# Unix in the internet - Communications Report

623190x - Stefan Katris

## I. Introduction

Unix and Unix-based operating systems are at the heart of the modern cloud computing landscape, offering a versatile foundation that benefits both cloud providers and end-users. With their modularity and open-source nature, these operating systems present opportunity for optimization on multiple fronts, ensuring the delivery of consistent and highly available services. Beyond the advantages of customization and flexibility, Unix-based systems often adopt GPL, MIT, or BSD licensing, which eliminates licensing costs to use the operating systems within the enterprise space [1]. This cost-efficiency empowers companies to scale their infrastructure without incurring prohibitive licensing expenses, particularly for those embracing automation and infrastructure as code.

The open-source aspect of Unix and Linux allows for fine-tuned adjustments to the operating system itself. These optimizations play a pivotal role in elevating the performance of applications hosted in the cloud, addressing the ever-increasing demand for reliability and uptime. The simplicity and reliability of Unix and Linux allows for their capacity to operate seamlessly without disruptions, an important characteristic for organizations delivering uninterrupted online services.

## II. Network Services

Unlike proprietary operating systems Unix and Unix-based operating systems are mostly free to use for personal and commercial use, this allows for a more cost-effective option when trying to run a server or services on the cloud [1]. This allows for users of a cloud environment to be able to spin up and down servers and services without having to jump through the hoops of calculating the costs of licensing. For example, licencing for windows server [2] is especially complicated as it requires a fee per core that the server is running on and client access licenses (CALs) for each client that is accessing a service that is running on a windows server, exponentially increasing the cost for running services as you add more users. As a majority of Unix and Unix-based operating systems don't require any type of licensing to use the software it also means that there is no expectation of support for the operating system. This means that if there is an issue encountered or unexpected behaviour companies and users are mostly left to their own devices to troubleshoot and figure out the cause of said issue. For businesses that do require at least some level of support there are distributions of Unix and Linux that you are able to pay for a support contract. Companies such as Canonical, Suse and Red Hat operate solely on providing support for Linux and Unix operating systems but you are often locked into their distribution.

Since Linux/Unix has proven its self as reliable and has little to no licensing cost this is a clear choice for operating system for anyone providing any type of network service. A good example of Unix reliability and performance is that it's trusted by Netflix to serve up 15% [3] of the world's global internet traffic from their own content delivery network servers. Open Connect [4] is a program run by Netflix in which they have implemented their own CDN by providing servers to ISPs to install within their own networks. This allows ISPs to reduce their costs due to less reliance on peering to retrieve shows that might be streamed by their users and to provide a better experience to their end users by enabling low latency access to the Netflix library. Speeds of up to 800gbit/s are able to be achieved from a single server through optimisation to the network stack as well as offloading some crypto functions to the NIC rather than using the kernel or user space. The modularity and simplicity of Unix in this case allows for this performance to be achieved whereas trying to tune a Windows or even Linux sever to reach these kinds of performance would be impossible.

Both Apache and Nginx, the two most popular web hosts are designed to run on *nix operating systems and as of 2022 both serve a combined 65% [5] of traffic from the worlds busiest websites. Apache is one of the oldest web servers still used today and is most commonly used along side web managers such as WordPress, as it's a modular and open platform for web hosting it can be used to create complex sites through regular html, css, php and javascript code or through managers such as WordPress for those who may not be as technically inclined. Nginx has further functionality for web services that Apache lacks [6], features such as reverse proxy and load balancing are some of its standout features but its performance while serving many requests makes it attractive for serving web content that is being accessed many thousands of times a second.

## III. Containers

From a cloud provider's perspective, transitioning customers from Infrastructure as a Service (IaaS) to Platform/Container as a Service (P/CaaS) offers several benefits. IaaS resembles purchasing a server, with varying levels of resources available for lease. Whereas containers, managed by tools like Docker and Kubernetes, are highly scalable and well-suited for cloud environments with variable workloads. [7] Containers can be easily scaled up or out based on demand, making them ideal for cloud environments where workloads can be highly variable. Tools such as docker and Kubernetes can assist both cloud providers and their customers to manage their containers. The orchestration aspect of Kubernetes allows for quick creation of containers with the specified images and resources needed. This allows for containers managed by Kubernetes to scale as load increases allowing for a more

dynamic use of resources. Compared to IaaS where you are only able to access the resources that you are paying for you are locked into a more rigid position and if you are to scale, it is more of a manual process. As containers are deployed on top of existing Unix based OS installs, the start time for a container can be done within seconds as the kernel and drivers are shared from the host machine that is running the containers. This means that containers are significantly smaller than operating system images which allow for fast initialisation time as well as being able to react quickly when demand is increasing to deploy more containers to help with the increased load.

