TNE30019 Research Report - Unix on the Cloud

Harrison Sheppard - 101613496

***Abstract*—Historically, Unix Operating Systems have dominated the on-premise server space. And with the rise of Cloud platforms in recent years, this dominance has continued. This report explores the benefits of Unix Operating Systems in the Cloud, and why it is the OS of choice for Cloud Engineers and Architects.**

I. The Competition

In the server there is only one relevant competitor to Unix-like OSs. And this is Windows Server [1]. So this report will only consider how Windows and Unix compare, in a Cloud environment.

A. Open Source

Unix operating systems, and their core components, are open source and free software. The source code for the BSD or Linux kernels, file systems, GNU tools, and other utilities can all be read, modified and used, free of charge. This philosophy has a number advantages, for using Unix in a Cloud environment.

1) Optimisation: For a Cloud service provider, their data centres are very unique computing environments. Large numbers of servers, varying in hardware, with virtual machines and software defined network links constantly being created and deleted. In such an environment, the Cloud provider is incentivised to maximize the performance and efficiency of their systems. Thanks to the Unix platform being open-source, the Cloud provider can optimise device drivers for their hardware (or virtual hardware), alter the network stack to reduce overhead, or make any other changes they need to the core of the OS, to better align the behaviour of Unix to the needs of the Cloud provider [2]. This cannot be done in Windows. In the Windows kernel, device drivers run in user mode. While this has a small performance penalty, that penalty makes a big difference at large scales, like in Cloud environments. And the Windows networking stack is fully closed source, so cannot be tweaked and optimised for the Cloud providers needs.

2) Security: Open-source code can be read by anyone in the world, for free. This means that security researchers, bug-bounty hunters, and white-hat hackers, can asses the code and look for vulnerabilities. Especially for large projects, like the Linux kernel, every line of code that submitted gets checked by hundreds of eyes, all looking for potential security flaws. This open, crowd-sourced nature means that there is a much higher chance of finding, and fixing, any vulnerabilities before they make it to production, and create entry points for malicious actors [3].

Proprietary code can only be read by the developers working on it, and possibly an internal security team. No matter how skilled and competent the developers and security team are, they cannot beat the sheer quantity of public reviewers.

Of course, this doesn't mean that all open-source code is perfectly secure. Vulnerabilities do still make it in to Unix OSs and their utilities [4]. But thanks again to the code being open-source, anyone can edit it, and quickly patch security issues as they arise. Security patches can be created independent of the software maintainers, and quickly applied and installed, without needing to wait for anyone else to release a patched version.

For vulnerabilities in proprietary software the user is dependant on the maintainer to fix the issue and release a new version. And depending on the priorities of the developer or company, a fix may take a long time to release, or it could never be fixed at all.

All these security benefits are even more important in the Cloud. For Cloud service users, it's always important to maintain the security of their systems, and protect themselves from attackers. This is even more true for the Cloud service provider. Cloud data centres are shared environments, so any vulnerability that could allow an attacker in to their backend system, could result in data theft or unavailability for hundreds of clients.

B. "Everything is a file"

A key aspect of Unix design philosophy is that "Everything is a file". Storage, running processes, networking, HID inputs. All these things are represented as files within the Unix file system [5]. An importantly, system configuration is done via plain text config files. Windows however, uses an inconsistent mix of APIs and system calls (such as the Windows Registry), or file based systems, depending on the service being configured.

In Unix, storage devices are mounted at boot according to the config in `/etc/fstab`. While Windows users will usually use the GUI utility "Disk Management". If using `rc` on Unix, startup programs can be set in the file `/etc/rc.conf`, or with symbolic links if using `systemd`. In Windows, one may need to use both the GUI utility "Service Manager", and/or create shortcut files in the Windows startup folder. Which one is used depends on the software that is being set to run on boot. A network configuration, such as a static IP address, can also be set in `/etc/rc.conf` under Unix (or other locations, depending on the OS/distro). But Windows network config must be done via GUI settings, or command line tools.

The file-based methodology of Unix has advantages that are particularly apparent in large scale deployments, such as in the Cloud. Plain text config files can be easily backed up, kept in

version control with tools like git, and deployed across a large number of machines. Configuration drift, where the expected config and the actual config on a server are different, is very easy to detect with plain text configuration, as the files can simply be compared.

