Selecting a Service Platform

Presenters

Animesh Gupta
&
Olivia Elston

IST 643
Enterprise Service &
Virtualized Systems

Introduction

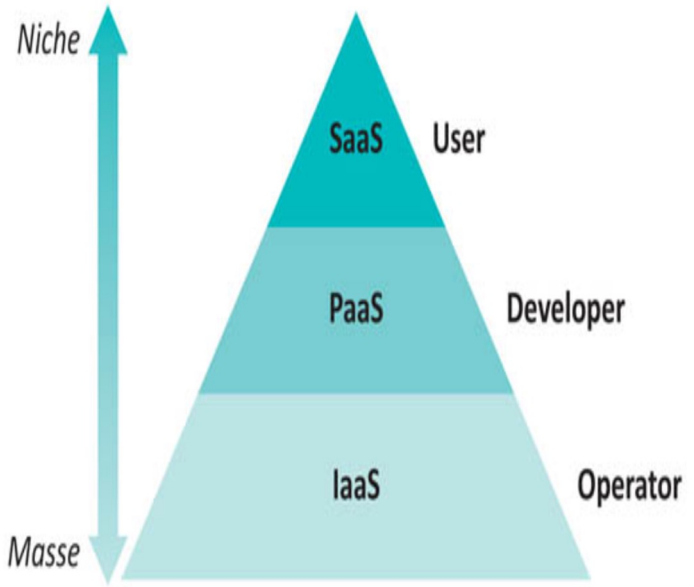
- This chapter provides an overview of the various types of platforms available in cloud computing, what each of them provides, and their strengths and weaknesses.
- We'll be talking about three main aspects of a platform,
  - Level of service abstraction: IaaS, PaaS, SaaS
  - Type of machine: Physical, virtual, or process container
  - Level of resource sharing: Shared or private
- Towards the end of this chapter, we also talk about the strategies for choosing between these different services.

When I hear someone touting the cloud as a magic-bullet for all computing problems, I silently replace "cloud" with "clown" and carry on with a zen-like smile.

Level of Service Abstraction

Abstraction is, essentially, how far users are kept from the details of the raw machine itself. That is, are you offered a raw machine (low abstraction) or are services provided as a high-level API that encapsulates what you need done rather than how to do it (high abstraction)?

The closer you are to the raw machine, the more control you have. The higher the level of abstraction, the less you have to concern yourself with technical details of building infrastructure and the more you can focus on the application.



The consumers of SaaS, PaaS, and IaaS.

Infrastructure as a Service

- IaaS provides bare machines, networked and ready for you to install the operating system and your own software. The service provider provides the infrastructure so that the customer can focus on the application itself.
- The machines provided by the vendor are usually virtual machines but may be physical machines. The provider takes care of the infrastructure: the machines themselves, power, cooling, and networking, providing internet access, and all datacenter operations.
- You still got a lot of work to do! YOU manage the OS.

- Providers charge for compute time, storage, and network traffic. These costs will affect how your application is architected. The software and operational choices have real costs and tradeoffs.
- Every Provider is different! Design decisions made for one provider may not be the right choice for other providers.

- Partitions or “reliability zones” for a reliable service.

IaaS Continued...

Why is Reliability an Issue?

For example, a service provider may offer four zones: U.S. East Coast, U.S. West Coast, Western Europe, and Eastern Europe. Each zone is built and managed to have limited dependencies on the others. At a minimum, customers should locate a service in one zone with plans for failover in another zone. A more sophisticated plan would be to have the service run in each zone with load balancing between all locations, automatically shifting traffic away from any zone that is down.

Having Geographic Diversity

Geographic diversity also permits customers to better manage the latency of their service.

For example, a service may be architected such that a user's data is stored in one zone with a backup kept in one other zone. Users from New York would have their data stored in the U.S. East Coast zone, with backup copies stored in the Western Europe zone. During an outage, the user is served from the backup zone; the service would not be as fast in such a case, but at least the data would be accessible.

IaaS Continued...

What's more?

Variety of Storage options:

- SQL/NoSQL DBs
- High Speed Storage, Cold Storage.

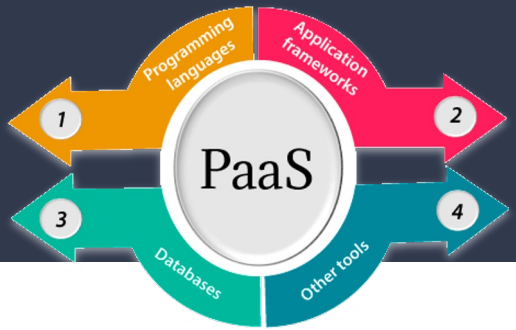
VPN accessible private networks.

Local and global load balancing services.

Elastic Scaling.

