CLOUD COMPUTING

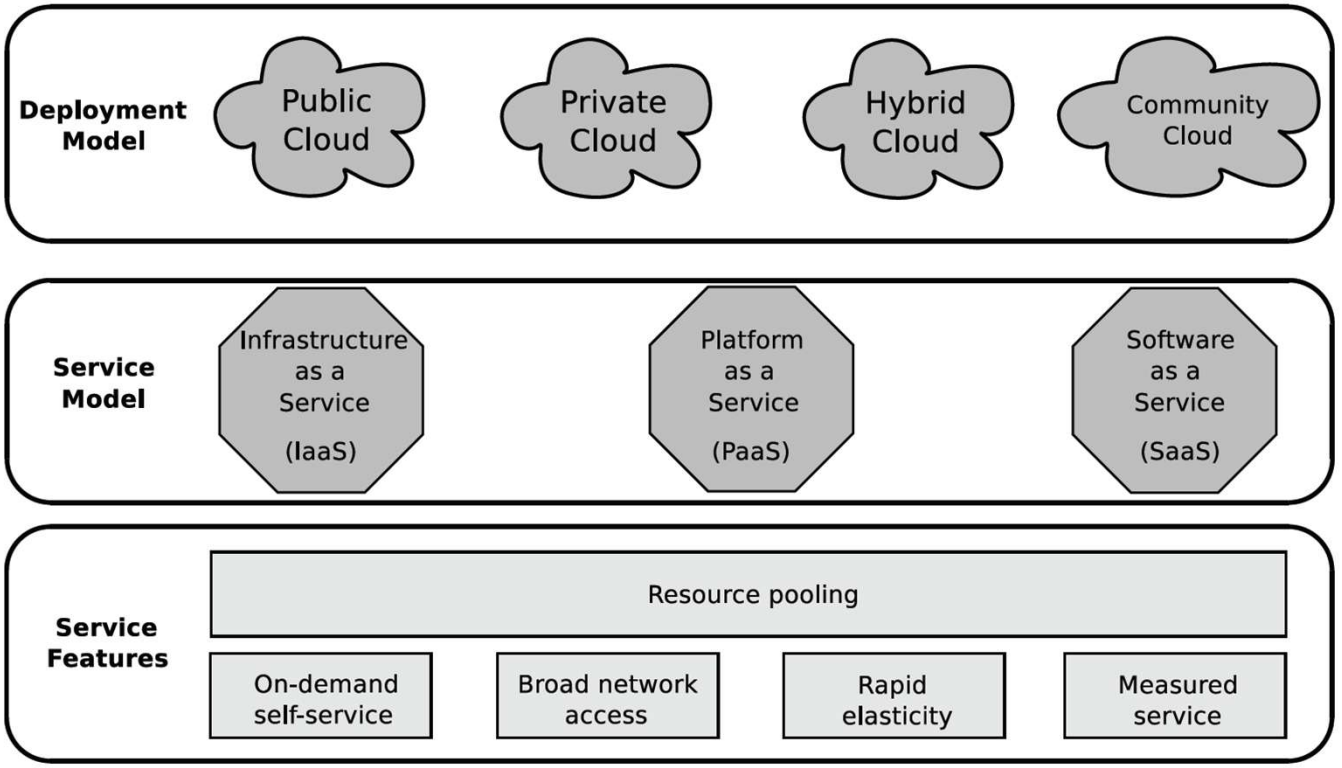
THE RETURN OF UTILITY COMPUTING

THE RETURN OF UTILITY COMPUTING

- “If computers of the kind I have advocated become the computers of the future, then computing may someday be organized as a public utility just as the telephone system is a public utility… The computer utility could become the basis of a new and important industry.” (John McCarthy, Turing Award, 1961)

John McCarthy (1927–2011) received the Turing Award in 1971 and was the inventor of Lisp and a pioneer of timesharing large computers. Clusters of commodity hardware and the spread of fast networking have helped make his vision of timeshared “utility computing” a reality.

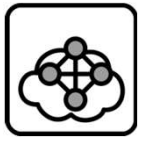
NIST DEFINITION OF CLOUD COMPUTING



DEPLOYMENT MODELS

- Public Cloud
  - The cloud is provisioned for open use by the general public

- Private Cloud
  - The cloud infrastructure is provisioned for exclusive use by a single organization.



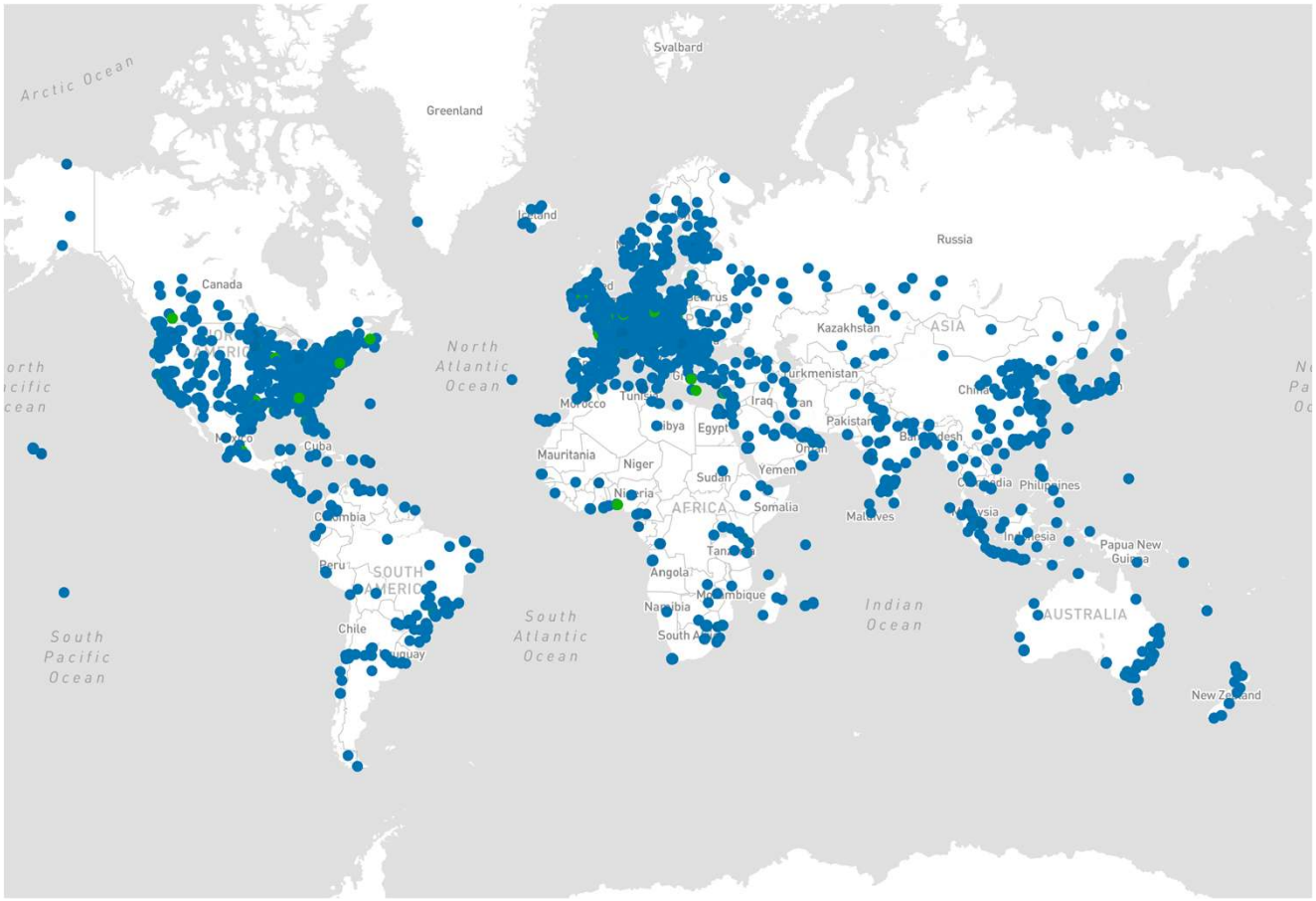
- Community Cloud
  - The cloud infrastructure is a composition of two or more distinct cloud infrastructures