"Configuration as code" tools, like Ansible, Puppet, and Chef, have become very popular with Unix administrators. As they allow one to create declarative scripts that deploy configuration across many hosts. And the most common task that these tools will be doing is simply copying a file from the repository, to a Unix host, via SSH. This is not possible on Windows; its inconsistent, API and GUI-centric config makes setting configuration, and detecting drift, much more complex.

II. Virtual machines

Virtualisation is the use of one computer (the host), to simulate another computer (the guest), and run an operating system inside of another [6]. The use of virtualisation, or virtual machines (VMs), has a number of benefits for system administrators. Some, but far from all, of these benefits are:

- Resource allocation. A virtual machine can be allocated a limited amount of resources, like CPU cores and RAM. It can be allocated just enough to perform it's task, so that it doesn't impede other VMs or services on the host.
- Security isolation. VMs provide an additional layer of security. If an attacker compromises a VM, they will only have access to the VMs resources, and nothing more. This limits the scope of damage that an attacker can cause.
- Backups. A virtual machines virtual disk, which can include all it's configuration and data, can be easily backed up.
- Mobility. A virtual machine can be moved between different hosts. This way, if the host hardware fails, a backup of the VM can be restored and ran on a different host, reducing downtime.

However, virtualisation does have a number of drawbacks, such as:

- Simulation overhead. The host must simulate virtual devices, such as disks and network interface cards, so that the guest can use them. This simulation consumes computing resources.
- Guest overhead. Each guest is an entire operating system, which will be running it's own CPU scheduler, managing memory, operating a file system, networking, and device drivers. All of these things take up resources, which takes resources away from the actual applications we want to run in the VM.
- Added complexity. Every virtual machine must still be administered as if it was a physical PC. Updates must be installed, and configuration kept up to date. This creates more potential points of failure, and more time and effort from the admin to maintain.

Virtual machines are ideal for the Cloud. Often described as "Infrastructure as a Service", an administrator can quickly and easily provision a virtual machine and network links, and get all of the above benefits. But they don't have to worry about the underlying hardware, as managing the hardware is the job of the Cloud service provider. The Cloud provider can create large data centres, full of powerful hardware, and rent out virtual machines to it's customers. VMs provisioned by different customers can share the same hardware, while not needing to worry about sharing private data between them.

Both Windows and Unix can be used as VM hosts, and as VM guests. And it is perfectly possible to run a Unix guest on a Windows host, and vice versa. In this way, VMs also provide flexibility, allowing admins to choose the right OS for the right task. But as discussed above, Unix has a number of advantages that make it more desirable than Windows. Thanks to VMs admin could run Windows in a VM, whenever they need to use Windows exclusive applications. And they can keep Windows as isolated as possible from the rest of the network.

A. Containers

But Unix, thanks to it's modular and open nature, has led to the rise of a new technology which helps alleviate the drawbacks to virtualisation listed above.

Containerisation systems, like Docker, are conceptually similar to virtual machines. But instead of encapsulating entire operating systems inside of a virtual computer, only the application code we want to run is encapsulated [7]. A container is essentially an isolated Unix user space, running on top of the same kernel. Each container has it's own directory structure, separate installed apps and configuration, and it can only access resources outside of the container, if explicitly allowed to by the host. While low-level tasks, like CPU scheduling, memory management, and device drivers, are handled by the host kernel. With containers, we don't need to duplicate these kernel tasks, and consume resources. We only need to only need to include our application code and dependencies, and run it.

This has great benefits in a Cloud context. As many Cloud providers charge based on CPU usage and run time, increasing efficiency will result in reduced cost in the Cloud. If using virtual machines, an admin may need to create several VMs to isolate the various services they need to run. But with containers, the admin can create a single VM, and run multiple containers on it. Saving on cost, but keeping the benefits of VMs.

Thanks to the modularity of Unix OSs, a container can be stripped of everything that the application doesn't need. All unneeded libraries and system utilities can be removed, reducing the size, start time, and complexity of the container. This also reduces the configuration needed by admins. While a Unix admin will need to configure many different systems in a virtual machine (networking, drive mounting, startup services, etc), they will need to do very little in a container.

There are a number of containerisation systems, like BSD Jails, but by far the most popular is Docker. Which itself is based on the Linux kernel feature "Linux Containers", or LXC. Windows has no native equivalent to this. In fact, Unix is so

dominant in this space, that Microsoft has created "Windows Subsystem for Linux" (WSL), which allows Windows users to run Unix applications, and even Docker containers, on top of Windows.