AWS vs. Azure vs. Google		
Provider	**Storage**	**Pricing**
Amazon S3	S3 Standard Storage	First 50 TB / Month $0.023 per GB Next 450 TB / Month $0.022 per GB Over 500 TB / Month $0.021 per GB
	S3 Standard-Infrequent Access (S3 Standard-IA) Storage	All storage / Month $0.0125 per GB
	S3 One Zone-Infrequent Access (S3 One Zone-IA) Storage	All storage / Month $0.01 per GB
Amazon EBS	Amazon EBS General Purpose SSD (gp2) Volumes	$0.10 per GB-month of provisioned storage
	Amazon EBS Provisioned IOPS SSD (io1) Volumes	$0.125 per GB-month of provisioned storage $0.065 per provisioned IOPS-month
	Amazon EBS Throughput Optimized HDD (st1) Volumes	$0.045 per GB-month of provisioned storage
Amazon Glacier	S3 Glacier Storage	All storage / Month $0.004 per GB
	S3 Glacier Deep Archive Storage	All storage / Month $0.00099 per GB
Google Cloud Storage	Multi-Regional	$0.026 - $0.036 per GB/month
	Regional	$0.02 - $0.035 per GB/month
	Nearline	$0.01 - $0.02 per GB/month
	Coldline	$0.004 - $0.014 per GB/month
Microsoft Azure	Block Blobs	$0.002/GB per month
	Azure Data Lake Storage	$0.001/GB per month
	Managed Disks	$1.54 per month
	Files	$0.060/GB per month

Platform as a Service



PaaS enables you to run your applications from a vendor-provided framework. These services offer you a high level of value, as they manage all aspects of the infrastructure, even much of the application stack. They offer very elastic scaling services, handling additional load without any input required from you. Generally you are not even aware of the specific resources dedicated to your applications.

Your application and hundreds of others might be sharing the same machine or your application may require the resources of hundreds of dedicated machines. You do not have to manage such decisions except to limit resource use to control costs.

Is there an Example?

Elastic Beanstalk is a platform within AWS that is used for deploying and scaling web applications. In simple terms this platform as a service (PaaS) takes your application code and deploys it while provisioning the supporting architecture and compute resources required for your code to run. Elastic Beanstalk also fully manages the patching and security updates for those provisioned resources.

There are many PaaS solutions in the cloud computing space including Redhat Open Shift, Google App Engine, Scalingo, Python Anywhere, Azure App Service, however AWS Elastic Beanstalk remains one of the leading PaaS choices among app developers.

PaaS Continued...

PaaS sounds way more fun! Let's ditch IaaS :D

- PaaS providers charge for their services based on how much CPU, bandwidth, and storage are used. This is similar to IaaS except the charges are higher to compensate for the more extensive framework that is provided.
- The downside of PaaS is that you are restricted to using what the vendor's platform provides. The platform is generally programmable but not necessarily extensible. You do not have direct access to the operating system.

For instance, you may not be able to add binaries or use popular libraries until the vendor makes them part of its service. Generally processes run in a secure "jail", which aims to prevent them from breaking out of the service's framework. For example, one PaaS offered the Python language but not the Python Imaging Library (PIL). It could not be installed by users because the framework does not permit Python libraries that include portions written in compiled languages.

PaaS provides many high-level services including storage services, database services, and many of the same services available in IaaS offerings. Some offer more esoteric services such as Google's Machine Learning service, which can be used to build a recommendation engine. Additional services are announced periodically.

Software as a Service

In simple lingo, It’s just a plain old Website.

A more technical definition would be, SaaS is a web-accessible application. The application is the service, and you interact with it as you would any web site. The provider handles all the details of hardware, operating system, and platform.

Some common examples would be Salesforce.com, Google Apps, Microsoft Office 365, Peoplesoft, Workday. The market is flooded with these softwares providing solutions for the most common business processes, be it HR, Payroll, Sales/Service/Marketing management, IT incident requests and much more!

The major selling point of SaaS is that customers do not have to concern themselves with software installation, upgrades, and operations. There is no client software to download. The service is fully managed, upgraded, and maintained by the provider. Because the service is accessed via the web, it can be used from any location.

SaaS Continued...

Customer Experience is the key!

Make it easy for customers to get started using your service. Rather than having to speak with a salesperson to sign up, signing up should be possible via the web site, possibly requiring submission of a credit card or other payment information.

Importing data and enabling features should also be self-service.

It is also important that people can leave the service in a self-service manner. This means users should be able to export or retrieve their data and close their accounts, even if this makes it easier to leave and move to a competitor.

Vendor Lock-in is a **RED** flag.

SaaS Continued…

Many SaaS offerings are upgraded frequently, often without warning, providing little opportunity for training. Users should be able to access major new releases for the purpose of planning, training, and user acceptance testing.

Rapid release: customers that want new features without delay.

Scheduled release: customers that would like new features to appear on a published schedule, perhaps two to three weeks after the rapid release track.

Data Privacy, Application hosting policy and Standards.

Your privacy policy will need to be a superset of all your customers' privacy policies. You may need to provide heightened security for certain customers, possibly segmenting them from other customers.

AWS's Compliance and Standards.

Machines and its types

Generally, there are three options for the type of machine that a service runs on: **physical machine**, **virtual machine**, and **process container**.

How do we decide?

The decision between physical, virtual, and container is a technical decision. Each has different performance, resource efficiency, and isolation capabilities.

The desired technical attributes should guide your decision on which to use.

IaaS generally provides the widest variety of options.

PaaS generally obscures what is used, as the user works in a framework that hides the distinction.

That said, most PaaS providers use containers.