- Hybrid Cloud
  - The cloud infrastructure is a composition of public and private clouds

PRIVATE CLOUD VS ≠ DATA CENTER

- There are many datacenter in the world (see https://www.datacentermap.com/)
- Some of them offers a housing or co-location service
- However, a data center does not necessarily imply a private cloud computing model
- This requires considering how computing resources are managed, as discussed next

DATA CENTER MAP (HTTPS://WWW.DATACENTERMAP.COM/)



SERVICE MODELS

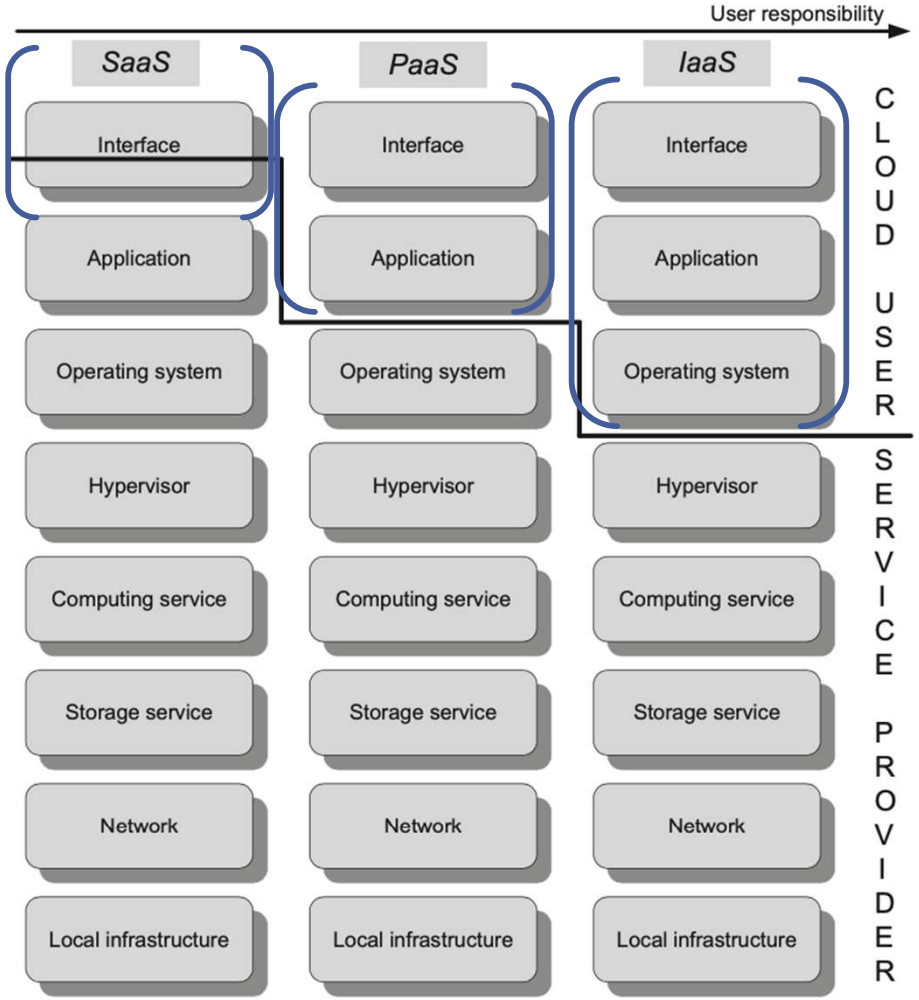
- IaaS Infrastructure as a Service

- PaaS: Platform as a Service

- SaaS: Software as a Service

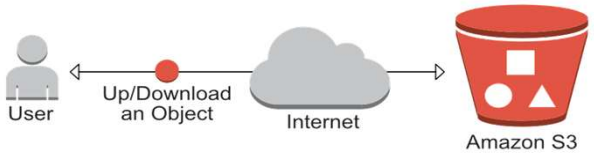
SERVICE MODEL – LAYERS

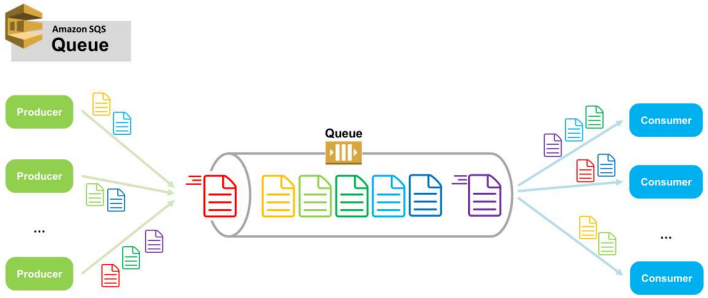
- The layered scheme represents the different components of an information system that provides a service.
- Any layer uses the layer below..
- IaaS Provides the highest flexibility and control
  - Allows to provision (virtualized) hw (such as servers, networks, ..) very quickly, scale them, shutdown,etc
  - Example: VM provisioned in some minute
- PaaS offers a way to develop sw without warrying about any management issue (os updates, patching,)…
  - Example: a sw that allows to create a web site
- SaaS is a full sw that the providers runs for the user
  - Example: email



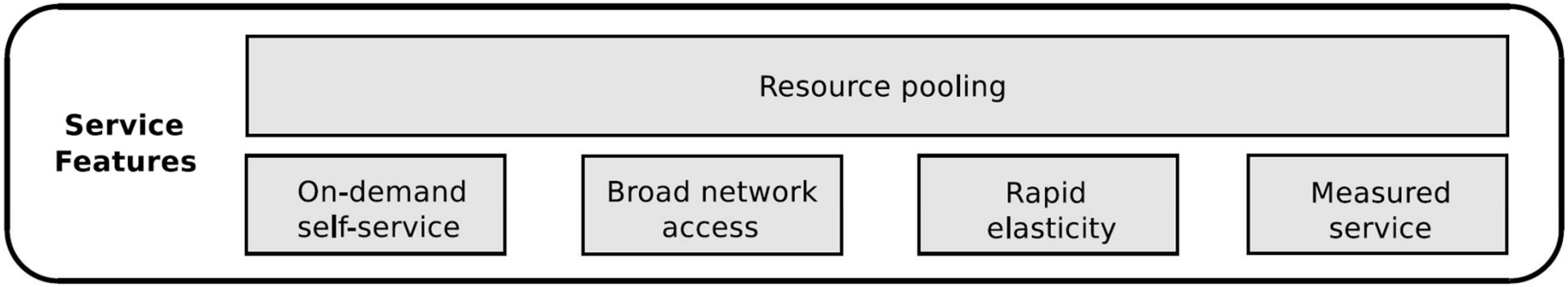
THE INITIAL COMPUTING SERVICE FROM AWS

- Cloud computing age started in 2006 with Amazon Web Services (AWS)'s Elastic Cloud Computing (**EC2**) and Simple Storage Systems (**S3**), plus a Simple Queue Service (**SQS**)
- **EC2** service allows to use provision size variable VMs
- **S3** is an object storage service (data are stored as objects into beckets), e.g. images, or web site . Ensure scalability, data availability, security
- **SQS**, messaging system used to connect other components





SERVICE FEATURES



Resource pooling

Computing resources are pooled to serve multiple consumers.

On-demand self-service

Consumer can provision computing capabilities automatically.

Broad network access

Capabilities are available and accessed over the network.

Rapid elasticity

Capabilities can be elastically provisioned to scale with demand.

Measured service

Automatically controlled and optimized resources with metering.