Cloud providers have also taken great interest in containers. Instead of provisioning entire virtual machines to run their apps, admins can use "Container as a Service" (CaaS) platforms to directly upload run their containers in the Cloud. The Cloud platform automatically handles the orchestration of the containers, like what hardware to them on, and how many resources they need. This further reduces the complexity and required computing resources for admins, saving on time and cost. Behind the scenes, the Cloud provider is likely running the containers in a virtual machine. But the admin doesn't have to worry about it, as it's all managed by the Cloud platform.

III. Unix, networking services, and the Cloud

A. Intergration with the Cloud

The Cloud provides a number of useful tools for managing VMs and containers. In order to build a reliable and secure system, we need to be able to deploy, safely access, scale, and monitor our Unix-based infrastructure.

1) Deployment: Unix dominates the server space. Cloud providers know this, so to provide a better experience for admins, Cloud providers offer pre-built Unix virtual machine images [8]. Instead of needing to download and mount an installation ISO in a cloud VM, and then go through the installation process, an admin can simply create a VM, select their preferred OS or distro, and click "Go". In short order, the admin will have a Unix VM, with known default settings, ready to be configured and turned in to something useful.

For CaaS/PaaS providers, they also provide container images for common and popular servers [9]. With containers, it's even easier. The admin can just select the service they want and run it. It will also have a set of known defaults, and be made available to the admin and/or business.

2) Secure access: When building systems on-premise, there is some built-in network security, as your servers are not accessible over the public internet by default. This way it can only be accessed and configured if you have physical access to the local network. However, in order to access and configure a VM in the cloud, you *must* connect to it over the public internet. While SSH is considered secure, security-in-depth is better. It's good practice to put your management network behind a VPN, when building your Cloud infrastructure [10]. Unix has a number of options for this. Most commonly are OpenVPN and Wireguard. OpenVPN is a comprehensive, mature, and reliable VPN system. Wireguard is newer, but is much simpler, has better performance, and is easier to set up. But at the cost of advanced configuration.

Cloud providers may also offer VPN's themselves as a service, eliminating the need to set up a VM entirely [11].

3) Scaling: For something like a high traffic website, we will need a load balancer in order to reliably scale our services [12]. A load balancer is a powerful server that sits in front of the servers that actually process and handle requests. It distributes and balances the requests between servers, so that no one server gets overwhelmed with more requests than it can handle, taking the website down. Unix has good options for this, such as HAProxy or Nginx. They can be installed and configured on a VM, or ran directly inside of containers. Cloud platforms may also offer their own load balancers as services [13].

However, if the traffic to our website varies a lot over the day, then it would be a waste of resources and money to have a lot of powerful web servers to handle our requests, when they only get used some of the time. This is something that the cloud is uniquely equipped to solve. Cloud providers can also give admins another tool, "auto-scaling groups" [14]. Using auto-scaling groups, we can configure the cloud to automatically provision as many web servers as needed to meet the demand. And when demand drops off, it will destroy the web servers we don't need, saving on cost.

4) Monitoring: Reliable as they are, even Unix servers are not just set and forget. Issues can occur, such as a disk fulling up with logfiles, hardware issues taking a host offline (yes, even in the cloud), or a server runs out of CPU and/or RAM, killing it's performance. While we can build reliable systems, it's impossible to guarantee 100% uptime. Problems can and do occur, for reasons outside of our control. And when they do occur, it's important the we know about it quickly, so that we can respond [15]. Open source tools like Icinga can monitor for server and service availability. Prometheus can gather metrics, and Grafana can put those metrics in to easily consumer graphs. Tools like these are essentially for monitoring the health and performance of our systems.

Cloud providers may also offer tools that do this, in particular for CaaS and IaaS [16]. As containers do not run on a VM (as far as the cloud admin can see), we can't ensure that the container is always running at full speed using traditional methods. So we'll need to rely on metrics provided by the cloud platform itself to asses the health and performance of our containers.

5) Implementing services: With all the above, we can build reliable, scalable, and secure systems. But there's no point to building those systems, if we're not actually running something useful on them. Services such as web servers, file servers, VPNs, and databases can all be used in the cloud, and Unix has a number of possible solutions that can be used. The networking services that Unix provides are widely considered to be high quality, efficient, reliable and secure tools.

