**CLOUD COMPUTING**

**Introduction:**

Cloud computing is the delivery of computing services including servers, storage, databases, networking, software, analytics, and intelligence over the Internet (“the cloud”) to offer faster innovation, flexible resources, and economies of scale. You typically pay only for cloud services you use, helping lower your operating costs, run your infrastructure more efficiently and scale as your business needs change.

**Top Benefits of Cloud Computing:**

Cloud computing is a big shift from the traditional way businesses think about IT resources. Here are seven common reasons organisations are turning to cloud computing services

**Cost:**

Cloud computing eliminates the capital expense of buying hardware and software and setting up and running on-site datacenters—the racks of servers, the round-the-clock electricity for power and cooling, the IT experts for managing the infrastructure. It adds up fast.

**Speed:**

Most cloud computing services are provided self service and on demand, so even vast amounts of computing resources can be provisioned in minutes, typically with just a few mouse clicks, giving businesses a lot of flexibility and taking the pressure off capacity planning.

**Global scale:**

The benefits of cloud computing services include the ability to scale elastically. In cloud speak, that means delivering the right amount of IT resources.

For example, more or less computing power, storage, bandwidth right when it is needed and from the right geographic location.

**Productivity:**

On-site datacentres typically require a lot of “racking and stacking” hardware setup, software patching, and other time-consuming IT management chores. Cloud computing removes the need for many of these tasks, so IT teams can spend time on achieving more important business goals.

**Performance:**

The biggest cloud computing services run on a worldwide network of secure datacentres, which are regularly upgraded to the latest generation of fast and efficient computing hardware. This offers several benefits over a single corporate datacentre, including reduced network latency for applications and greater economies of scale.

**Reliability:**

Cloud computing makes data backup, disaster recovery and business continuity easier and less expensive because data can be mirrored at multiple redundant sites on the cloud provider’s network.

**Security:**

Many cloud providers offer a broad set of policies, technologies and controls that strengthen your security posture overall, helping protect your data, apps and infrastructure from potential threats.

**Types of Cloud Computing:**

Not all clouds are the same and not one type of cloud computing is right for everyone. Several different models, types and services have evolved to help offer the right solution for your needs.

First, you need to determine the type of cloud deployment or cloud computing architecture, that your cloud services will be implemented on. There are three different ways to deploy cloud services: on a public cloud, private cloud or hybrid cloud.

**Public Cloud:**

Public clouds are owned and operated by a third-party [cloud service providers](https://azure.microsoft.com/en-in/overview/choosing-a-cloud-service-provider/), which deliver their computing resources like servers and storage over the Internet. Microsoft Azure is an example of a public cloud. With a public cloud, all hardware, software and other supporting infrastructure is owned and managed by the cloud provider. You access these services and manage your account using a web browser.

**Private Cloud:**

A private cloud refers to cloud computing resources used exclusively by a single business or organisation. A private cloud can be physically located on the company’s on-site datacentre. Some companies also pay third-party service providers to host their private cloud. A private cloud is one in which the services and infrastructure are maintained on a private network.

**Hybrid cloud:**

Hybrid clouds combine public and private clouds, bound together by technology that allows data and applications to be shared between them. By allowing data and applications to move between private and public clouds, a hybrid cloud gives your business greater flexibility, more deployment options and helps optimise your existing infrastructure, security and compliance.

**Types of cloud services: IaaS, PaaS, serverless and SaaS**:

Most cloud computing services fall into four broad categories: infrastructure as a service (IaaS), platform as a service (PaaS), serverless and software as a service (SaaS). These are sometimes called the cloud computing stack because they build on top of one another. Knowing what they are and how they are different makes it easier to accomplish your business goals.

* **Infrastructure as a service (IaaS)**

The most basic category of cloud computing services. With IaaS, you rent IT infrastructure servers and virtual machines (VMs), storage, networks, operating systems from a cloud provider on a pay-as-you-go basis.

* **Platform as a service (PaaS)**

Platform as a service refers to cloud computing services that supply an on-demand environment for developing, testing, delivering and managing software applications. PaaS is designed to make it easier for developers to quickly create web or mobile apps, without worrying about setting up or managing the underlying infrastructure of servers, storage, network and databases needed for development.

* **Serverless computing**

Overlapping with PaaS, serverless computing focuses on building app functionality without spending time continually managing the servers and infrastructure required to do so. The cloud provider handles the setup, capacity planning and server management for you. Serverless architectures are highly scalable and event-driven, only using resources when a specific function or trigger occurs.

* **Software as a service (SaaS)**

Software as a service is a method for delivering software applications over the Internet, on demand and typically on a subscription basis. With SaaS, cloud providers host and manage the software application and underlying infrastructure and handle any maintenance, like software upgrades and security patching. Users connect to the application over the Internet, usually with a web browser on their phone, tablet or PC.