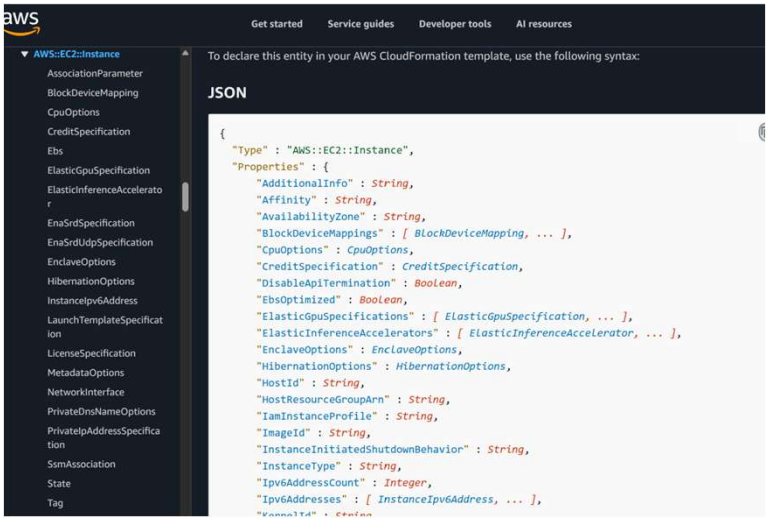
MORE ABOUT THE NAME

- The name *"Cloud Computing"* evokes the cloud as an iconic representation of the Internet (a cloud is often used to represent a network), and *computing* to indicate the resources required to perform computation, such as storage, CPUs, and memory.
- In simple terms, cloud computing refers to the shift of computing from a single server or data center to an equivalent service accessed via the Internet.

1. ON-DEMAND SELF-SERVICE

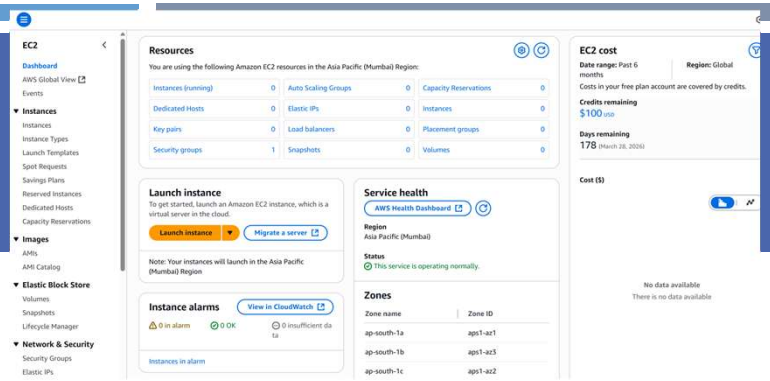
- The on-demand service feature refers to the ability to consume the computing facility as much as needed at any moment, i.e., through a user-friendly UI or programmatically, e.g. python script, or Infrastructure as Code (IaC)
- The requested cloud services is provisioned in a short period of time and without any need of human intervention at vendor's end.

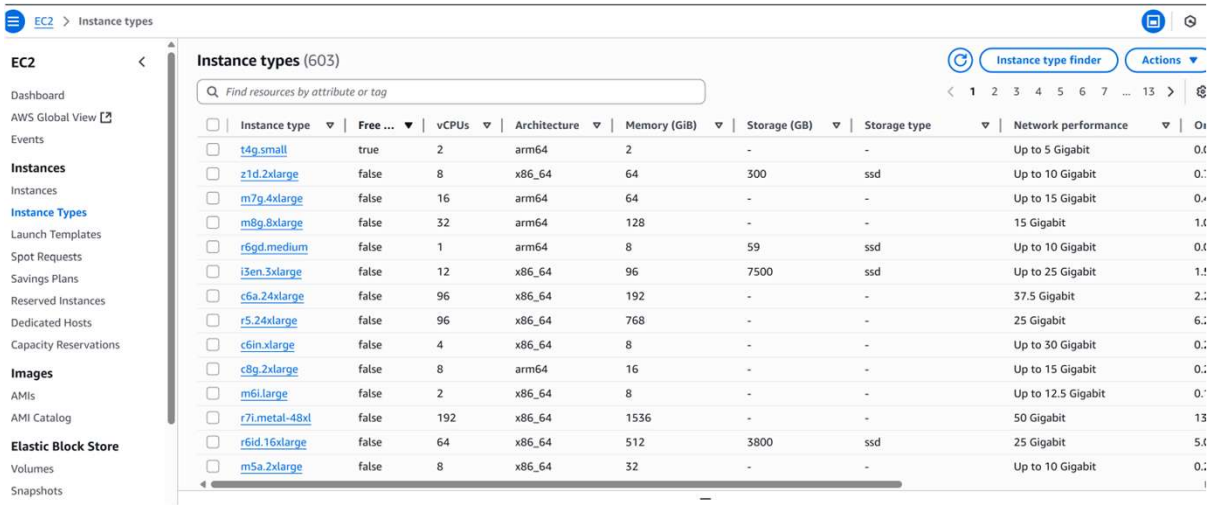
```
{
    "Resources": {
        "Instance": {
            "Type": "AWS::EC2::Instance",
            "Properties": {
                "AvailabilityZone": "us-east-1a",
                "InstanceType": "c3.large"
            }
        }
    }
}
```



1. EXAMPLE FROM AWS (1/3)

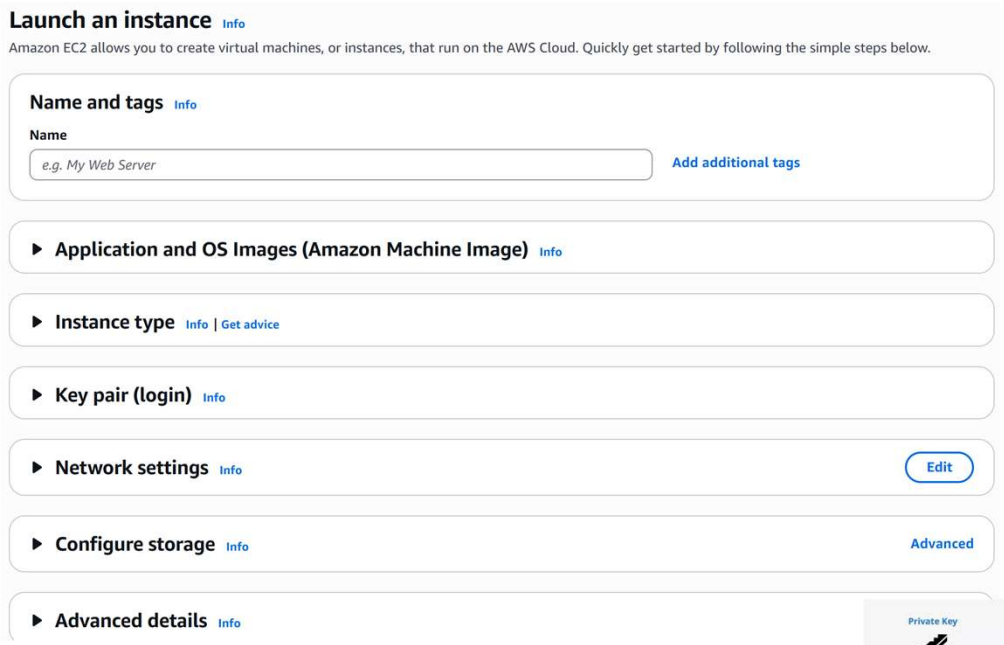
1. Sign in to the management console (aws)
2. Open EC2 console choosing EC2 under compute
3. Select a AWS Region
4. Lunch the instance

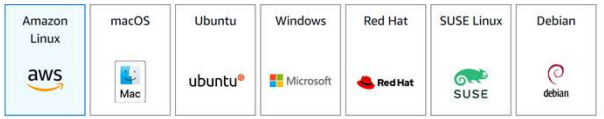




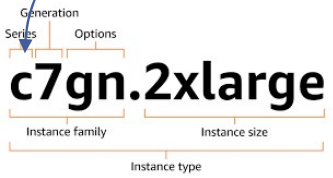
1. EXAMPLE FROM AWS (2/3)

Lunch the instance





General purpose
Compute optimized (C)
Memory optimized ..