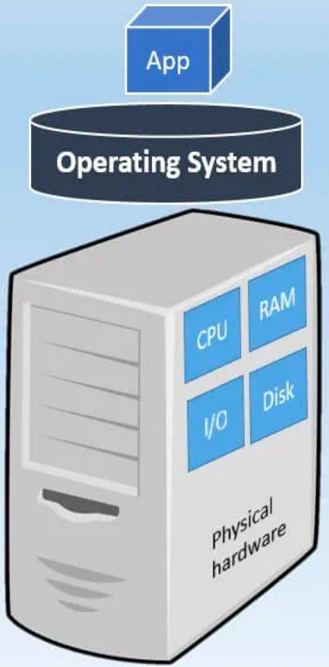
Physical Machines

A physical machine is a traditional computer with one or more CPUs, and sub- systems for memory, disk, and network.

These resources are controlled by the operating system, whose job it is to act as the traffic cop coordinating all the processes that want to share these resources.

The resources allocated to a running process (a program running on the system) are actual hardware resources.

As a result their performance is relatively predictable.

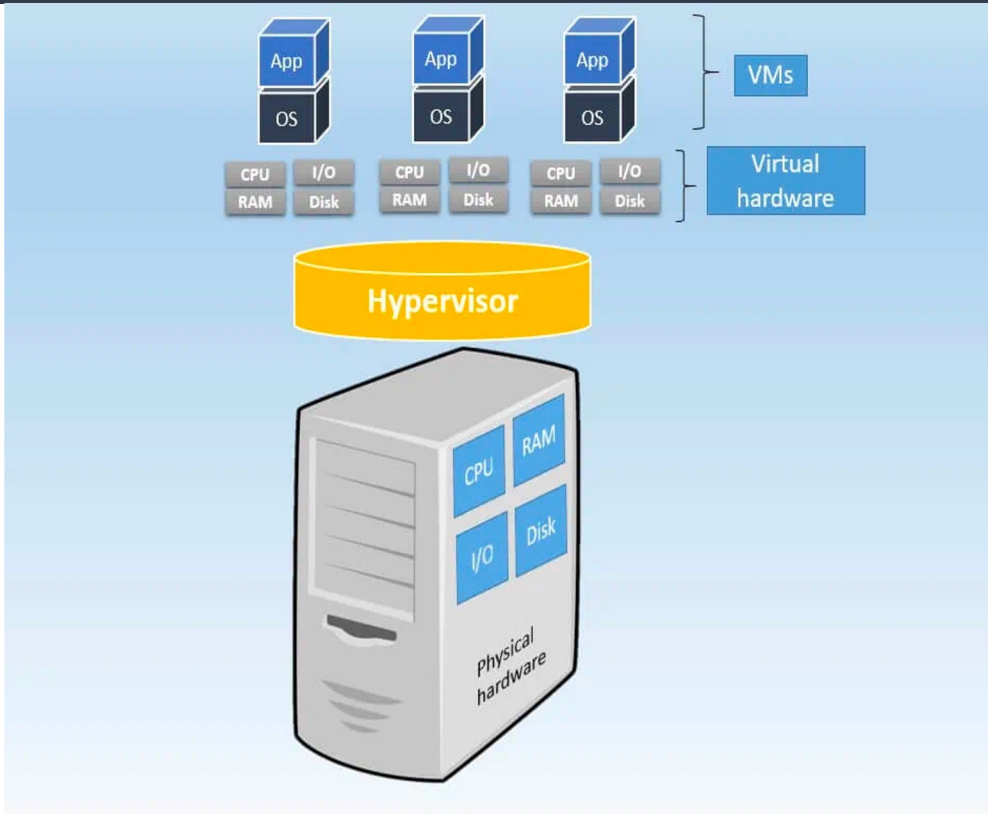


Virtual Machines

Virtual machines are created when a physical machine is partitioned to run a separate operating system for each partition.

Processes running on a virtual machine have little or no awareness that they are not on a physical machine. They cannot access the resources, such as disk or memory, of other virtual machines running on the same physical machine.

VMs make computing more efficient!



VMs Continued...

How does it really help?

The excess capacity of a physical machine aka, **stranded capacity** can be brought to use by permitting the creation of virtual machines that are the right size for their requirement.

Stranded capacity can also be mitigated by running multiple servers on the same machine. However, virtualization provides better isolation than simple multitasking.

Can you give me an example?

When two applications share a machine, when one application gets overloaded or has a problem that causes it to consume large amounts of CPU, disk space, or memory, it will affect the performance of the other application. Now suppose those two programs each ran on their own virtual machines, each with a certain amount of CPU, disk space, and memory allocated. This arrangement provides better isolation for each application from problems caused by the other.

VMs Continued...

Organizational Context

Different departments within an organization may not trust each other enough or have sufficient cross-department billing options to run software on the same machine. Nevertheless, they can share a pool of physical machines if each is able to create its own virtual machine.

Logistical Context

Running five services on one machine requires that any OS patches or upgrades be approved by all five services. If each service runs in its own virtual machine, then upgrades and patches can be done on different schedules for different services. In all these cases, virtual machines permit isolation at the OS level.

Benefits of VMs

VMs are fast to create and destroy. Some systems can spin up a new virtual machine in less than a minute.

Consequently, it is easy to create a virtual machine for a specific task and delete it when the task is completed. Such a virtual machine is called an **ephemeral machine** or short-lived machine.

