container images are smaller than VM images - multi tenant containers run in the OS
- Cloud providers offer “containers as a service” (Amazon ECS Fargate, Azure Kubernetes, Google Kubernetes)



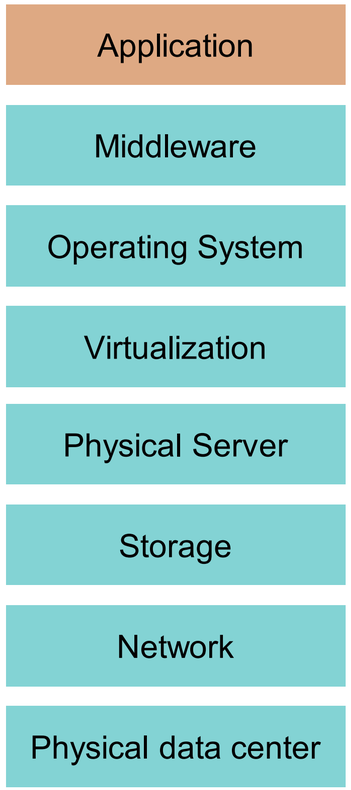
Many Apps Rely on Common Middleware

- Middleware is the stuff between our app and a user's requests:
  - Load balancer: route client requests to one of our app containers
  - Application server: run our handler functions in response to requests from load balancer
  - Monitoring/telemetry: log requests, response times and errors
- Cloud vendors provide managed middleware platforms too: "Platform as a Service"



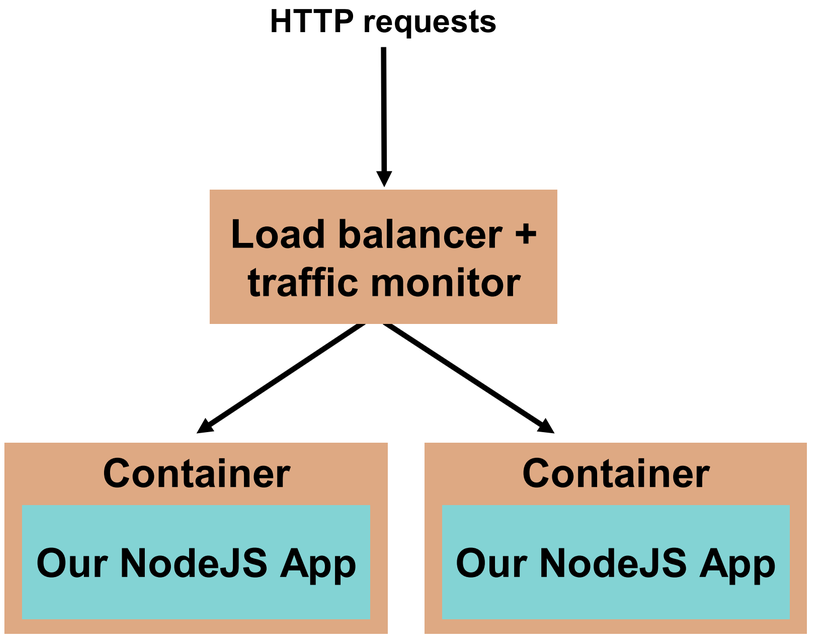
PaaS is the Simplest Choice for App Deployment

- Platform-as-a-Service (PaaS) products provide common components that most apps need, fully managed by the vendor: load balancer, monitoring, application server
  - Examples: Heroku, AWS Elastic Beanstalk, Google App Engine
- Some PaaS products are designed to deploy apps as *single functions* that are invoked when a web request is made, and don't run otherwise ("function-as-a-service")
  - Examples: AWS Lambda, Google Cloud Functions, Azure Functions
- Some PaaS products also provide databases and authentication
  - Examples: Google Firebase, Back4App