**Uses of Cloud Computing:**

You are probably using cloud computing right now, even if you don’t realise it. If you use an online service to send email, edit documents, watch movies or TV, listen to music, play games or store pictures and other files, it is likely that cloud computing is making it all possible behind the scenes. The first cloud computing services are barely a decade old, but already a variety of organisations—from tiny start-ups to global corporations, government agencies to non-profits—are embracing the technology for all sorts of reasons.

Here are a few examples of what is possible today with cloud services from a cloud provider:

* **Create Cloud-Native Applications**

Quickly build, deploy and scale applications web, mobile and API. Take advantage of [cloud-native](https://azure.microsoft.com/en-in/overview/cloudnative/) technologies and approaches, such as containers, [Kubernetes](https://azure.microsoft.com/en-in/topic/what-is-kubernetes/), microservices architecture, API-driven communication and DevOps.

* **Test and Build Applications**

Reduce application development cost and time by using cloud infrastructures that can easily be scaled up or down.

* **Store, Back up and Recover Data**

Protect your data more cost-efficiently—and at massive scale—by transferring your data over the Internet to an offsite cloud storage system that is accessible from any location and any device.

* **Analyse Data**

Unify your data across teams, divisions and locations in the cloud. Then use cloud services, such as machine learning and artificial intelligence, to uncover insights for more informed decisions.

* **Stream Audio and Video**

Connect with your audience anywhere, anytime, on any device with high-definition video and audio with global distribution.

* **Embed Intelligence**

Use intelligent models to help engage customers and provide valuable insights from the data captured.

* **Deliver Software on Demand**

Also known as software as a service (SaaS), on-demand software lets you offer the latest software versions and updates around to customers anytime they need, anywhere they are.

Content from, [ **https://azure.microsoft.com/en-in/overview/what-is-cloud-computing/#uses** ]

**Kubernetes:**

**Introduction:**

Kubernetes (also known as k8s or "kube") is an [open source](https://www.redhat.com/en/topics/open-source/what-is-open-source) container orchestration platform that automates many of the manual processes involved in deploying, managing, and scaling containerized applications.

In other words, you can cluster together groups of hosts running Linux® containers, and Kubernetes helps you easily and efficiently manage those clusters.

[Kubernetes clusters](https://www.redhat.com/en/topics/containers/what-is-a-kubernetes-cluster) can span hosts across on-premise, [public](https://www.redhat.com/en/topics/cloud-computing/what-is-public-cloud), [private](https://www.redhat.com/en/topics/cloud-computing/what-is-private-cloud), or [hybrid clouds.](https://www.redhat.com/en/topics/cloud-computing/what-is-hybrid-cloud) For this reason, Kubernetes is an ideal platform for hosting [cloud-native applications](https://www.redhat.com/en/topics/cloud-native-apps) that require rapid scaling, like real-time data streaming through[Apache Kafka](https://www.redhat.com/en/topics/integration/what-is-apache-kafka).

Kubernetes was originally developed and designed by engineers at Google. Google was one of the early contributors to Linux container technology and has talked publicly about how [everything at Google runs in containers](https://speakerdeck.com/jbeda/containers-at-scale). (This is the technology behind Google’s [cloud services](https://www.redhat.com/en/topics/cloud-computing/what-are-cloud-services).)

Google generates more than 2 billion container deployments a week, all powered by its internal platform, [Borg](http://blog.kubernetes.io/2015/04/borg-predecessor-to-kubernetes.html). Borg was the predecessor to Kubernetes, and the lessons learned from developing Borg over the years became the primary influence behind much of Kubernetes technology.

**Benefits of Using Kubernetes:**

The primary advantage of using Kubernetes in your environment, especially if you are [optimizing app dev for the cloud](https://www.redhat.com/en/topics/cloud-native-apps), is that it gives you the platform to schedule and run containers on clusters of physical or [virtual machines](https://www.redhat.com/en/topics/virtualization/what-is-a-virtual-machine?extIdCarryOver=true&intcmp=701f20000012m2KAAQ&sc_cid=701f2000000RmAOAA0) (VMs).

More broadly, it helps you fully implement and rely on a container-based infrastructure in production environments. And because Kubernetes is all about automation of operational tasks, you can do many of the same things other application platforms or management systems let you do—but for your containers.

Developers can also create cloud-native apps with Kubernetes as a runtime platform by using [Kubernetes patterns](https://www.redhat.com/en/topics/cloud-native-apps/introduction-to-kubernetes-patterns). Patterns are the tools a Kubernetes developer needs to build container-based applications and services.