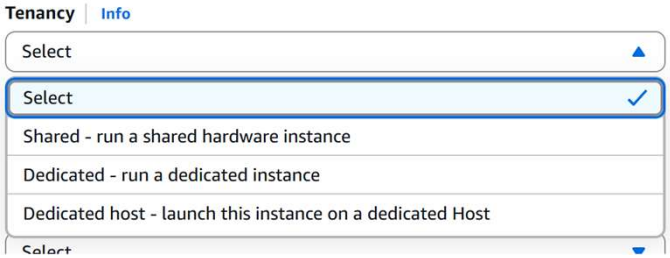


Key pair generation to access via ssh

Assign a public IP and define a security group (a security group is like a firewall, initially all traffic is denied)

1. EXAMPLE FROM AWS (3/3)

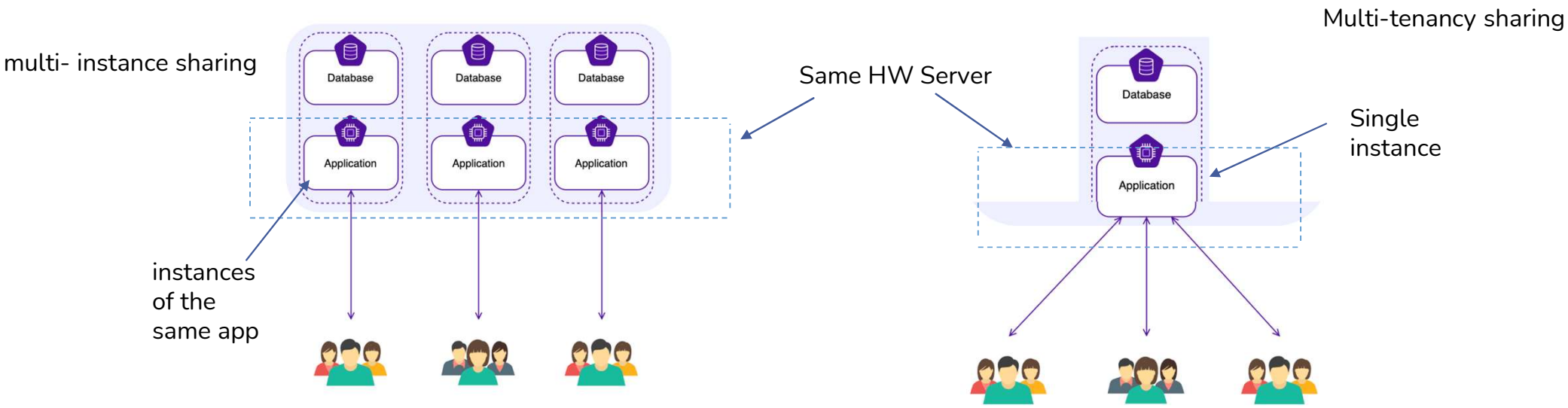
- Among other aspects, in the advanced setting is it possible to specify the Tenancy option
- Shared tenant: default, means that the VM shares the same HW with other tenants from which it is isolated
- Dedicated Instance: the instance run on HW only used by the customer
- Dedicated Host:the customer has full visibility of the underlying hardware. In particular, it allows the use of licensed software with hardware-based licensing requirements, by bringing your own licenses (BYOL).



2. RESOURCE POOLING

- The provider manages its computing resources as a **pool**, accessed by all the users
- Even if not mentioned in the NIST document **multi-tenancy** is very important in cloud computing
- Multi-tenancy means that multiple independent users (tenants), who are not related to each other, share the same application or infrastructure, while keeping their data and configurations logically isolated.
- In simple form it means that a single set of resources has multiple tenants who are not linked with each other.
- For example, web-server multi-tenancy means that the same server runs multiple unrelated sites
- Hardware resources are shared by tenant using hw virtualization

2. RESOURCE POOLING (MULTI-TENANCY)



3. MEASURED SERVICE

- Service Level Agreement (SLA): Formal agreement (contract) between a provider and a consumer of a service, defining
- Service Level Indicator (SLI): metric that can be monitored, e.g., response time, or uptime
- Service Level Objective (SLO): a predicate over a set of SLIs condition on a measure of a specific metric (e.g., mean response time <= 1 s)
- penalty and/or compensation in case of SLA violation

3. MEASURED SERVICE

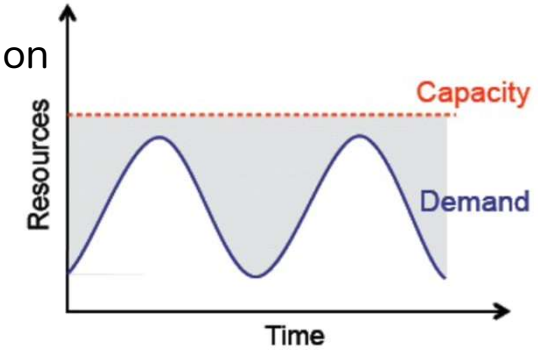
- Uptime: the most common SLI for Cloud services: https://uptime.is/

- Example of SLA is a fixed threshold on a single SLI,
- "monthly uptime percentage for a VM is at least 99.99%" (system-level indicator)
- "At least 90% or requests have a response time less than 100 ms" (application-level indicator)
  - Not given by cloud providers

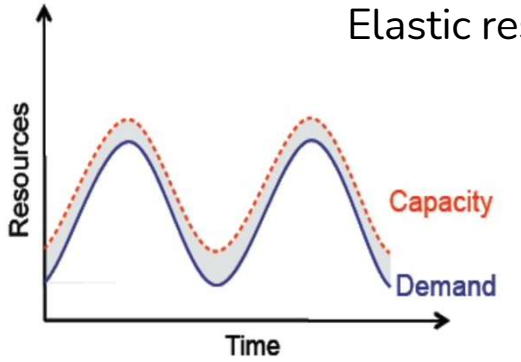
4. RAPID ELASTICITY

- Elasticity is the degree to which a system can adapt to workload changes by provisioning and de-provisioning resources in an autonomic manner, such that at each point in time the available resources match the current demand as closely as possible