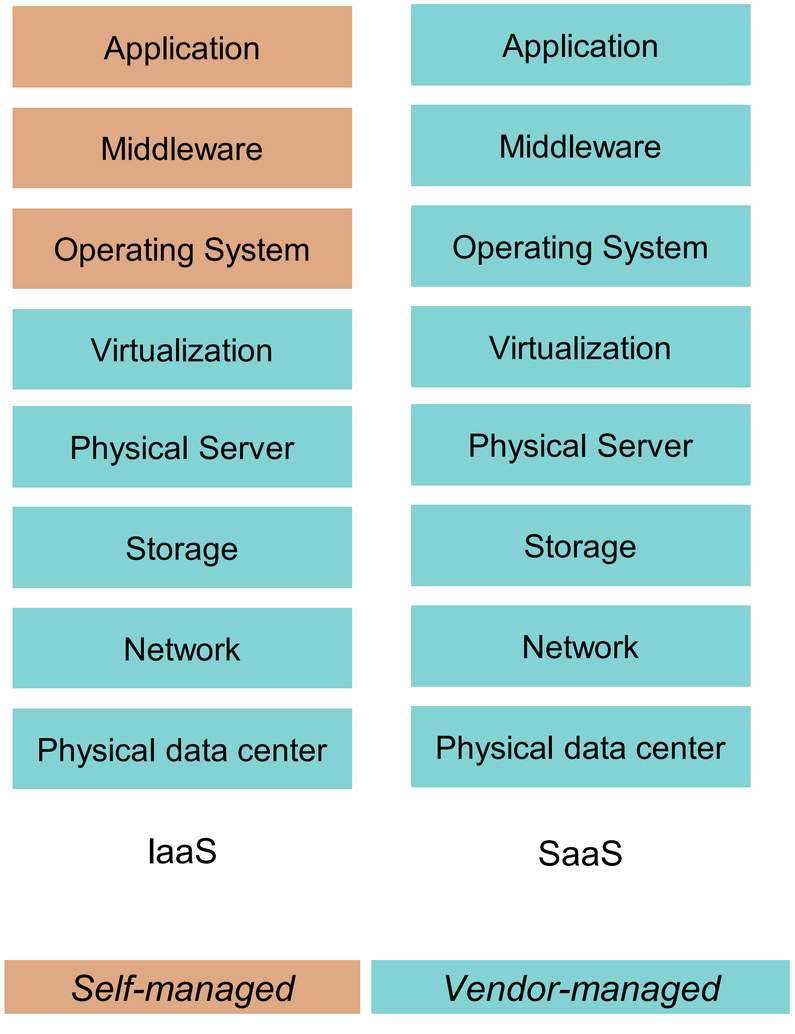
Heroku is a Platform as a Service

- Takes as input: a web app (e.g. NodeJS app)
  - No need to provide a container, entry point to our code is enough, e.g. “npm start”
- Provides: hosted web app at our choice of URL, with ability to scale resources up/down on-demand
  - Load balancer is fully managed by Heroku, makes scaling transparent
  - Can auto-scale down to use no resources, then only launch a container once a request has been received
  - Dashboard provides monitoring/reporting