With Kubernetes you can:

* Orchestrate containers across multiple hosts.
* Make better use of hardware to maximize resources needed to run your enterprise apps.
* Control and automate application deployments and updates.
* Mount and add storage to run stateful apps.
* Scale containerized applications and their resources on the fly.
* [Declaratively manage services](https://www.redhat.com/en/topics/containers/what-is-kubernetes-deployment), which guarantees the deployed applications are always running the way you intended them to run.
* Health-check and self-heal your apps with autoplacement, autorestart, autoreplication, and autoscaling.

However, Kubernetes relies on other projects to fully provide these orchestrated services. With the addition of other open source projects, you can fully realize the power of Kubernetes. These necessary pieces include (among others):

* Registry, through projects like Docker Registry.
* Networking, through projects like OpenvSwitch and intelligent edge routing.
* Telemetry, through projects such as Kibana, Hawkular, and Elastic.
* Security, through projects like LDAP, SELinux, RBAC, and OAUTH with multitenancy layers.
* Automation, with the addition of Ansible playbooks for installation and cluster life cycle management.
* Services, through a rich catalogue of popular app patterns.

Content From,

[ [**https://www.redhat.com/en/topics/containers/what-is-kubernetes?exa47498546=&adobe\_mc\_sdid=SDID%3D4F167DCFE2959AF8-23EDCC0D9EE78FF4%7CMCORGID%3D945D02BE532957400A490D4C%40AdobeOrg%7CTS%3D1603111386**](https://www.redhat.com/en/topics/containers/what-is-kubernetes?exa47498546=&adobe_mc_sdid=SDID%3D4F167DCFE2959AF8-23EDCC0D9EE78FF4%7CMCORGID%3D945D02BE532957400A490D4C%40AdobeOrg%7CTS%3D1603111386) ]

**Docker:**

**Introduction:**

[Docker](https://github.com/docker/docker) is a tool designed to make it easier to create, deploy, and run applications by using containers. Containers allow a developer to package up an application with all of the parts it needs, such as libraries and other dependencies, and deploy it as one package. By doing so, thanks to the container, the developer can rest assured that the application will run on any other Linux machine regardless of any customized settings that machine might have that could differ from the machine used for writing and testing the code.

In a way, Docker is a bit like a virtual machine. But unlike a virtual machine, rather than creating a whole virtual operating system, Docker allows applications to use the same Linux kernel as the system that they're running on and only requires applications be shipped with things not already running on the host computer. This gives a significant performance boost and reduces the size of the application.

And importantly, Docker is [open source](https://opensource.com/resources/what-open-source). This means that anyone can contribute to Docker and extend it to meet their own needs if they need additional features that aren't available out of the box.

**Benefits of Using Docker:**

Docker is a tool that is designed to benefit both developers and system administrators, making it a part of many DevOps (developers + operations) toolchains. For developers, it means that they can focus on writing code without worrying about the system that it will ultimately be running on. It also allows them to get a head start by using one of thousands of programs already designed to run in a Docker container as a part of their application. For operations staff, Docker gives flexibility and potentially reduces the number of systems needed because of its small footprint and lower overhead.

**Docker and Security**

Docker brings security to applications running in a shared environment, but containers by themselves are not an alternative to taking proper security measures.

Dan Walsh, a computer security leader best known for his work on SELinux, gives his perspective on the importance of making sure [Docker containers are secure](https://opensource.com/business/14/7/docker-security-selinux). He also provides a detailed breakdown of [security features](https://opensource.com/business/14/9/security-for-docker) currently within Docker, and how they function.

**Understanding Containers**

Containers can be thought of as necessitating three categories of software:

* Builder: technology used to build a container.
* Engine: technology used to run a container.
* Orchestration: technology used to manage many containers.

One of the appeals of using containers is their ability to die gracefully and respawn upon demand. Whether a container’s demise is caused by a crash or because it’s simply no longer needed when server traffic is low, containers are *cheap* to start, and they’re designed to seamlessly appear and disappear. Because containers are meant to be ephemeral and to spawn new instances as often as required, it’s expected that monitoring and managing them is not done by a human in real-time, but is instead automated.

Linux containers have facilitated a massive shift in high-availability computing, and there are many toolsets out there to help you run services (or even your entire [operating system](https://silverblue.fedoraproject.org/)) in containers. Docker is one option among many, as defined by [Open Container Initiative (OCI)](https://www.opencontainers.org/), an industry standards organization meant to encourage innovation whilst avoiding the danger of vendor lock-in. Thanks to the OCI, you have a choice when choosing a container toolchain, including Docker, [OKD](https://www.okd.io/), [Podman](http://podman.io/), [rkt](https://coreos.com/rkt/), [OpenShift](http://openshift.io/), and others.

If you decide to run services in containers, then you probably need software designed to host and manage those containers. This is broadly known as container orchestration. The [Kubernetes](https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/) provides container orchestration for a variety of container runtimes.

Content From,

[ **https://opensource.com/resources/what-docker** ]