No adaptation

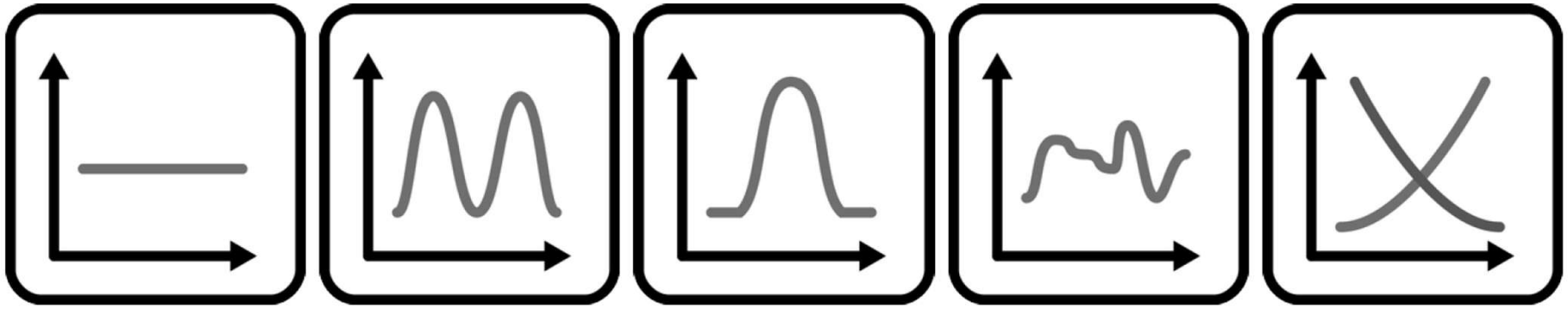


Elastic resource provisioning



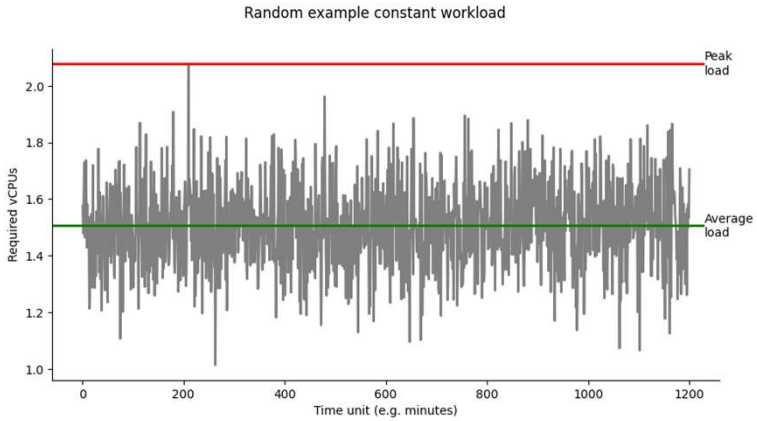
4. RAPID ELASTICITY – TYPE OF WORKLOADS

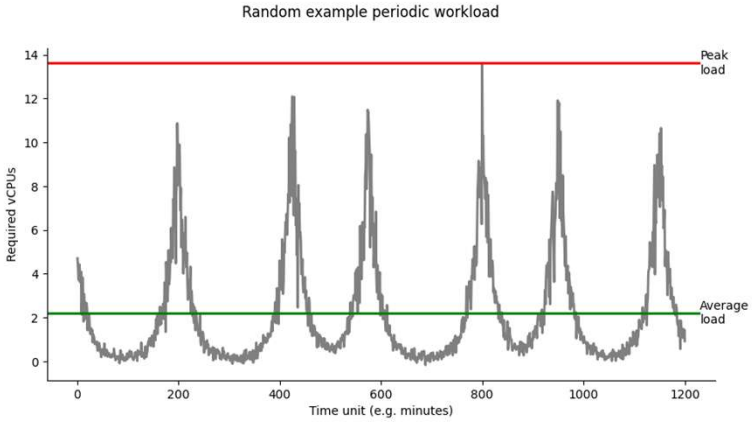
The demand of service can change in different ways

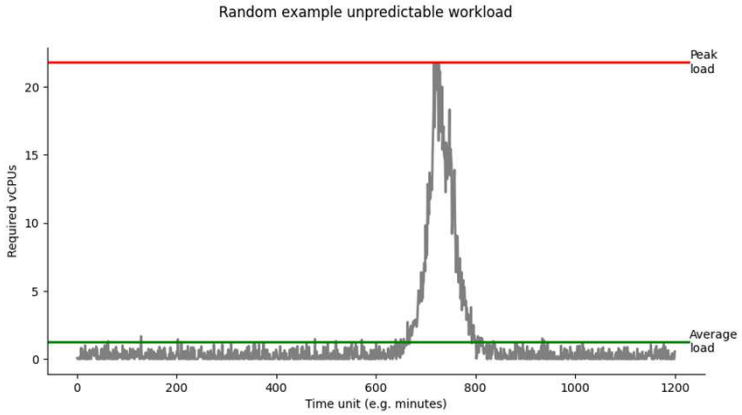


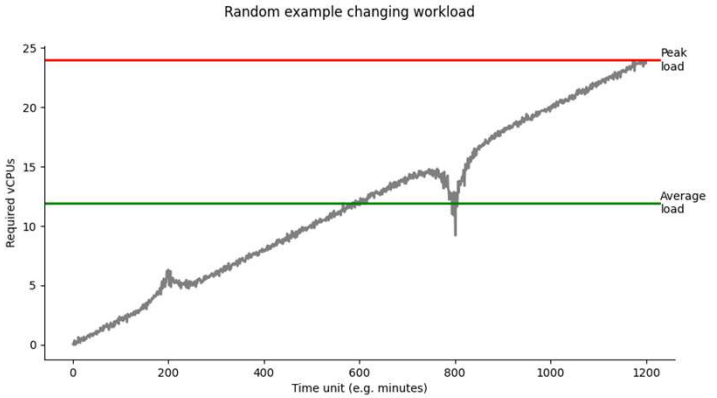
(a) Static (b) Periodic (c) Once-in-a-lifetime (d) Unpredictable (e) Continuously changing

