**A provisional exploration of Kubernetes**



# CSC7071: Cloud Computing

Led by Esha Barlaskar

Report conducted by Lee Service - 40312791

[**CSC7071: Cloud Computing**](#_yfjy6o5zqot8) **0**

[An overview and history of Kubernetes:](#_enefvnsaj479) 1

[Understanding & exposition of Kubernetes within the current situation:](#_xw9vshzcszf3) 3

[Risks & Opportunities associated with Kubernetes:](#_yqjtqagpwpiu) 7

[The Future of Kubernetes:](#_y5whuhfjoz3k) 9

[Appendix:](#_b1z94nuucu34) 11

### An overview and history of Kubernetes:

Before we dive into the functions and history of Kubernetes (also known as K8s or ‘Kube’), the most important thing to know about it is that it allows organisations to grow by addressing their infrastructural or operational task needs. Kubernetes’ scalability and workload diversity when dealing with things like microservice applications allows users to manage hundreds or thousands of containers at production scale, which ultimately translates to businesses being able to be more flexible in how they build and deploy applications. Kubernetes helps organisations provide optimal service continuity through the systems handling of fault tolerance and load balancing, and when combined with the Docker container system, Kubernetes makes deploying, managing and monitoring microservices on a cloud system like AWS much safer and more stable than using in-house / manual processes.

Way back in 2004, Google had wanted to run their applications with some form of resource limiter put on them when running them in isolation, and had created an early form of containers to do so. In 2013, Google released the project ‘Let Me Contain That For You’ (LMCTFY), an open-source version of Google’s container stack which provided Linux application containers to the public, revealing that they had been using LMCTFY for all of Google’s resource isolation needs since 2007[[1]](#footnote-0). LMCTFY was able to integrate with something called Docker, an open source containerisation platform which would soon set the standard/best practices for containerisation within the industry.

With the introduction of Docker in 2013, the modern container era was born and enterprises like Netflix began to move away from traditional virtual machine (VM) hardware virtualisation in favour the more lightweight implementation of OS virtualisation which containers provide due to their ability to help developers build and test applications more easily in a production like environment[[2]](#footnote-1). Containers beat hypervisor-based VM virtualisation because containers don’t need to install an OS system which eats up resources on the server such as RAM, CPI and bandwidth. Furthermore, containers are fast to utilise and destroy - “container resource utilization is much more efficient. If a container is not executing anything, it is not using up resources….starting and stopping a container is more akin to starting and quitting an application, and is just as fast’[[3]](#footnote-2). With containers sharing the host machine OS, each individual container contains only the application, libraries and dependencies relevant for its purpose - making them a smaller and neatly packaged size which is fast to spin up as a portable alternative to VMs.

Google quickly began to run over 2 billion containers per week - that’s over 3000 started per second, and not counting long-running containers either. Google are using them internally for things like resource isolation, setting predictions for what resources are needed and accounting for available resources, as well as supporting existing cloud services[[4]](#footnote-3). With so many containers running at once, Google needed a tool to automatically manage large groups of containers to run hundreds of thousands of jobs from many different applications, across tens of thousands of machines, so they created Borg[[5]](#footnote-4), a system which Kubernetes themselves have cited as a direct influence on their platform[[6]](#footnote-5). Borg had several features which were incorporated into Kubernetes such as the ability to schedule containers to run together in one resource envelope, in Borg this was known as running an ‘alloc’ (short for resource allocation) and Kubernetes have implemented this feature as ‘Pods’. A pod is a group of one or more containers deployed to a single node (called a ‘Kubelet’) within Kubernetes - all containers in a pod share an IP address and port space among other resources and are controlled as a single application. Borg was able to provide services a name and map them to different allocs and balance the workloads of these via load balancing but Kubernetes goes one step further with this through automatically load-balancing connections to services among pods which match their label selector (name) and keeps track of where the pods are running as they get rescheduled over time due to any failures which might occur. In Borg, all tasks on a machine use the IP address of a host and share the hosts port space. This means Borg must schedule ports as a resource and tasks need to declare how many ports they need. Kubernetes addresses this pain point by giving every pod and service its own IP address, removing this kind of infrastructure complexity, resulting in services being able to find and communicate with each other, much more effectively.