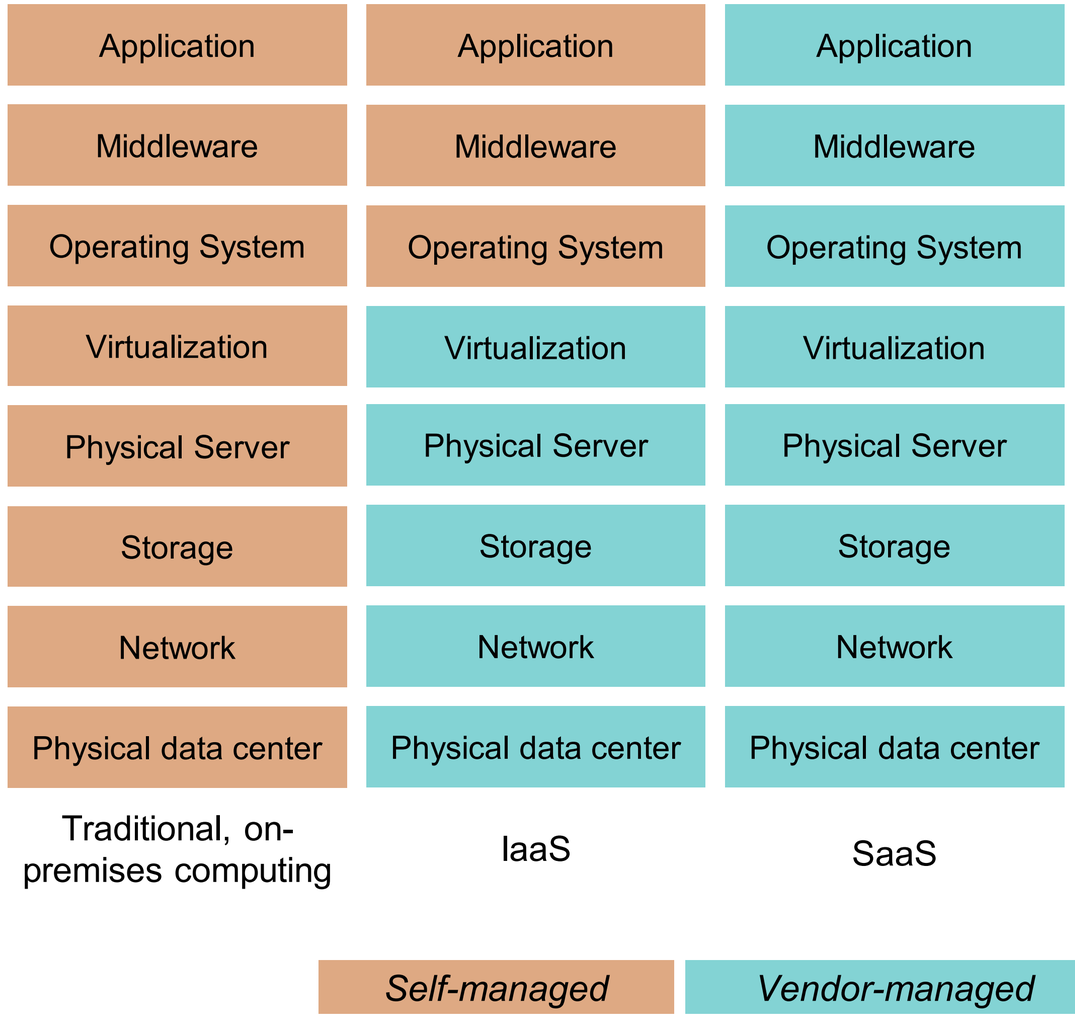
Software as a Service is Fully Managed

- Many applications require the *same* software components, cloud providers offer to operate those components for us
- Cloud providers also develop custom software for the market, offered only as a service
- Examples:
  - PostgreSQL database (open source product)
  - Twilio Programmable Video (proprietary video chat product)



Self-managed vs Vendor-managed Infrastructure

- Benefits to vendor-managed options:
  - Greater opportunity to reduce resource consumption, improve resource utilization
  - Less management burden
  - Less capital investment, greater operating expenses
- Benefits to self-managed options:
  - Greater flexibility and avoid vendor lock-in
  - More capital investment, less operating expenses



Cloud Infrastructure is Best Suited for Variable Workloads

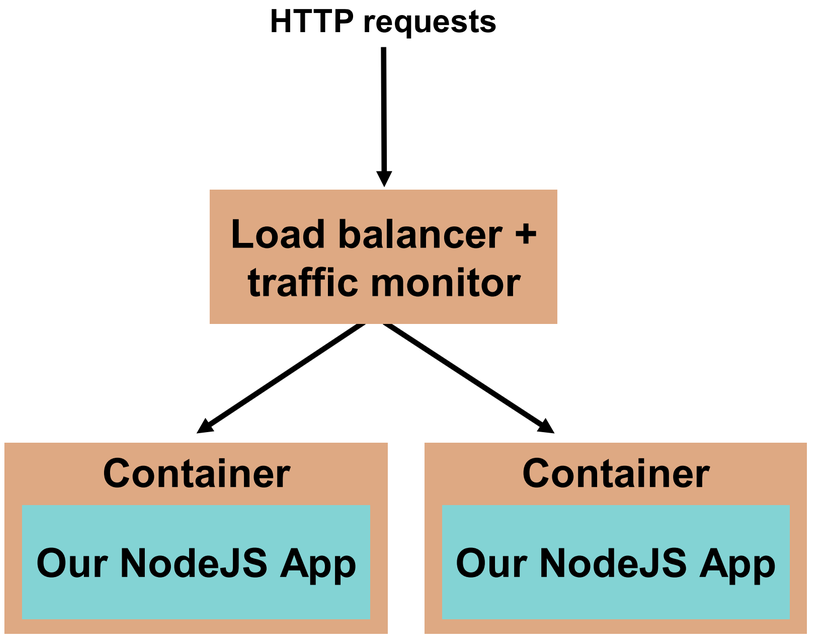
- Consider: Does your workload benefit from ability to scale up/down?
- Example: need to run 300 VMs, each with 4 vCPUs, 16GB RAM
- Private cloud: Dell PowerEdge Pricing (AMD EPYC 64 core CPUs)
  - 7 servers, each with 128 cores/256 threads, 512GB RAM, 3 TB storage = $162,104
- Public cloud: Amazon EC2 Pricing (M5.xlarge instances, $0.121/VM-hour)
  - 10 VMs for 1 year + 290 VMs for 1 month: $36,215.30
  - 300 VMs for 1 year: $317,988

Public clouds are not the only option

- “Public” clouds are connected to the internet and available for anyone to use
  - Examples: Amazon, Azure, Google Cloud, DigitalOcean
- “Private” clouds use cloud technologies with on-premises, self-managed hardware
  - Cost-effective when a large scale of baseline resources are needed
  - Example management software: OpenStack, VMWare, Proxmox, Kubernetes
- “Hybrid” clouds integrate private and public (or multiple public) clouds
  - Effective approach to “burst” capacity from private cloud to public cloud

Activity: Deploying Transcript Server to Heroku

- Heroku is one of the easiest platforms to use to host web apps
- Objective: deploy our (completed) transcript server to Heroku
- Download the handout from the module 13 page
- Handout has detailed README



Review: Learning Objectives for this Lesson

You should now be able to…

- Describe what “cloud” computing is
- Understand the role of virtual machines and containers in cloud computing
- Deploy a web app to the cloud