4. RAPID ELASTICITY – TYPE OF WORKLOADS



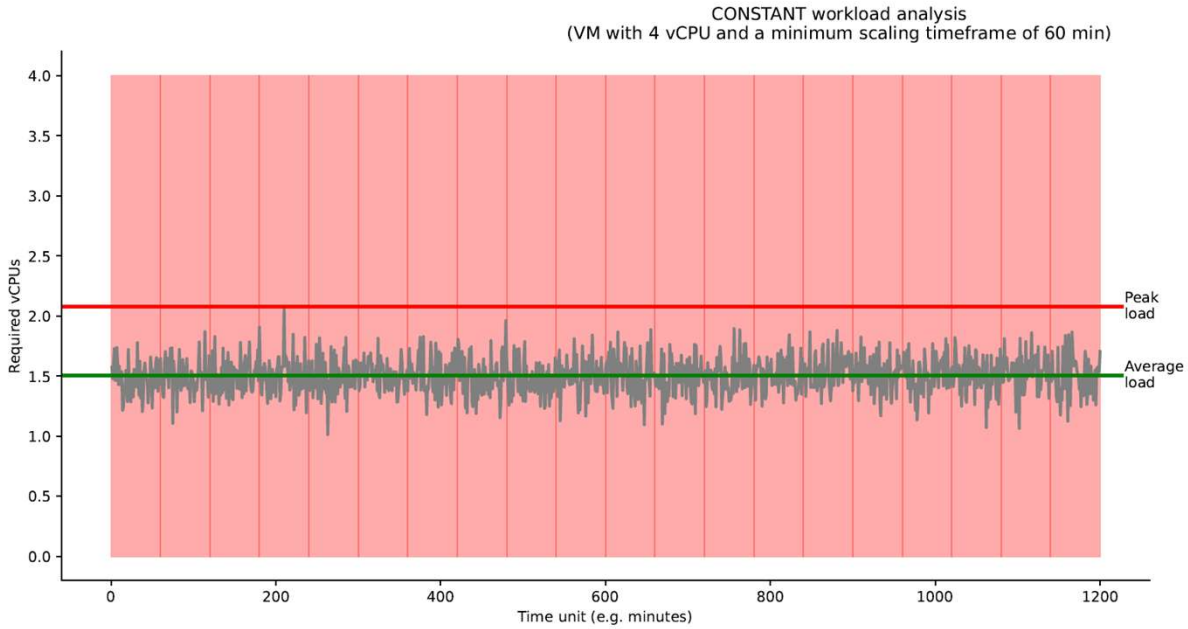


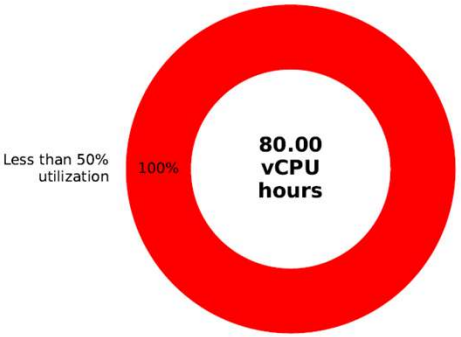




https://git.mylab.th-luebeck.de/cloud-native/lab-workload-analysis

4. RAPID ELASTICITY – TYPE OF WORKLOADS: CONSTANT

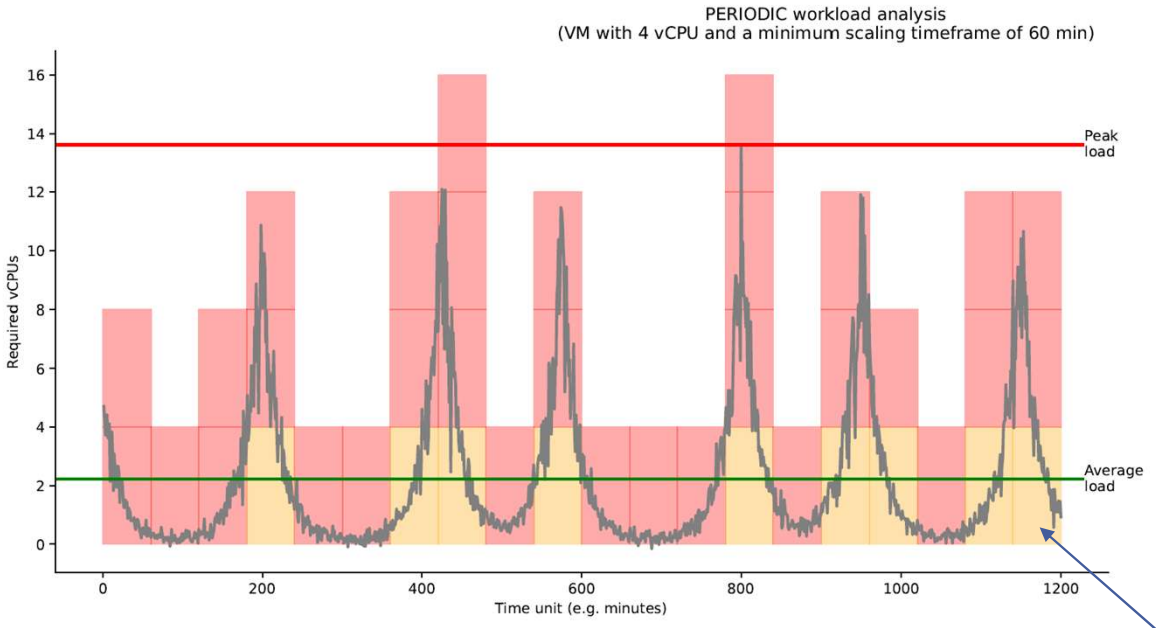


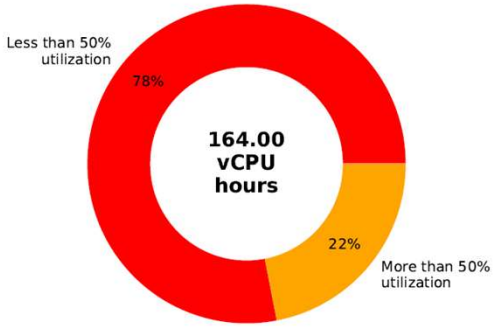


1200 min = 20 hours

4 CPU x 20 = 80 CPU

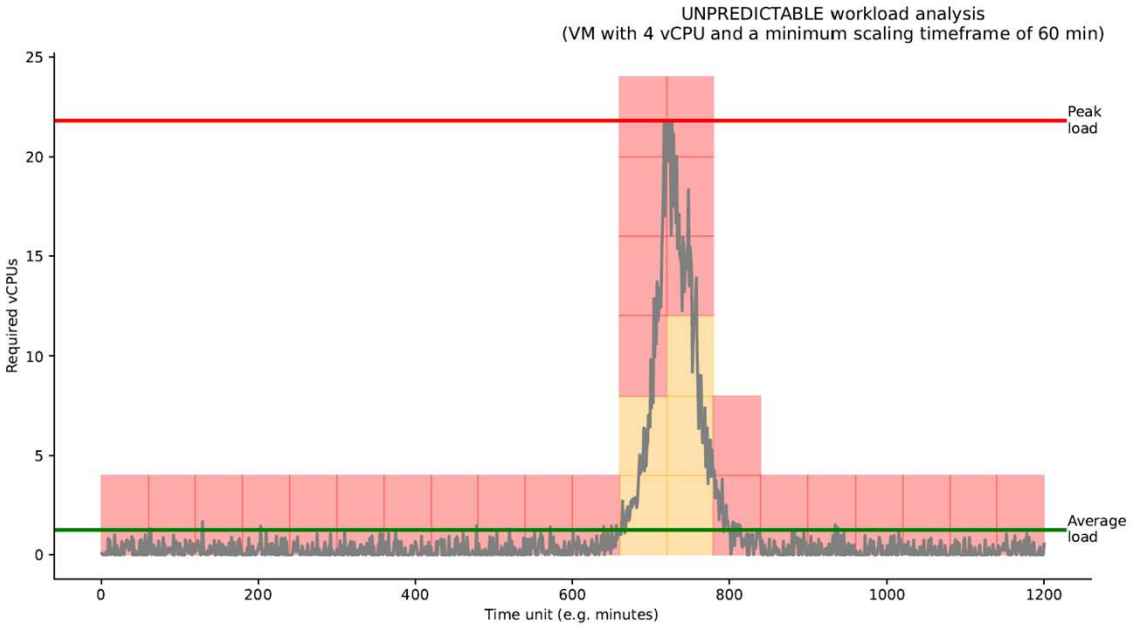
4. RAPID ELASTICITY – TYPE OF WORKLOADS: PERIODIC

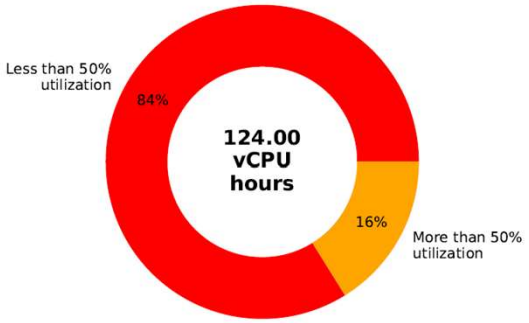




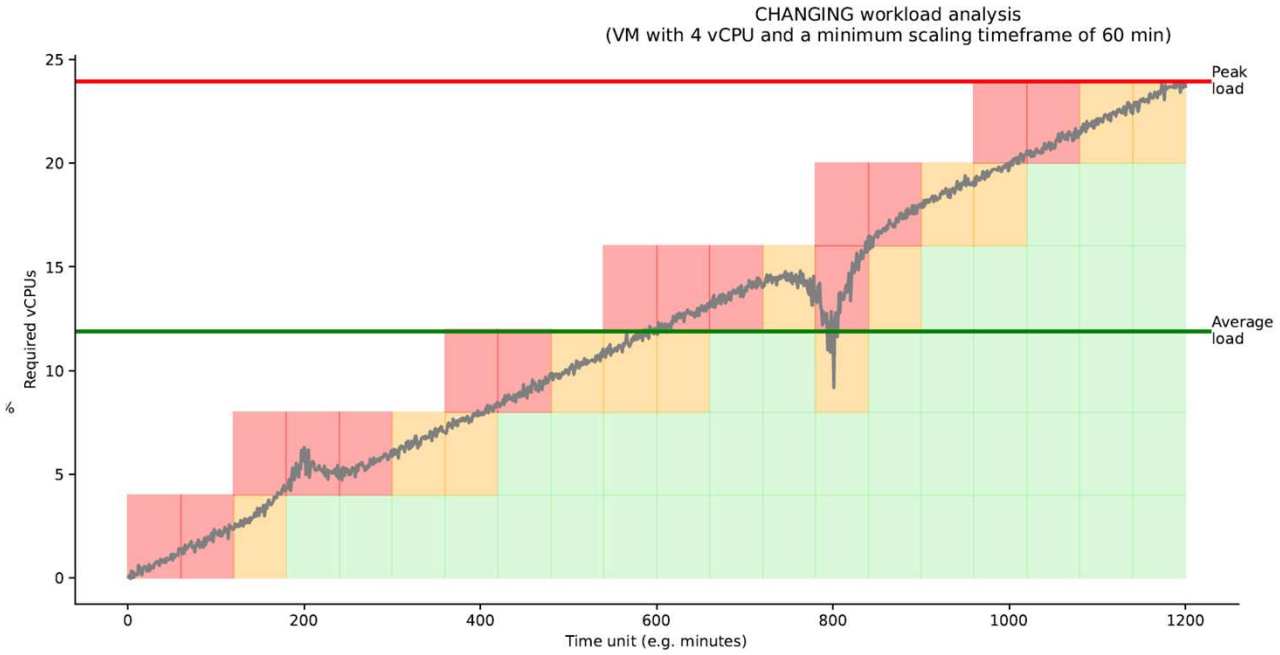
More than half vCPU are used

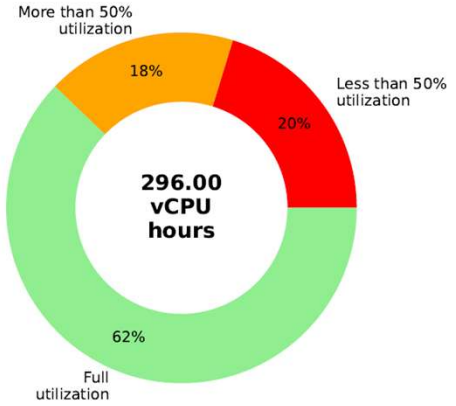
4. RAPID ELASTICITY – TYPE OF WORKLOADS: UNPREDICTABLE