Just as Google required Borg to manage their containers, other companies too found themselves requiring a service to automate many of the manual processes involved in deploying, managing, scaling and networking containerised applications. These processes were addressed via the means of ‘Container orchestration’. Container orchestration can be used in any environment where containers reside, it can be used to deploy the same application across different environments, and it makes it easier to orchestrate microservices within containers when these microservices can involve things like storage, networking or security applications. When we think about container orchestration, we think about how we can automate and manage tasks such as provisioning and deployment, scheduling and allocating resources to sustain our microservices, container availability, load balancing and traffic routing as well as monitoring container health and keeping interactions between containers secure[[7]](#footnote-6).

From the list above, we can appreciate container orchestration being an integral part of any containerised applications’ growth. This is where Kubernetes comes into play as a framework for managing containers and microservice architecture at scale. As a framework backed by the open-source community, Kubernetes orchestration allows you to “build application services that span multiple containers, schedule containers across a cluster, scale those containers and manage their health over time”[[8]](#footnote-7). Imagine being a developer who is asked to deploy 100 containerised applications across 10 different countries, across a hybrid cloud - can you imagine how long and difficult that might take to organise and develop for one person, a small team or even a small-medium sized company?

With Kubernetes, developers and DevOps teams can eliminate these manual processes involved by clustering large groups of hosts using a ‘Kubernetes cluster’ which is made up of ‘worker nodes’ - physical iron or virtual machines running Linux containers. These nodes are managed by a master controller which oversees resources in the cluster from one central platform. These clusters can have different system requirements and they can span hosts across public, private or hybrid clouds in different countries, which is ideal for hosting cloud-native apps which require access to many other microservices to work or that require rapid scaling to be efficient. Through Kubernetes built-in load balancing tools, user workloads will be distributed evenly and seamlessly among your applications - this cuts down on the development need to make applications as ‘skimmed down’ as possible to use less resources, Kubernetes just works with what you have, as best as it can.

### Understanding & exposition of Kubernetes within the current situation:

According to DataDog’s study on Docker Containers, “the typical organization that uses a container orchestrator runs 11.5 containers per host” on an ongoing basis and in organizations that adopt Docker, “deployments tend to increase steadily in scale long after the initial rollout”[[9]](#footnote-8) - in 2018, when the study was recorded, the average user deployed between 300-350 containers a year. Today it is commonplace for organisations to utilise hundreds or thousands of containers when producing applications, and this is especially the case if they want to support the creation of cloud-native applications to scale horizontally (add more machines to the resource pool, rather than adding more power via increasing RAM, CPU to existing machines)[[10]](#footnote-9). According to the annual State of the Cloud report conducted by Flexera for 2020, the adoption rate of Kubernetes has increased each year among enterprises since it’s inception and the adoption rate for Kubernetes currently sits at 58% whereas container adoption sits at 65% - signifying just how popular platforms like Docker and Kubernetes are in the US[[11]](#footnote-10).

What about container orchestration options outside of Kubernetes however? Other options do exist such as Docker Swarm and Apache Mesos but Kubernetes dominates the market - searches on Google for ‘Docker Swarm in 2021’ present articles like “Is Docker Swarm Dead? The Future of Docker Swarm”[[12]](#footnote-11) and Mirantis, the company who acquired the Docker Enterprise platform states that “The primary orchestrator [for Docker Enterprise] going forward is Kubernetes”[[13]](#footnote-12) and that they will be helping enterprise customers transition from their PaaS model to ‘K8s-as-a-service’. Docker Swarm mode is still alive and included in Docker’s community edition, but whilst this smoothly integrates with the Docker API and tools such as Docker Compose, and is easy to set up for Docker environments, it is less functionality-rich than Kubernetes[[14]](#footnote-13).