Some systems create hundreds of ephemeral machines across many physical machines, run a parallel compute job, and then destroy the machines when the job is complete.

HVM vs PV

VMs are controlled through software, which means, you can use an API to create, start, stop, modify, and destroy virtual machines. Software can be written to *orchestrate* these functions on a large scale. This is not possible with physical machines, which have to be racked, cabled, and configured via manual labor.

Virtual machine functionality is provided by a combination of virtualization support in modern CPUs and virtualization control software called the **virtual machine monitor** (VMM)

Some virtualization systems permit a virtual machine to be moved between physical machines. The VMM puts the machine to sleep, copies its memory and all other state to a different physical machine, and continues its activities there. This permits a virtual machine to be moved to a different physical machine when the current one needs to be upgraded or repaired, or moved to a different failure domain in advance of a planned maintenance outage.

Disadvantages of VMs

Virtual machines are very heavy-weight. They run a full operating system, which requires a lot of disk space. They hold on to all the memory allocated to them, even if it isn't being used. The underlying OS cannot reallocate this memory to other machines.

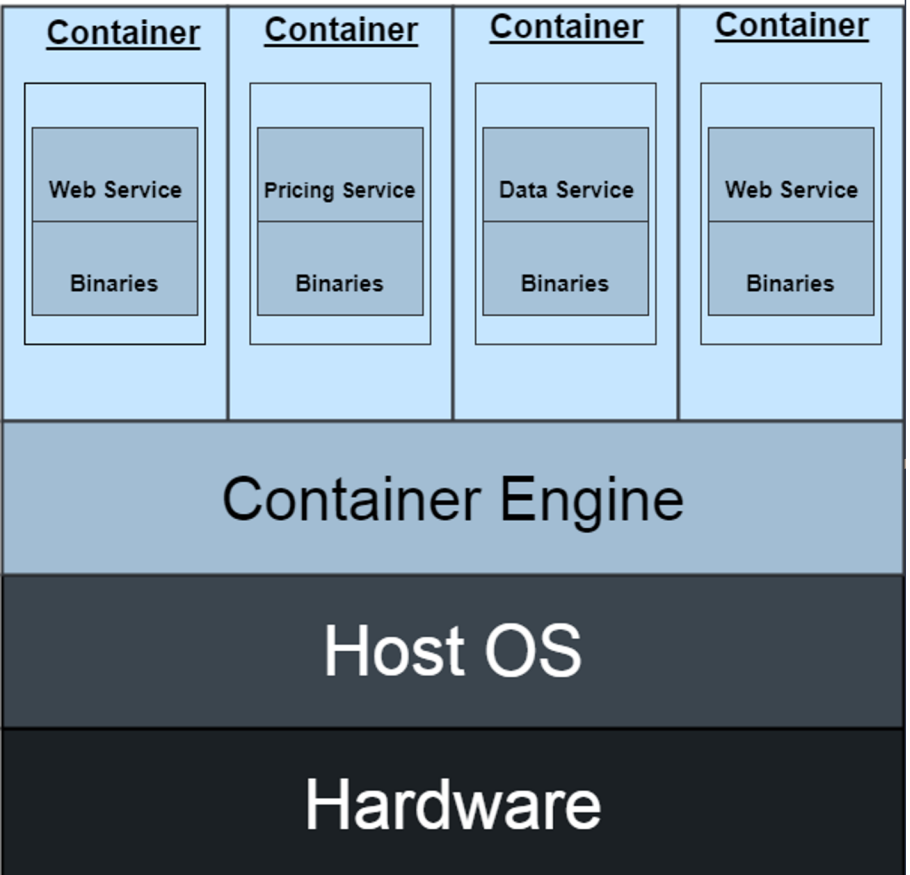
Because virtual machines run a complete operating system, the operational burden is similar to a full machine that needs to be monitored, patched, upgraded, and so on.

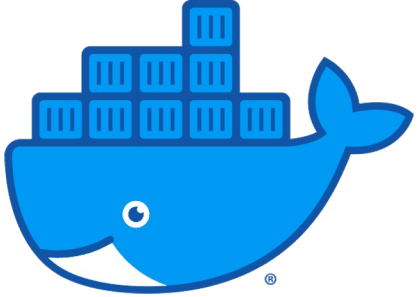
Also, because a complete operating system is running, each OS is running many background service processes such as maintenance tasks and service daemons. Those take up resources and add to the operational burden on the system administration team.

Steal time: it is the amount of CPU time that your virtual machine is missing because it was allocated to other virtual machines

Netflix tracks CPU Steal Time closely. In fact, if steal time exceeds their chosen threshold, they shut down the virtual machine and restart on a different physical server.

Containers





A **container** is a group of processes running on an operating system that are isolated from other such groups of processes. Each container has an environment with its own process name space, network configuration, and other resources.

The file system to which the processes have access consists of a subdirectory on the host machine. The processes in a particular container see that subdirectory as their root directory, and cannot access files outside that subdirectory without special accommodation from the host machine.

Unlike a virtual machine, which is allocated a large chunk of RAM and disk, containers consume resources at the same fine-grained level as processes. Thus they are **less wasteful**.

Containers Continued...

Let's talk about isolation

A container can kill or otherwise interact with only processes in its container. In contrast, processes that are not in containers can kill or interact with all processes, even ones in individual containers.