6) Web servers: Apache and Nginx are two great options for running web servers on Unix. They are open-source, mature, secure (when configured correctly), fast, and reliable [17].

Windows has Microsoft Internet Information Services (IIS). The biggest advantage that IIS has is it's great support for .NET. However, it's closed source nature, and Windows exclusivity has meant that it has struggled to gain popularity [18].

7) File servers: For file servers, we'll consider both FTP and SFTP. For Unix "Very Secure FTP Daemon", or vsftpd,

is the standard FTP server. It is considered fast and secure, and reliable [19].

For Windows, IIS is capable of FTP. While it is not consider to be a bad or insecure FTP server, it still carries all of the baggage of IIS being proprietary and Windows exclusive [20].

SFTP is an extension of the SSH protocol. And OpenSSH provides Unix with an open-source, secure, and fully featured SSH and SFTP server [21].

Windows however, has no standard SSH server. As the OS is very GUI focused, remote command-line access has not been needed, and as a side effect, does not have a standard SFTP server. It is still possible to SSH into Windows however. Microsoft have ported OpenSSH (originally built for Unix systems) to Windows, and it can be used to get remote command line access to a Windows system [22].

8) Databases: The predominant database software available for Windows is Microsoft SQL Server. It is considered to be a mature, powerful, and secure relational database management system [23]. It is proprietary and requires license purchasing to use.

There are a wide variety of database servers available for Unix. MySQL/MariaDB and Postgres provide relational DBs on Unix, with security and features comparable to MS SQLServer. But the added benefit of being free and open-source [24]. Additionally, there are many newer and innovative database systems being develop, with Linux as their OS of choice. Unix also has access to document database systems like MongoDB, key-value databases like Redis, and multi-model systems like SurrealDB.

9) Mail servers: There are two components to email. Sending and receiving mail, and accessing your own mailbox. The former is handled by mail transfer agents (MTAs), and the latter for IMAP servers [25]. On Unix, Postfix or Exim are well established, mature and reliable MTAs. And Dovecot is considered the standard IMAP server Unix. Together, these two systems provide reliable and high performance email services.

Microsoft Exchange server is a proprietary, Windows exclusive email server. Exchange includes both MTA and IMAP functionality [26]. However, Microsoft extends Exchange beyond basic email. Exchange is integrated with their other services, like their proprietary email client Outlook; and their user directory system, Active Directory. And Active Directory itself can be tightly integrated with Windows. This combination of Active Directory, Exchange, Outlook, and Windows, allows Microsoft to provide a well-integrated user account and communications stack, for businesses. This integration allows Microsoft to add more features and functionality, the kind of which would not be possible without controlling the entire software stack. It also makes the new account creation process very seamless for the systems administrators. Due to this, Exchange is a very common and popular email system for medium to large enterprises.

IV. Conclusion

Unix is the current operating system of choice for Cloud virtual machines, and is essential for the containerisation of applications. There are a large number of high quality, reliable and secure tools that Unix provides, and it capable of meeting the needs for most users and system administrators. While the Windows platform does have some advantages, like Exchange server, it is more often inferior, or at equal to Unix, in terms of it's functionality and flexibility, for both Cloud and on-premise environments.

References

[1] Fortune Business Insights. "Server Operating System Market" fortunebusinessinsights.com. https://www.fortunebusinessinsights.com/server-operating-system-market-106601 (accessed Sept. 9 2023)

[2] J. Burt. "Microsoft has made Azure Linux generally available. Repeat, Azure Linux" theregister.com. https://www.theregister.com/2023/05/26/microsoft_azure_linux_container (accessed Sept. 9 2023)

[3] B. Schneier. "Open Source and Security" schneier.com. https://www.schneier.com/crypto-gram/archives/1999/0915.html#OpenSourceandSecurity (accessed Sept. 9 2023)

[4] A. Murray. "The Top 10 Linux Kernel Vulnerabilities You Must Know" mend.io. https://www.mend.io/blog/top-10-linux-kernel-vulnerabilities/ (accessed Sept. 9 2023)

[5] C. Hoffman. "What Does "Everything Is a File" Mean in Linux?" howtogeek.com. https://www.howtogeek.com/117939/htg-explains-what-everything-is-a-file-means-on-linux/ (accessed Sept. 9 2023)

[6] R. Fernandez. "What is Server Virtualization? How It Works, Types, and Examples" serverwatch.com. https://www.serverwatch.com/virtualization/server-virtualization/ (accessed Sept. 9 2023)