When choosing an orchestrator, a major selling point of Kubernetes is that it sets the industry best practices and is supported by every major cloud provider (IBM Cloud, AWS, Microsoft Azure, Google Cloud Platform, etc.). According to IBM “Kubernetes is more powerful , customizable, and flexible [than Docker Swarm], which comes at the cost of a steeper initial learning curve. Running Kubernetes through a managed service simplifies open-source management responsibilities, which allows you to focus on building your applications.”[[15]](#footnote-14) Mesos on the other hand is used by organisations such as Twitter, Apple and AirBnB due to its strength in being able to analyse analytics and conduct Big Data processing for processing continuous streams of data at scale or performing scientific computations like matrix, graph or network algorithms[[16]](#footnote-15). Kubernetes is generally easier, lighter and quicker to use over Mesos, offering a very high level of portability and from a business point of view, can be recommended for companies which are production ready and require any type of containerised environment - big or small.[[17]](#footnote-16) Whilst companies may still support Mesos, their twitter feed[[18]](#footnote-17) contains few updates since 2019 and Stack Overflow is disheartening for anyone who might want to utilise Mesos as there seems to be a dead community surrounding it, questions are asked rarely (almost 1-2 per month) and they rarely have multiple answers or over 100 views per topic.[[19]](#footnote-18) In contrast, Kubernetes flagship conference Kubecon has attendance rates which have almost doubled in size each year since its inception in 2017[[20]](#footnote-19).

When considering ethical issues around Kubernetes, Anne Currie, a lecturer in tech ethics at the university of Hertfordshire, talks about how important it is to be aware of the ethics regarding data centres and DevOps. Containers allow for speed when deploying applications, but the haste in which deploy applications can prevent us from reflecting on how ethical our applications may be when being released into the wider world. If we do not understand each microservice we utilise, we could lose the ability to see the bigger picture and our feelings of responsibility for the whole product we are creating, ultimately warping out intentions for something good into a possibly unethical product and due to Kubernetes enabling scale, any unethical applications developed may have a bigger negative impact on the world than we can account for. When we build our own infrastructure from scratch, we can control who our content impacts which allows us to scale we can consider how ethical each component is- if we make a mistake which only affects 100 people, then we can recover, but if we make a mistake which affects 100 million people, then this is a much more serious issue. As developers using Kubernetes, we do not need to have this slow, methodological and reflective way of thinking - as we don’t need to consider the manual processes associated with infrastructure, we just scale when we think we need it.[[21]](#footnote-20)

Another concern ethically to think about when using platforms like Docker and Kubernetes is the increased power consumption required to run data centres across the world. In 2018, it was reported that data centres used an estimated 200 terawatt hours (TWh) each year, more than the national energy consumption of some countries, including Iran[[22]](#footnote-21). More enterprises adopting cloud computing practices and utilising the scalability of orchestration platforms like Kubernetes means more data centres and an increased chance that data centres will handle energy consumption of these centres poorly. In 2017, an IT expert in California found “16,000 servers tucked into corporate closets and basements and found that about one-quarter of them were “zombies”, sucking up power without doing any useful work — perhaps because someone simply forgot to turn them off”[[23]](#footnote-22) so in the words of Spidermans’ Uncle Ben - “with great power, there must also come great responsibility”[[24]](#footnote-23).

That isn’t to say we should not use Kubernetes or slow down, as Anne Curry mentions, we just need to be aware of ethics when using orchestration platforms like Kubernetes and understand what we’re committing to at every step of production. When it comes to ethical power consumption by data centres, large corporations like IBM, Microsoft and Amazon are doing their best to reduce their electricity bills to save themselves money but also to increase overall efficiency within their data centres via hyperscalers[[25]](#footnote-24) - scaled back servers which are extremely efficient which old data can be ported onto to cut overall data centre power consumption[[26]](#footnote-25). Furthermore, in most cases within the west, large ICT companies are committing to renewable green energy - Google is the largest corporate purchaser of green energy on the planet and Facebook and Apple have committed to run 100% on renewable energy[[27]](#footnote-26). Ultimately, adopting a common sense approach such as only scaling to the point you or your organisation needs to when using Kubernetes can help mitigate our ethical concerns around the platform.