For example, when running the shell command ps within a container, it displays only processes running in that container. This is not a parlor trick; the container has no visibility to other processes. However, when the same command is run on the host from outside any container, it shows all processes, including those inside the each container. Thus, if you are logged into the main host (no particular container), you have global visibility and can serve as administrator for all containers.

How it tackles dependencies

Each container has its own copy of the packages, shared libraries, and other supporting files that it requires. For example, without containers one program might require a particular version of a library while another requires a very different version and cannot operate with the other version installed. This “dependency hell” is common. When each program is put in a different container, however, each can have its own copy of the library and thus the conflict is avoided.

Containers Continued...

Containers only run the files that are relevant to the software. No unnecessary background processes related to the OS and other daemons that usually run in case of a VM.

Software distribution with Docker!

Systems like Docker define a standardized container for software. Rather than distributing software as a package, one can distribute a container that includes the software and everything needed for it to run. This container can be created once and run on many systems.

The magic with PaaS!

Containers are usually the underlying technology in PaaS. They enable customers to be isolated from each other while still sharing physical machines. Because they consume the exact amount of resources they need at the time, containers are also much more efficient means of providing such shared services.

A Simple Analogy

Since many containers can coexist on the same machine, the resulting machine works much like a large hotel that is able to provide for many customers, treating them all the same way, even though they are all unique.

[Containers Vs VMs]

3.3 Level of Resource Sharing

Public Cloud

- A **third company owns** the infrastructure and serves multiple consumers.
- **Fine-grained sharing** mixes different client processing and data on the **same machine**
- Or sharing may be separated, like apartment renters with **well-defined partitions.**

Private Cloud

- runs on a company's **own premises**.
- This infrastructure may be built up for a specialized **internal project** or, more typically, as an **internal service provider**(offering available to projects and departments within the company)

Hybrids

- **combination** of public and private cloud, such as private clouds in rented datacenters.

PUBLIC CLOUD

. Many 3rd party providers
. Over the internet resource
. Convienient, cost-effective, easy-to-manage solution
. Scalable

HYBRID CLOUD

-Shares best of both public and private cloud
- Greater security - Shared from private cloud DNA
- Better control over data and processes

PRIVATE CLOUD

- Closed Resouce, available to select
- Highly-secure
- Traditional approach of Management and Maintenance
- VPN or Internet based accessibility to chosen users

3.3 Level of Resource Sharing cont.

A platform's private or public use is a business decision based on **compliance, privacy, cost, and control**

3.3.1 Compliance

- Rules depend on the **business, size, location, and public or private status**
- Regulatory compliance can be **audited**
    - **Failing an compliance audit** means the company can't do business until it passes another audit.
- Using a **public cloud** for data or services may **fail a compliance** audit.
    - I.e. Certain EU citizen data cannot leave the EU, according the EU Data Protection Directive.
- **Unless** the public cloud provider has **sufficient controls**,
- Even in **Failover scenario** a corporation that moves data to the cloud **will fail** an audit.
    - **the ability to switch automatically and seamlessly to a reliable backup system**

★ Auditing provides security assurance
★ Auditing lowers the risk of data breaches in the cloud.

3.3.2 Privacy

Public Cloud

Your data and code sit on **someone else's** equipment in their facility.

They may not have direct access to your data, but t**hey could sneak in.**

1. **Curious or malicious employees** could use diagnostic tools to view your data.

2. A **service provider may leak data** by not properly **erasing old storage systems.**

Service Providers:

- Because of these risks they **outline data protection in their contracts**.
- **require users' trust** to keep them as clients.

I.e. They maintain confidence by **being clear about their practices** and doing **external audits** to ensure compliance.

3.3.2 Privacy CONT.

Law Enforcement- Public vs. Private Cloud

PUBLIC

If **police have a warrant**, they can have a **third party** give them access **without alerting you**.

In a public cloud, **exposure might be to anybody**, including competitors, and become **front-page news**.

THREATS

Your data may be **accidentally leaked**.

By **software bugs, employee missteps**, or other issues, your data could be exposed to **other customers or the world.**

PRIVATE

In a **private cloud**, they **must notify you** before accessing your data (although clandestine techniques can be hidden even at your own site).

In a private cloud, **all customers** are from the **same organization**, which may be an acceptable risk; an **event may be contained**.

3.3.3 Cost

Public cloud costs **may be** less than creating your own infrastructure.

For Private

- Building infrastructure **demands engineering talent**
  - datacenter's cooling system, electric service, and design to its operation and services.
- **Pricey**
  - Amortizing costs over several customers **cuts costs**.

Vertical integration

1. helps DIY **save money**.
2. reduces costs by **eliminating "middlemen"** and **service provider profit margins**.
3. **gets cheaper** at a certain point, TCO and ROI can assist decide the optimal alternative.

3.3.3 Cost CONT.

Private vs. public cloud costs are like **renting vs. buying a home.**

Long-term, **owning** is **cheaper** than renting. **Short-term**, **renting** may be **cheaper**.

Renting a hotel room for a night is smarter than **buying and selling a city building**.

A private cloud makes sense if you'll **utilize it all** and need it **long** enough to **pay for itself**.

If the need is **minimal or temporary**, use a **third-party service**, **public cloud**

3.3.4 Control

Private clouds provide **more control**.