[7] Google. "What are Containers?" cloud.google.com. https://cloud.google.com/learn/what-are-containers (accessed Sept. 9 2023)

[8] Microsoft. "Find Azure Marketplace image information using the Azure CLI" learn.microsoft.com. https://learn.microsoft.com/en-us/azure/virtual-machines/linux/cli-ps-findimage (accessed Sept. 9 2023)

[9] Microsoft. "Introduction to Container registries in Azure" learn.microsoft.com. https://learn.microsoft.com/en-us/azure/container-registry/container-registry-intro (accessed Sept. 9 2023)

[10] Palo Alto Networks. "What Is a Remote Access VPN?" https://www.paloaltonetworks.com/cyberpedia/what-is-a-remote-access-vpn (accessed Sept. 9 2023)

[11] Microsoft. "What is Azure VPN Gateway?" learn.microsoft.com https://learn.microsoft.com/en-us/azure/vpn-gateway/vpn-gateway-about-vpngateways (accessed Sept. 9 2023)

[12] Nginx. "What Is Load Balancing?" nginx.com. https://www.nginx.com/resources/glossary/load-balancing/ (accessed Sept. 9 2023)

[13] Microsoft. "What is Azure Load Balancer?" learn.microsoft.com. https://learn.microsoft.com/en-us/azure/load-balancer/load-balancer-overview (accessed Sept. 9 2023)

[14] Microsoft. "Overview of autoscale with Azure Virtual Machine Scale Sets" learn.microsoft.com. https://learn.microsoft.com/en-us/azure/virtual-machine-scale-sets/virtual-machine-scale-sets-autoscale-overview (accessed Sept. 9 2023)

[15] I. Khan. "Why System Monitoring is Required? Importance of System Monitoring" linkedin.com. https://www.linkedin.com/pulse/why-system-monitoring-required-importance-imtiyaz-khan/ (accessed Sept. 9 2023)

[16] Microsoft. "Azure Monitor overview" learn.microsoft.com. https://learn.microsoft.com/en-us/azure/azure-monitor/overview (accessed Sept. 9 2023)

[17] J. Kiarie. "The 8 Best Open Source Web Servers" tecmint.com. https://www.tecmint.com/best-open-source-web-servers/ (accessed. Sept. 9 2023)

[18] Microsoft. "IIS Web Server Overview" learn.microsoft.com. https://learn.microsoft.com/en-us/iis/get-started/introduction-to-iis/iis-web-server-overview (accessed. Sept. 9 2023)

[19] M. Anderson, K. Juell, J. Horcasitas. ”How To Set Up vsftpd for a User’s Directory on Ubuntu 20.04” digitalocean.com. https://www.digitalocean.com/community/tutorials/how-to-set-up-vsftpd-for-a-user-s-directory-on-ubuntu-20-04 (accessed Sept. 9 2023)
[20] Microsoft. ”Configure FTP with IIS Manager Authentication in IIS 7” learn.microsoft.com. https://learn.microsoft.com/en-us/iis/publish/using-the-ftp-service/configure-ftp-with-iis-manager-authentication-in-iis-7 (accessed Sept. 9 2023)
[21] M. Friedl. ”sftp-server — OpenSSH SFTP server subsystem” man.openbsd.org. https://man.openbsd.org/sftp-server.8 (accessed Sept.9 2023)
[22] Microsoft. ”Get started with OpenSSH for Windows” learn.microsoft.com. https://learn.microsoft.com/en-us/windows-server/administration/openssh/openssh_install_firstuse (accessed Sept. 9 2023)
[23] Microsoft. ”What is SQL Server?” learn.microsoft.com. https://learn.microsoft.com/en-au/sql/sql-server/what-is-sql-server?view=sql-server-ver16 (accessed Sept. 9 2023)
[24] S. Ravoof. ”The Best in Open Source Database Software: Top 10 Picks”. https://kinsta.com/blog/open-source-database/ (accessed Sept. 9 2023)
[25] Arch Wiki. ”Mail server” wiki.archlinux.org. https://wiki.archlinux.org/title/Mail_server (accessed Sept. 9 2023)
[26] Microsoft. ”Exchange Server documentation” learn.microsoft.com. https://learn.microsoft.com/en-us/exchange/exchange-server?view=exchserver-2019 (accessed. Sept. 9 2023)