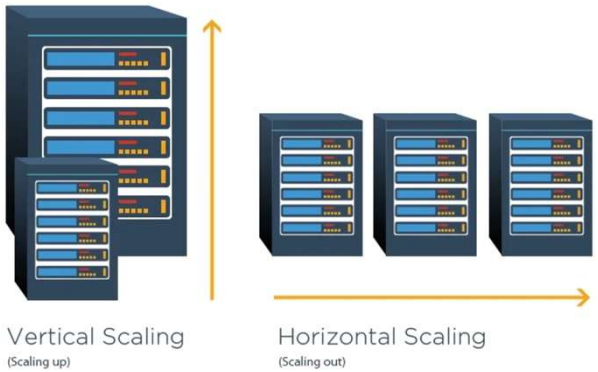
4. RAPID ELASTICITY – TYPE OF WORKLOADS; CHANGING (REGULAR)





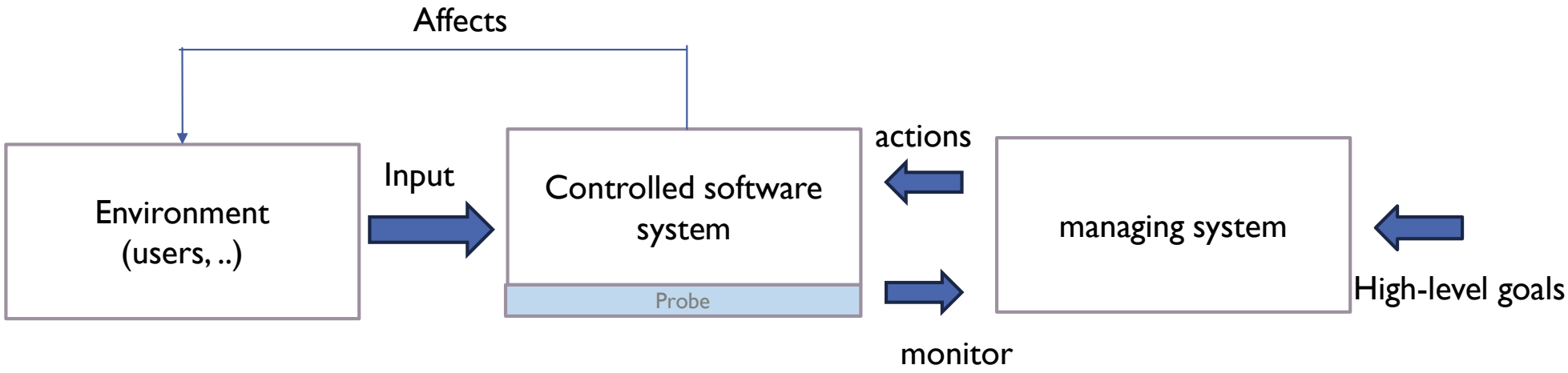
ELASTICITY AND AUTO-SCALING

- Goal: allocate the minimum number of instances that meet the required SLA (this is called **horizontal scaling**)
- But how?
- Unless the traffic is regular (e.g. periodic), the main strategy is to forecast the demand of resources
- Then, scaling can be done manually (not practical) or automatically
- Automatic scaling aka **auto-scaling** can be seen as a feature of **self-adaptive** software systems



MORE ON AUTO-SCALING

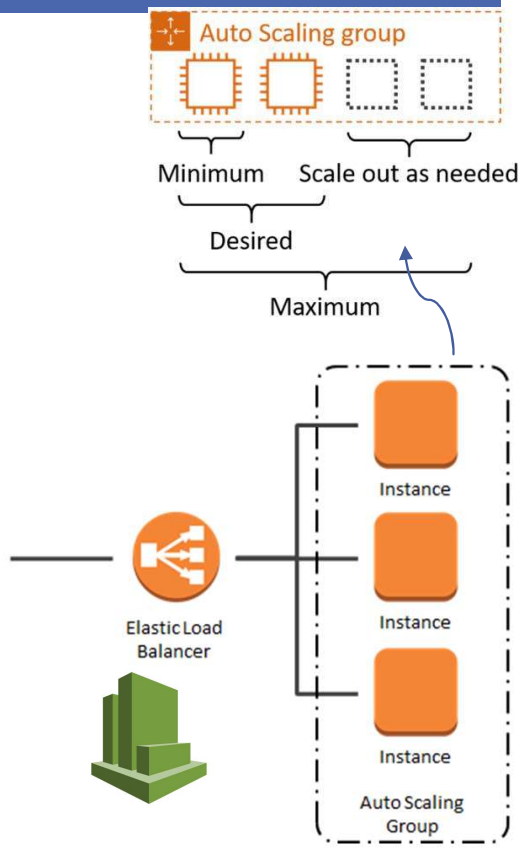
- Autoscaling can be seen as a feature of a self-adaptive system.



Conceptual model of a self-adapting system

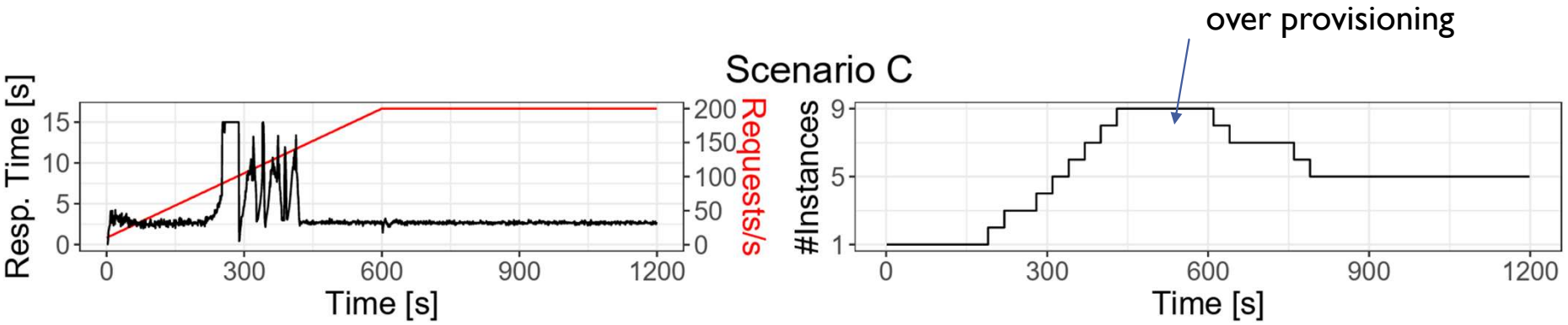
EXAMPLE: AUTOSCALING IN AWS

- EC2 Autoscaling is a service that automatically scales the capacity of EC2 up or down according to user-defined monitored conditions
- For example, the number of replicas can be increased during a spike in the application workload to meet the performance requirements and scaled down when the workload decreases
- The autoscaling Service defines the policy to follow(how to add/remove replicas)
  - if average CPU utilization of all instances > 70% in last 1 minute, then add 1 new instance
  - if average CPU utilization of all instances is <35% in last 5 min, then remove 1 instance
- It also requires
- A load balancer service to distribute requests (Elastic Load Balancing, ELB)
- A monitoring service (CloudWatch)
  - Metric to monitor (avg CPU, connection bandwidth)
- Instances must belong to a group (Auto Scaling group)



AUTOSCALING (OVER-PROVISIONING)