You can **define hardware, network topology**, etc.

Create, buy, or license whatever **feature** you need.

*Changes move as fast as your company does

Public clouds **lack control**.

limited options

Most **providers** respond to **feature requests**, but you're just one client.

Providers must **focus** on **popular features** and may **not** offer **specialized ones**.

Vendor choose hardware implies **losing** the opportunity to describe **low-level needs**

(specific CPU types or storage products).

3.4 Colocation

Colocation is **not a "cloud environment,"** but it's a handy technique to **supply services**.

Occurs when a **datacenter owner rents space to “tenants”**.

Renting datacenter space is **inexpensive** for **all sizes** of **organizations.**

Building your own datacenter

- **major investment**
- involves **specialist expertise**
  - cooling, design, networking, location selection, real-estate administration, etc.

Brief History

"colocation" derives from **telecommunications.**

Telecommunication companies established datacenters to **house their equipment and systems**.

third-party companies supplied services to telecom consumers, and it was **easier to colocated their equipment** with the **telecom's datacenters.**

3.4 Colocation CONT.

- **Colocation service** now refers to **any datacenter renting.**
- This service is good for **small or medium** datacenter needs.
- **Colocation datacenters**
  - **Well-designed and well-run**
  - **faster than building** a datacenter, which can take years.
  - **advantageous** for **small spaces**.
- Example: A corporation may desire a **single rack of equipment** in 12 countries to **improve client access**.

ISP-internet service provider & colocation

- ISPs **extend** their network into colocation spaces so tenants can **connect directly**.
  - **enhances user access.**

Colocation companies may also **offer internet access** to renters by **combining multiple ISP connections**.

Tenants use internet access **instead** of **managing their own** ISP connections.

3.5 Selection Strategies

There are many strategies one may use to choose between IaaS, PaaS, and SaaS

Strategies:

1. Default to Virtual
2. Make a Cost-Based Decision:
3. Leverage Provider Expertise
4. Get Started Quickly
5. Implement Ephemeral Computing
6. Use the Cloud for Overflow Capacity
7. Leverage Superior Infrastructure
8. Develop an In-House Service Provider
9. Contract for an On-Premises, Externally Run Service
10. Maximize Hardware Output
11. Implement a Bare Metal Cloud

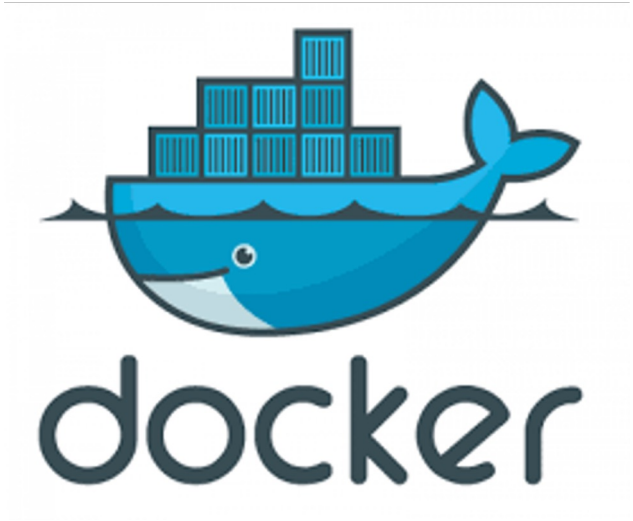
Default to Virtual:

When possible, **use containers or virtual machines** instead of **real ones**.

Opt for **Physical machines ONLY** when performance **goals cannot be met**; they are better at **disk and network I/O**.

Fully virtualized environments are **typical**, with the exception of I/O-intensive databases.

A **web load balancer** may also require **physical hardware** due to **high-bandwidth network interface cards** or predictable performance from **dedicated network connections**.



VM > PM
Easy maintenance, application provisioning, availability and convenient recovery

Make a Cost-Based Decision:

Cost-wise, choose **public or private cloud**

Create a **business plan** that compares the **cost** of doing a **project in-house versus on the cloud**

Pick the **cheaper option**

DIY may be cheaper for **multiyear projects**

Public cloud is cheaper for a **short-term project**.

Public Cloud

- Services are owned and operated by a third party provider.
- The maintenance is bared by the service provider.
- Pay-as-you-go model.Thus, the setting and operating cost is less.
- Lesser security as the platform is shared.
- Lesser flexibility & control over the cloud environment.

Hybrid Cloud

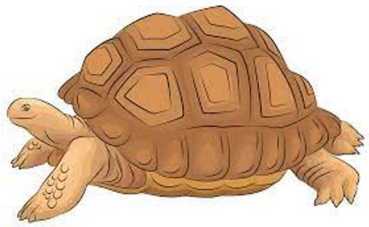
- Often called as 'the best of both worlds', it combines both public & private cloud.
- Greater flexibility & more deployment options.
- Cloud bursting is also possible.
- Network complexities & compliance issues.
- Can be extremely expensive.

Private Cloud

- Dedicated to a single organization.
- Higher security as the resources are not shared.
- Greater flexibility to control the cloud environment.
- Purchase and maintenance has to be bared by the organization
- Expensive than public cloud.