### Risks & Opportunities associated with Kubernetes:

As mentioned above, Kubecon is growing only more popular each year and with that, the ecosystem around Kubernetes grows as more companies adopt the technology. 91% of survey respondents for the Cloud Native Computing Foundation Survey 2020 reported that they were using Kubernetes and 83% of these users reported that they were using it in production. This continues a steady increase from 78% in 2019 and 58% in 2018, indicating that it’s only becoming more popular as time goes on.[[28]](#footnote-27) With growth, comes an increased need to focus on platform security, and this can be a big concern with enterprises who may want to use Kubernetes within their operations.

In June 2021, RedHat published an article on ‘The State of Kubernetes Security’ which states that of over 500 DevOps professionals interviewed, “94%...stated they have experienced a security incident in their Kubernetes and container environments during the last 12 months. And more than half of respondents (55%) have needed to delay deploying Kubernetes applications into production due to security.”[[29]](#footnote-28) Redhat points to human error thanks to misconfigured settings as the source of most data breaches and hacks. Misconfigured settings are more likely to occur as teams begin to run Kubernetes when the learning curve is at its steepest - “With k8s in particular, organizations are still learning how to run k8s, which makes it easy to make mistakes that can increase risk.”[[30]](#footnote-29) It's when teams overlook configurations and don’t realise that communications start happening between their cloud provider, Kubernetes orchestration layer and containerised workloads that these configurations become attack vectors for hackers.

Another element of security risk is associated with leaving Kubernetes features unguarded such as the Kubelet and the Kubernetes API server. If unprotected (as a result of misconfiguration or otherwise) these can also become security vulnerabilities[[31]](#footnote-30). Additionally, leaving sensitive ports exposed creates unnecessary risks. The CNCF recommends configuring your network to block access to the ports which the kubelet uses, as a [best practice](https://www.cncf.io/blog/2019/01/14/9-kubernetes-security-best-practices-everyone-must-follow/); CNCF also recommends restricting access to the Kubernetes API server to only trusted networks, otherwise you leave yourself at the risk of being exposed.[[32]](#footnote-31) Please refer to the appendix for 29 best practices to maintain security using Kubernetes.

Finally, developers should really consider a layered approach to container security to reduce security risks. By baking security into container infrastructure this reduces the risk of the entire environment being compromised by one weak link in the chain as each layer of a containers’ deployment is thoroughly secure.[[33]](#footnote-32)

Although we have talked about the risks and a number of advantages Kubernetes has as a container orchestration platform throughout this report, the opportunities which come from utilising the platform is fantastic if it can be maintained. It’s open source, allowing developers to take advantage of on-premises, hybrid or public cloud infrastructure so developers can effortlessly move workloads where it matters, without being locked into one provider like AWS or Azure by providing capabilities for containers without restrictions, achieved through its Pods and Services.

As mentioned above, Kubernetes pods share resources and an IP address, this removes the temptation to cram too much functionality into a single container image, improving modularity of applications[[34]](#footnote-33). Services group together a collection of pods and can be configured for things like horizontal scaling and load-balancing. Kubernetes can scale without increasing your ops team by deploying and updating software at scale through simplifying version control (where pods are updated based on newer application images being added and being able to roll back to earlier deployment if something is unstable). Deployments can be scaled out at any time across pods and paused. Pods can also be autoscaled if there is an increased need for resources within the users defined limits. Kubernetes can self-heal, killing containers that don’t respond to health checks or replacing containers which fail[[35]](#footnote-34).

Finally, Kubernetes helps us better use our containers when we want to create ‘cloud-native’ apps, which are defined by the Cloud-native foundation as software which “empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds”. These techniques produce software which is loosely coupled, “resilient, manageable and observable...they allow engineers to make high-impact changes frequently and predictably with minimal toil”[[36]](#footnote-35). Kubernetes has been a breakthrough for DevOps teams because before the platform came along, teams were often forced to script their own software deployment, scaling and manually update workflows[[37]](#footnote-36). Now we can leverage the platform to get the most out of our containers and build cloud-native applications that can run independent of cloud specific requirements.