Instances of IaaS and Containers can share physical memory and other hardware both are isolated from other workloads that may be running, even containers which share kernel information are securely isolated from other containers using linux tools such as namespaces. [**?**] Namespaces allows for resource and process separation on a single system, this then allows for containers to only view certain aspects of the operating system that they have been assigned. This includes Process ID separation, network isolation so each container can have its own separate network stack, files can be isolated using Mount namespaces and user and group namespaces ensure that users that are created within a container are only applicable to that container.

A large advantage for a cloud provider through CaaS is the ability to run many containers on a single host, a few IaaS will commonly be run on a single host but you are limited mainly by CPU and Memory resources on that host. This is due to the fact that when a customer is utilising IaaS, they are entitled to the entirety of the compute resource that they are renting. For example, if you had a host with 16 cores the most instances of IaaS you can provide without over subscribing and possibly degrading your client's performance is 16 where each VM gets a single core to run compute on. Containers on the other hand are able to be configured similarly to VM's with certain amounts of CPU or memory available but those values can dynamically change on the fly allowing for dynamic allocation of resources based on the loads being experienced on certain containers in relation to the other containers on the same system. [8]

## IV. Tools

Docker and Kubernetes are two of the most commonly used tools in the containerized application ecosystem [7], each with their own features that complement the other. Docker serves as a containerization platform that allows for the packaging and distribution of applications within a container. Developers use Docker to encapsulate applications and their dependencies into self-contained units known as Docker containers. These containers are constructed from Dockerfiles, which define the application's environment, code, and configurations. Images are then created from these Dockerfiles which then provide a consistent and portable representation of the application allowing it to then be run in most environments.

On the other hand, Kubernetes functions as an orchestration platform designed to manage the deployment and operation of containers within a cluster of machines. Kubernetes introduces the concept of a "pod,". Each pod can contain one or more Docker containers that share the same network namespace and storage volumes. Kubernetes abstracts away the complexities of container management, allowing developers and operations teams to focus on defining the desired state of their applications. The desired state is typically written in YAML files, where developers specify the number of replicas, resource requirements, configurations, and other attributes of their applications. Kubernetes can then monitor the cluster's state and employ automation to scale pods up or down to maintain resources based on load. Kubernetes further simplifies the operational aspects of containerized applications by providing a suite of essential services. It handles networking ensuring that each pod has its own unique IP address and that traffic is correctly routed to the appropriate pods. Load balancing is built into Kubernetes, allowing for distribution of traffic among pod replicas. Resource management is another key feature, enabling resource quotas and limits to be set, preventing resource contention and ensuring efficient utilization.

## V. Conclusion

Unix and Unix-based operating systems are the go-to operating systems in the landscape of modern cloud computing. Their inherent modularity, open-source nature, and potential for optimization contribute significantly to the provision of reliable and highly available cloud services. Their adoption of GPL, MIT, or BSD licensing models removes the financial barrier, making them an economical choice for enterprises aiming to scale their infrastructure without incurring large licensing expenses. Netflix's integration of FreeBSD demonstrates why Unix and Unix like systems have a track record of reliability and performance which makes it a trusted choice for serving a significant portion of global internet traffic. Its modularity and simplicity allow for high-performance use cases that would be challenging to achieve with other operating systems. Additionally, Unix-based systems play a pivotal role in web hosting, with web servers like Apache and Nginx designed to operate seamlessly on these platforms. From a cloud provider's perspective, the transition from Infrastructure as a Service (IaaS) to Platform/Container as a Service (P/CaaS) can increase the utilisation of hardware for a cloud provider increasing their profits as well as providing a service that is now one of the most popular ways to run applications within the cloud. Containers, managed by tools such as Docker and Kubernetes, offer scalability, agility, and dynamic resource allocation, providing a more efficient and responsive solution for fluctuating workloads.

In summary, Unix and Unix-based operating systems, are the operating system of choice for cloud providers and many of their customers due to its proven reliability, free or cheap licensing and customisability. These combined strengths enable organizations to deliver robust, cost-effective, and highly available services.

## REFERENCES


[1] Licensing summary. [Online]. Available: https://fossa.com/learn/developers-guide-open-source-software-licenses

[2] Windows server pricing. [Online]. Available: https://www.microsoft.com/en-au/windows-server/pricing

[3] D. Gallatin. Nab show streaming summit. [Online]. Available: https://nabstreamingsummit.com/wp-content/uploads/2022/05/2022-Streaming-Summit-Netflix.pdf

[4] (2021) Netflix open connect. [Online]. Available: https://openconnect.netflix.com/Open-Connect-Briefing-Paper.pdf

[5] (2023, Sep) Web server overview. [Online]. Available: https://w3techs.com/technologies/overview/web$_server$

[6] Nginx about page. [Online]. Available: https://nginx.org/en/

[7] S. Hardikar, P. Ahirwar, and S. Rajan, "Containerization: Cloud computing based inspiration technology for adoption through docker and kubernetes," *2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC)*, 2021.

[8] Difference between containers and vms. [Online]. Available: https://aws.amazon.com/compare/the-difference-between-containers-and-virtual-machines/