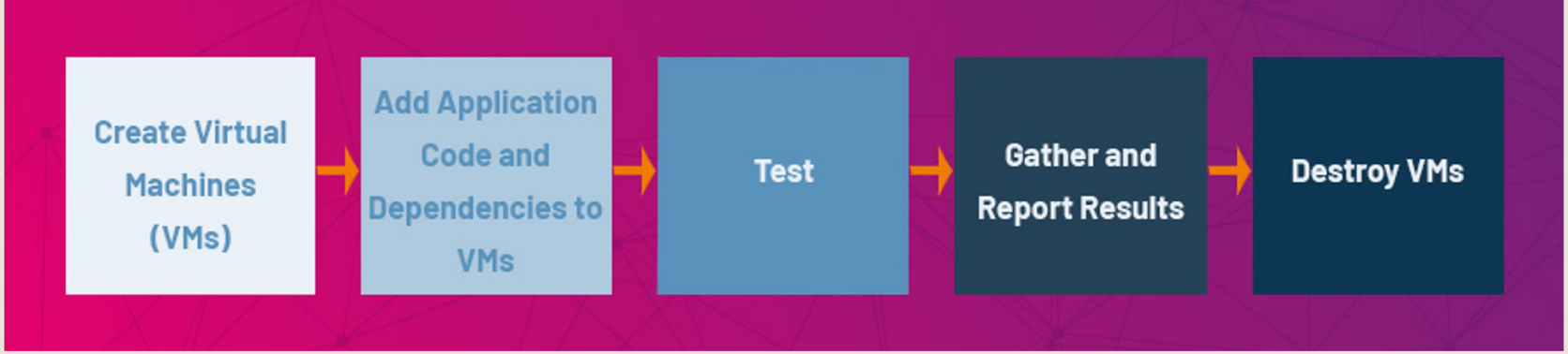
Leverage Provider Expertise:

- **Service providers** can create an **infrastructure** so staff can **focus on the application**
- This method appeals to **small firms and startups**
    - It's hard to picture one or two people constructing massive infrastructure when public cloud services are available
- **Public clouds** are run **more professionally** than **private clouds** because of **client competition**
- **Private clouds** can be handled **professionally**, but it's **hard on a small scale** without a **dedicated team**.



Get Started Quickly:

- **Cloud providers speed up launches.**
- If you **miss an opportunity** because you're **too slow**, your **savings won't help**.
- A **public cloud service** may be **more expensive** than **doing it yourself** but it may take **too long** and the **opportunity passes.**
- **Developing infrastructure** is a **waste of time** when product **success is uncertain.**
- **Early on**, a **product** may be **experimental** and iterate to **try new features.**
- **Early testers** can get a simple **product quickly** for **viability** using **public cloud services.**
- **Private infrastructure** can yield viable products later on



Implement Ephemeral Computing:

- **Short-term** projects
- Ephemeral computing uses a **massive computational infrastructure** for a **brief time**

Imagine a company's ad campaign draws millions of users to its website for a few weeks, then drops sharply. Using a public supplier allows the organization to quickly add thousands of devices and then discard them

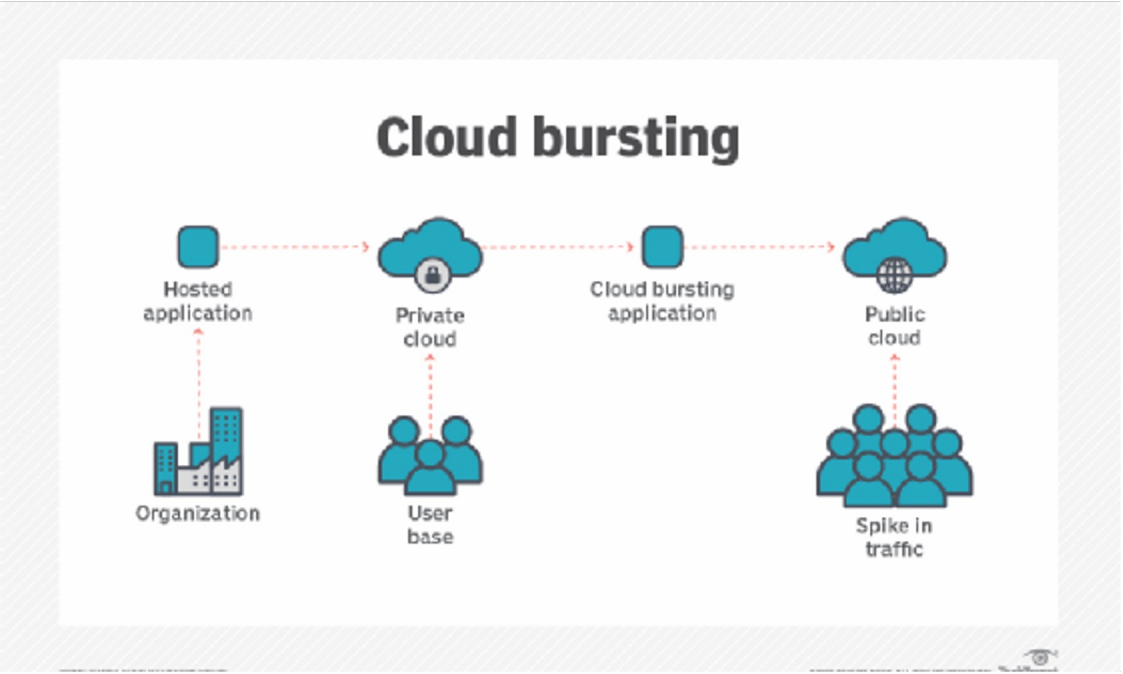
- For **massive data processing** and **analysis projects**, **proof-of-concept projects, biotechnology data mining,** and **unanticipated traffic spikes**
- It's **impractical** to develop a **massive infrastructure** for such a **short time**
- **cloud service provider** may **offer** such **computing facilities**
- Overall, **usage will be constant.**