### The Future of Kubernetes:

With the release of its 1.19 release in August of 2020, 382 companies and over 2,464 individuals contributed to Kubernetes, this highlights the continual growth of the ecosystem around the platform[[38]](#footnote-37). From examining Kubernetes’ case-studies page, companies like Adidas, IBM, Spotify and The New York Times continue to log their success with the platform from as recently as 2020[[39]](#footnote-38) but as more businesses come to use the platform, the initial learning curve of Kubernetes may put companies who want to be cloud-native off. Bruno Andrade, CEO of Shipa - a Kubernetes development platform - believes that to keep new customers, Kubernetes needs to become more automated to increase accessibility. Developers shouldn’t need to know about the Kubernetes API or clusters, Kubernetes should be more like a user-friendly hypervisor where all developers need to do is focus on their applications and that’s it. ‘Managed Kubernetes’ is already offered by cloud providers such as AWS[[40]](#footnote-39) and Google[[41]](#footnote-40) because they understand its complexity, so bundling this preinstalled with their cloud services is something many companies may wish to look into because it saves a lot of time when it comes to infrastructure management. The cons of this however is that cloud providers want to lock in users and if portability is essential then they’ll likely need to keep an eye on dependencies between vendors so they are not isolated to one service provider's clouds[[42]](#footnote-41).

As the ‘Internet of Things’ continues to expand, we need increased connectivity across all of our devices, whether it’s a fridge or a phone. Kubernetes integrates into 5G internet, and big agencies like telecom or household appliance companies have been developing applications using Kubernetes managed containers[[43]](#footnote-42). Connections will only become more powerful via edge computing thanks to the development of 5G[[44]](#footnote-43).

Red Hat CEO Paul Cormier has said, if [edge computing is going to be a realistic future](https://www.redhat.com/en/blog/edge-open-why-scale-out-computing-doesnt-exist-without-open-hybrid-cloud) for enterprise IT, it needs the hybrid cloud AND open source to thrive. To truly enable hybrid cloud, enterprise Kubernetes must come to the network’s edge and customers must be able to manage edge sites as seamlessly as they can with apps deployed on private or public clouds.

### 

### Appendix:

Further reading:

<https://www.redhat.com/en/engage/container-security-20170802?intcmp=701f2000000tjyaAAA> - A layered approach to container and Kubernetes security Whitepaper by Redhat - signup is needed.

<https://www.stackrox.com/post/2020/05/kubernetes-security-101/> - Kubernets security 101: Risks and 29 best practices - Stack Rox, 2020

<https://www.sdxcentral.com/articles/news/kubernetes-opportunities-challenges-escalated-in-2019/2019/12/>

<https://dzone.com/articles/the-state-of-serverless-computing-2021>

<https://www.ibm.com/downloads/cas/ZJLWQOAQ>

~~Overview of the topic including background - 15% - background to the topic in question. For example, what’s the history or motivation behind it, how does it relate to earlier technologies? High quality overview with detailed background showing extensive understanding of the topic. Background is relevant and all key issues are well explored.~~

~~Understanding and exposition of the current situation - 15% - is this widely used? To what extent? Are there any current legal or ethical issues around it?~~

~~Risks and Opportunities - 15% – are there any significant risks or opportunities applicable to the topic? Is it seen as a new hope for example or a disruptive technology? This should explicitly include security considerations if applicable for the topic (and security is applicable to almost everything).~~

~~Future - 10% – what does the future seem to hold for the topic?~~

~~An excellent identification and analysis of future direction with all points covered in very good detail.~~

~~Overall Quality - 15%~~

~~Excellent report very well written throughout with clear logical structure suitable for the topic~~

~~Range, Suitability, and Use of References - 10%~~

~~Excellent range of appropriate references from different suitable sources. Referencing used and styled consistently throughout.~~

~~Clarity of Message and Content Quality of Video Presentation - 20%~~

~~Excellent presentation, well thought out and presented with all key messages covered.~~