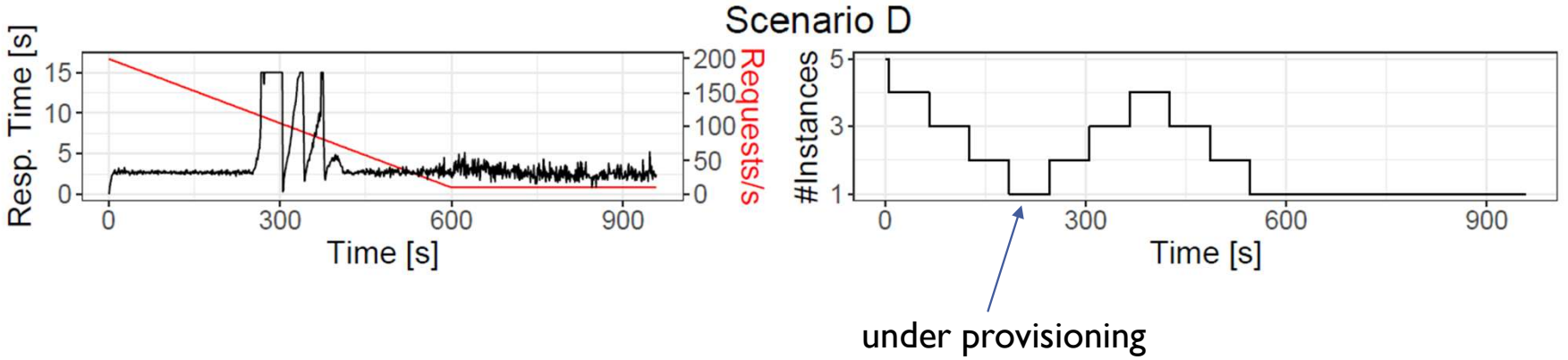
- Although auto-scalers are indispensable parts of modern cloud deployments and determine the service quality and cost of a cloud application in dynamic workloads, effective tuning is not trivial.
- The following plots report an experiment of an auto-scaler taking 30 s to scale and a new instance is lunched if the CPU load is higher than 80%, and removed if less than 25%. This result in over-provisioning because when the traffic is low again the number of replicas remains high to 5. The auto-scaler takes a new decision before the effect of the previous decision is visible (see * for details)



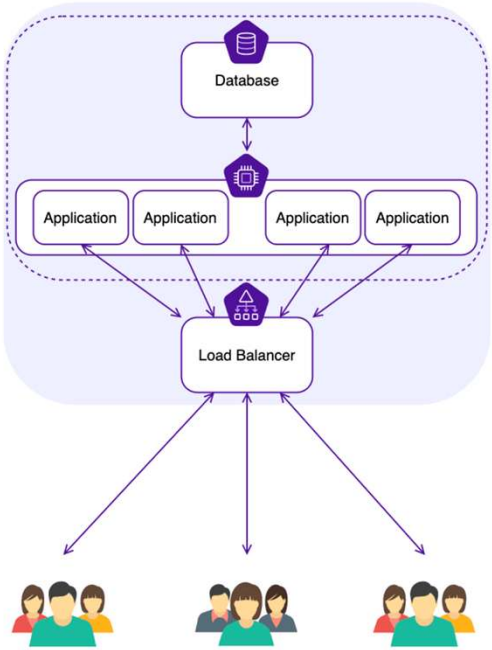
(*) Strasser et al., Autoscaler Evaluation and Configuration: A Practitioner's Guideline, ICPE 2023 https://research.spec.org/icpe_proceedings/2023/proceedings/p31.pdf

AUTOSCALING (UNDER-PROVISIONING)

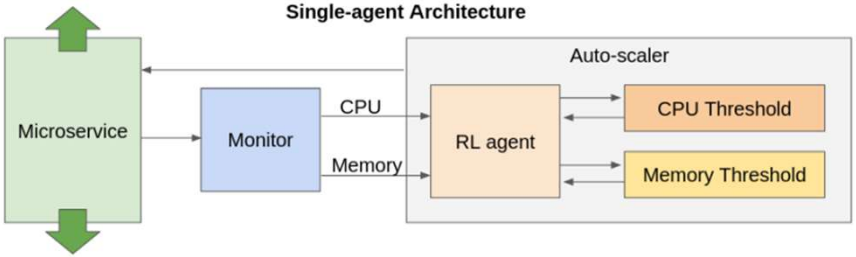
In this case the load decreases linearly, the auto-scaler initially removes too many replicas (the deploy time is 60 s) and then it adds replicas again



AUTO-SCALING CHALLENGES

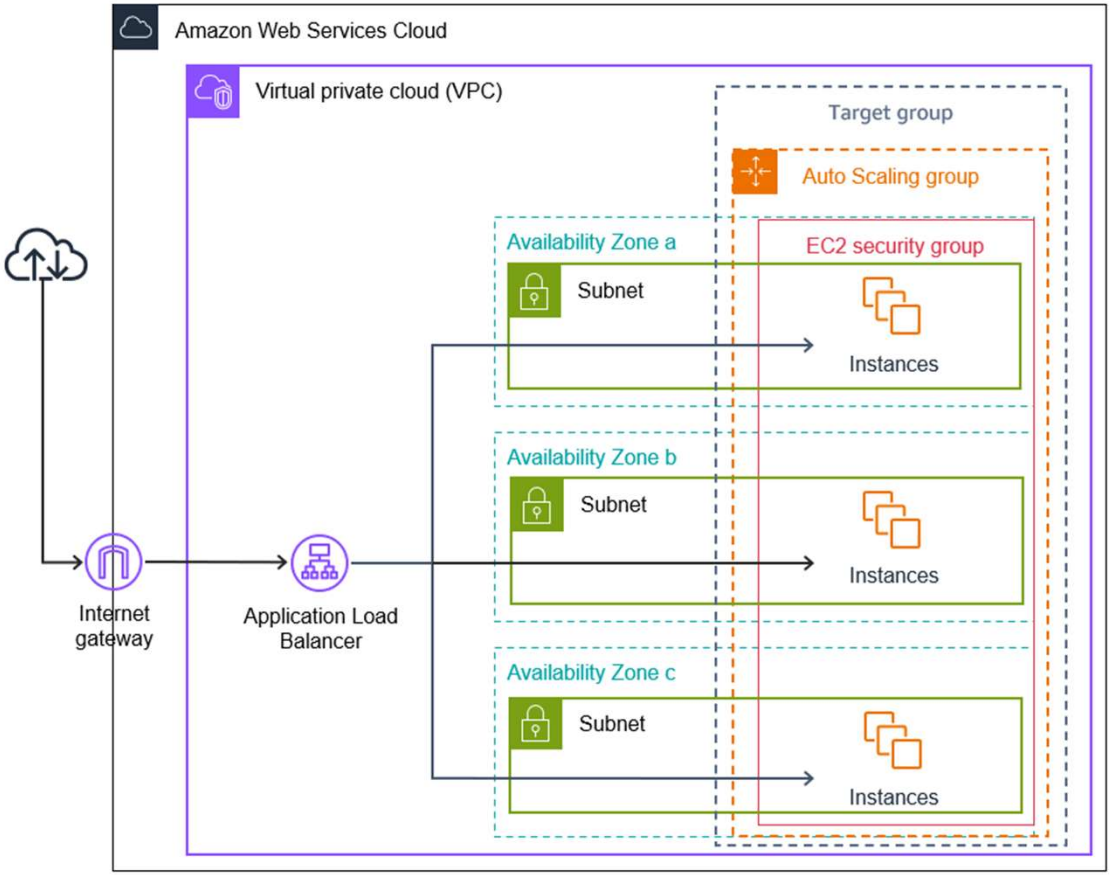


- Scaling a stateful service is even more challenging.
- In recent years, ML based auto-scaler is becoming studied
- Reinforcement Learning seeks to identify the optimal scaling policy, e.g. it modifies the thresholds



ANOTHER EXAMPLE IN AWS

- An Availability Zone (AZ) in AWS is one or more physically separate data center facilities within an AWS geographic region.
- Each AZ is designed to be independent and isolated from failures that could affect other AZs in the same region, providing high availability and fault tolerance.
- For example, in Europe there are 24 zones



CLOUD MONITORING

- Goal: track health of system and services deployed
- Tools allow to measure system-oriented metrics (CPU utilization, Disk utilization and throughput – MB/s, Memory-use,free-memory, Network interface)
  - Most providers offer monitoring tools (CloudWatch); other tools exist (e.g., Prometheous)
- Hard to measure application-oriented metrics, like the response time

- SLA sometimes not clear, e.g. Availability is an average and hides short outages over a long mean
- Not easy to detect SLA violation
- Data durability never 100% (it's the user responsibility)

5. BROADBAND ACCESS

- Cloud resources accessed over Internet using standard access mechanisms that provide platform-independent access, e.g. published service interface/API

MAIN CLOUD PROVIDERS