Use the Cloud for Overflow Capacity:

Develop **baseline capacity in-house**

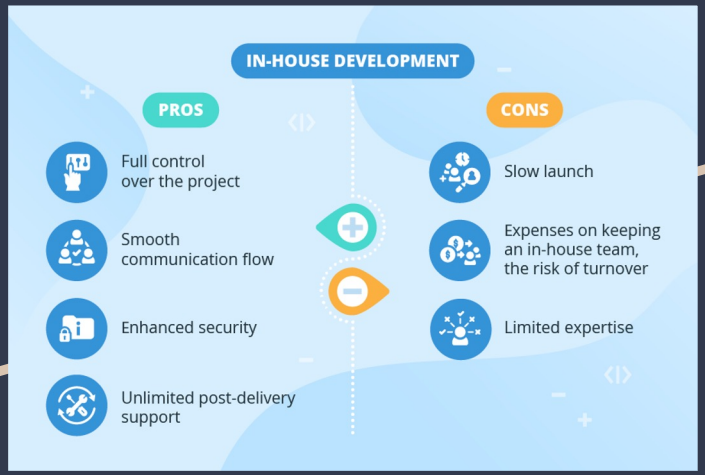
Use the **cloud** to **exceed your limits**

This is **more economical** than **building in-house capacity** or **utilizing a public supplier**.



Leverage Superior Infrastructure:

- Gain an edge with **excellent infrastructure**
- In this technique, **customized infrastructure** is used to obtain **competitive advantage**
- This may include **establishing your own datacenters** to **control cost** and **resource utilization** or **adopting IaaS over PaaS** to customize **OS and software.**



Develop an In-House Service Provider:

Create an **in-house service provider** to **control costs** and **ensure privacy**

Computing **infrastructures** are often **only cost-effective** at **very large scale**

which is why **public cloud providers** can **offer services** so **cheaply**

However, a **major corporation** can obtain **similar economies of scale** by constructing a **vast infrastructure** shared by **numerous in-house clients**

Because it's **in-house**, it's **private**—a criterion commonly required by **highly regulated companies**.

Contract for an On-Premises, Externally Run Service:

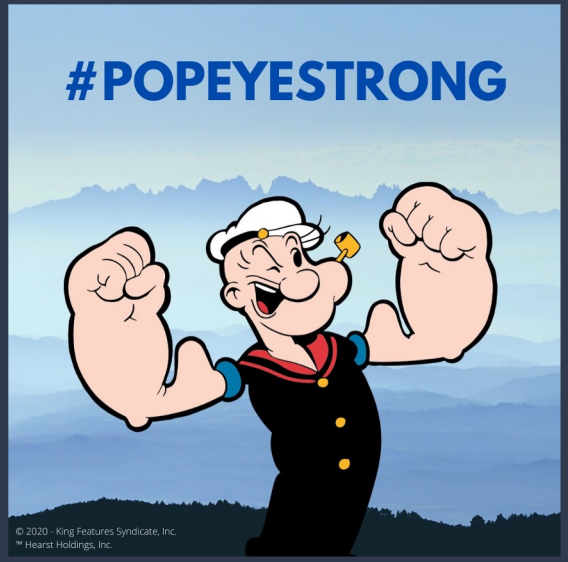
Some companies run **in-house cloud services**

Their **control and customization vary**

You **gain** from their **experience** and **reduce privacy risks** by **owning/controlling** the equipment.

Maximize Hardware Output:

- **Avoid virtualization** to boost **computing productivity** and **efficiency**
- When an **infrastructure contains** hundreds of **thousands of computers** or **millions of cores**, **1% efficiency** improvement is like **adding thousands of new computers**
- **Virtualization inefficiency** can be **costly**
- **Instead of virtual computers**, **actual machines** are employed and **services are densely packed** to **enhance utilization**.



- If you need sheer horse-power performance-sensitive applications and heavy workloads
- More processing power, I/Os per second, and network and disk performance
- single-tenant devices that leverage the whole hardware
- newest generation CPUs, RAM, and Non-Volatile Memory Express (NVMe) solid-state drives (SSDs)

Implement a Bare Metal Cloud:

- **public cloud service where the customer rents dedicated hardware resources from a remote service provider**
- Manage **hardware** like a **VM cloud**
- Use the **same API** for virtual and physical machines
- With planning, **virtual machine benefits** can be **applied** to **actual machines**.
- some businesses purchase **hundreds or thousands** of physical computers and **manage them as a pool** that can be **booked by departments or individuals**.
- This is accomplished by providing an **API for allocating computers**, **deleting and reinstalling OS, rebooting, controlling access, and returning them to the pool**
- Allocations may **not** be as **fast or dynamic as virtual machines**, but numerous of the **same benefits are achieved**.

Kahoot quiz