1. <https://stackoverflow.com/questions/19196495/what-is-the-difference-between-lmctfy-and-lxc/19200008#19200008> - Rohit Jnagal, commenting on how LMCTFY was used within google and its benefits [↑](#footnote-ref-0)
2. <https://dl.acm.org/doi/fullHtml/10.1145/3155112.3158370> - andrew leung, andrew spkyer and tim bozarth - titus introducing containers to the netflix cloud [↑](#footnote-ref-1)
3. <https://www.seltzer.com/margo/teaching/CS508.19/papers/merkel14.pdf> - Dirk Merkel - Docker: Lightweight Linux Containers for Consistent Development and Deployment [↑](#footnote-ref-2)
4. <https://speakerdeck.com/jbeda/containers-at-scale?slide=2> - Joe Beda’s Google Presentation - Containers At Scale, 2014 [↑](#footnote-ref-3)
5. <https://research.google/pubs/pub43438/> - Large Scale cluster management at Google with Borg, 2015, [Abhishek Verma](https://research.google/people/AbhishekVerma/),Luis Pedrosa Madhukar R. Korupolu, David Oppenheimer, [Eric Tune](https://research.google/people/EricTune/), [John Wilkes](https://research.google/people/JohnWilkes/) [↑](#footnote-ref-4)
6. <https://kubernetes.io/blog/2015/04/borg-predecessor-to-kubernetes/> - Borg: The Predecessor to Kubernetes, 2015 [↑](#footnote-ref-5)
7. <https://www.redhat.com/en/topics/containers/what-is-container-orchestration> [↑](#footnote-ref-6)
8. <https://www.redhat.com/en/topics/containers/what-is-container-orchestration> What is container orchestration?, 2019 [↑](#footnote-ref-7)
9. <https://www.datadoghq.com/docker-adoption/> 8 surprising facts about real docker adoption [↑](#footnote-ref-8)
10. <https://www.ibm.com/blogs/cloud-computing/2014/04/09/explain-vertical-horizontal-scaling-cloud/> - How to explain vertical and horizontal scaling in the cloud, IBM - David Beamont, 2014 [↑](#footnote-ref-9)
11. <https://www.forbes.com/sites/janakirammsv/2020/05/02/10-key-takeaways-from-rightscale-2020-state-of-the-cloud-report-from-flexera/> - Janakiram MSV, 2020 [↑](#footnote-ref-10)
12. <https://www.optimum-web.com/is-docker-swarm-dead-is-anyone-using-swarm-in-production-in-2021-future-docker-swarm/> - 2021 [↑](#footnote-ref-11)
13. <https://www.mirantis.com/blog/mirantis-acquires-docker-enterprise-platform-business/> - Adrian Ionel, CEO of Mirantis, 2019 [↑](#footnote-ref-12)
14. <https://thenewstack.io/kubernetes-vs-docker-swarm-whats-the-difference/> - Kubernetes vs. Docker Swarm: What’s the Difference? - 2018 [↑](#footnote-ref-13)
15. <https://www.ibm.com/cloud/blog/docker-swarm-vs-kubernetes-a-comparison> - Docker Swarm vs Kubernetes: A Comparison, Chris Rosen, 2021 [↑](#footnote-ref-14)
16. <https://opensource.com/business/14/9/open-source-datacenter-computing-apache-mesos> - Open source datacenter computing with Apache Mesos [↑](#footnote-ref-15)
17. <https://vexxhost.com/blog/kubernetes-mesos-comparison-containerization> - Kubernetes vs Mesos - Hind Naser, 2017 [↑](#footnote-ref-16)
18. <https://twitter.com/apachemesos?lang=en> - Apache Mesos [↑](#footnote-ref-17)
19. <https://stackoverflow.com/questions/tagged/mesos?tab=newest&page=7&pagesize=1> - apache mesos questions tagged mesos [↑](#footnote-ref-18)
20. <https://events.linuxfoundation.org/wp-content/uploads/2020/09/KubeCon_EU_20_Report_final.pdf> - KubeCon + Cloud Native Con Conference transparency report 2020 - a report showing how each year since 2017, Kubecon registration has increased significantly. [↑](#footnote-ref-19)
21. <https://www.oreilly.com/radar/kubernetes-good-or-evil-the-ethics-of-data-centers/> - Kubernetes: Good or Evil? - The ethics of data centers - Ann Currie, 2018 [↑](#footnote-ref-20)
22. How to stop data centres from gobbling up the world's electricity

    * [Jones, Nicola](https://ui.adsabs.harvard.edu/#search/q=author:%22Jones%2C+Nicola%22&sort=date%20desc,%20bibcode%20desc)

    Abstract

    The energy-efficiency drive at the information factories that serve us Facebook, Google and Bitcoin.

    Publication:

    Nature, Volume 561, Issue 7722, p.163-166

    Pub Date:

    September 2018

    DOI:

    [10.1038/d41586-018-06610-y](https://ui.adsabs.harvard.edu/link_gateway/2018Natur.561..163J/doi:10.1038/d41586-018-06610-y)

    Bibcode:

    [2018Natur.561..163J](https://ui.adsabs.harvard.edu/#abs/2018Natur.561..163J/abstract)

    <https://media.nature.com/original/magazine-assets/d41586-018-06610-y/d41586-018-06610-y.pdf> [↑](#footnote-ref-21)
23. How to stop data centres from gobbling up the world's electricity [↑](#footnote-ref-22)
24. <https://en.wikipedia.org/wiki/With_great_power_comes_great_responsibility> [↑](#footnote-ref-23)
25. <https://en.wikipedia.org/wiki/Hyperscale_computing> [↑](#footnote-ref-24)
26. <https://www.wired.com/story/amazon-google-microsoft-green-clouds-and-hyperscale-data-centers/> - Amazon, Google, Microsoft: Here's Who Has the Greenest Cloud - 2019 - Daniel Oberhaus [↑](#footnote-ref-25)
27. How to stop data centres from gobbling up the world's electricity [↑](#footnote-ref-26)
28. <https://www.cncf.io/wp-content/uploads/2020/11/CNCF_Survey_Report_2020.pdf> - CNCF Cloud Native Survey 2020 Results, 2020 [↑](#footnote-ref-27)
29. <https://www.redhat.com/en/blog/state-kubernetes-security> - The state of Kubernetes security [↑](#footnote-ref-28)
30. <https://enterprisersproject.com/article/2020/5/kubernetes-security-new-tech-familiar-attack-vectors?page=0%2C0> - Kubernetes security: New tech, familiar attack vectors [↑](#footnote-ref-29)
31. <https://enterprisersproject.com/article/2020/5/kubernetes-security-new-tech-familiar-attack-vectors?page=0%2C1> [↑](#footnote-ref-30)
32. <https://enterprisersproject.com/article/2020/5/kubernetes-security-new-tech-familiar-attack-vectors?page=0%2C1> [↑](#footnote-ref-31)
33. <https://enterprisersproject.com/article/2020/5/kubernetes-security-new-tech-familiar-attack-vectors?page=0%2C1> [↑](#footnote-ref-32)
34. <https://www.infoworld.com/article/3173266/4-reasons-you-should-use-kubernetes.html> - 4 reasons you should use Kubernetes, Sirish Raghuram, 2017 [↑](#footnote-ref-33)
35. <https://kubernetes.io/> - Kubernetes home page [↑](#footnote-ref-34)
36. <https://github.com/cncf/toc/blob/main/DEFINITION.md> - The definition of cloud native by the Cloudf native computing foundation, 2018 [↑](#footnote-ref-35)
37. <https://www.infoworld.com/article/3173266/4-reasons-you-should-use-kubernetes.html> [↑](#footnote-ref-36)
38. [↑](#footnote-ref-37)
39. <https://kubernetes.io/case-studies/> [↑](#footnote-ref-38)
40. <https://aws.amazon.com/eks/> - Amazon Elastic Kubernetes Service [↑](#footnote-ref-39)
41. <https://cloud.google.com/kubernetes-engine> - Google Kubernetes Engine [↑](#footnote-ref-40)
42. <https://puppet.com/blog/the-future-of-kubernetes/> - The Future of Kubernetes, Demetrius Malborough, 2021 [↑](#footnote-ref-41)
43. <https://kubernetes.io/case-studies/nokia/> - Nokia - enabling 5g and devops at a telecom company [↑](#footnote-ref-42)
44. <https://www.cncf.io/blog/2020/06/01/5g-rollout-how-kubernetes-and-edge-computing-is-making-5g-a-reality/> - 5G Rollout: How Kubernetes and Edge Computing is making 5G a reality, 2020Chad Serino [↑](#footnote-ref-43)