|  |  |
| --- | --- |
| Analysing The Raspberry Pi as a COST-EFFECTIVE Solution for SMEs  Cardiff University School of Computer Science and Informatics | Abstract  This project will analyse the Raspberry Pi as a low energy consumption and low-cost solution for SMEs and start-ups. This will look at the Raspberry Pi vs a traditional server set up vs a cloud solution.  Rhys Connor  Supervisor: Martin Caminada Moderator: Crispin Cooper |

# Acknowledgements

I would like to thank my supervisor Martin Caminada and my parents for all the support throughout the project. As well as all the support received from Cisco Systems, Country Connect in helping facilitate physical elements of the project and Dave Smith for providing layer 1 support on the UCS used in the project.

Table of Contents

[Acknowledgements 2](#_Toc101259503)

[Table of Figures 5](#_Toc101259504)

[1 Introduction 8](#_Toc101259505)

[Requirement 1 8](#_Toc101259506)

[Requirement 2 8](#_Toc101259507)

[Requirement 3 8](#_Toc101259508)

[Requirement 4 9](#_Toc101259509)

[E-Waste 9](#_Toc101259510)

[2 Background 9](#_Toc101259511)

[Explanation of Services 9](#_Toc101259512)

[Pi-Hole/DNS 9](#_Toc101259513)

[NAS 10](#_Toc101259514)

[Webserver 10](#_Toc101259515)

[AD 11](#_Toc101259516)

[Honeypot 11](#_Toc101259517)

[DHCP 11](#_Toc101259518)

[CRM 12](#_Toc101259519)

[IPS and IDS 12](#_Toc101259520)

[Portainer 12](#_Toc101259521)

[Other Services to Note 12](#_Toc101259522)

[RPi 13](#_Toc101259523)

[Traditional Server Set-up 14](#_Toc101259524)

[Cloud 14](#_Toc101259525)

[Hardware Costs of Each System 15](#_Toc101259526)

[Raspberry Pi 15](#_Toc101259527)

[Cisco UCS C220-M3 15](#_Toc101259528)

[GCP 15](#_Toc101259529)

[3 Approach 15](#_Toc101259530)

[Hardware Decisions 15](#_Toc101259531)

[OS/Platform 17](#_Toc101259532)

[RPi OS 18](#_Toc101259533)

[Debian 19](#_Toc101259534)

[Proxmox vs ESXi 19](#_Toc101259535)

[GCP 19](#_Toc101259536)

[Raw Performance 20](#_Toc101259537)

[CPU/Memory 20](#_Toc101259538)

[Network 21](#_Toc101259539)

[How to Benchmark Performance of Different Services 22](#_Toc101259540)

[DNS/Pi-Hole 22](#_Toc101259541)

[AD 23](#_Toc101259542)

[Webserver 23](#_Toc101259543)

[DHCP 24](#_Toc101259544)

[NAS 25](#_Toc101259545)

[IaC 25](#_Toc101259546)

[Docker 25](#_Toc101259547)

[Portainer 26](#_Toc101259548)

[Bash 26](#_Toc101259549)

[4 Implementation 28](#_Toc101259550)

[Network Infrastructure 28](#_Toc101259551)

[Home 28](#_Toc101259552)

[DMZ 29](#_Toc101259553)

[Docker Infrastructure 29](#_Toc101259554)

[VM Infrastructure 30](#_Toc101259555)

[IaC Examples 30](#_Toc101259556)

[Portainer 30](#_Toc101259557)

[Pi Hole 30](#_Toc101259558)

[5 Results 31](#_Toc101259559)

[Raw Performance Results 31](#_Toc101259560)

[Iperf Network Performance Test 31](#_Toc101259561)

[Timed Linux Kernel Compile 33](#_Toc101259562)

[Power Draw of Systems 39](#_Toc101259563)

[Comparison of RPi 4 vs UCS C220-M3S Power Draw 43](#_Toc101259564)

[Results of Performing Services on RPi 4 44](#_Toc101259565)

[DNS 44](#_Toc101259566)

[DHCP 46](#_Toc101259567)

[AD 46](#_Toc101259568)

[NAS 46](#_Toc101259569)

[Web Server 47](#_Toc101259570)

[6 Evaluation and Analysis 49](#_Toc101259571)

[Raw Performance 49](#_Toc101259572)

[Sysbench 49](#_Toc101259573)

[Timed Linux Kernel Compile 49](#_Toc101259574)

[Service Testing 52](#_Toc101259575)

[DNS/Pi-Hole 52](#_Toc101259576)

[DHCP 52](#_Toc101259577)

[AD 53](#_Toc101259578)

[NAS 53](#_Toc101259579)

[Webserver 53](#_Toc101259580)

[Energy Usage of Systems 53](#_Toc101259581)

[Energy Cost of Linux Kernel Compile 53](#_Toc101259582)

[Energy Cost of Idle Time Between Business Hours 54](#_Toc101259583)

[Energy Cost to Run Systems 55](#_Toc101259584)

[Cost To Run Analysis 55](#_Toc101259585)

[7 Future Work 55](#_Toc101259586)

[8 Conclusions 55](#_Toc101259587)

[9 Reflection on Learning 55](#_Toc101259588)

[Challenges Faced 55](#_Toc101259589)

[References 56](#_Toc101259590)

# Table of Figures

[Figure 1 – A Live/Production Pi-Hole dashboard using Pi-Hole’s Dark Theme 10](#_Toc101259431)

[Figure 2 – An example Honeynet from imperva.com (Imperva, 2022) 11](#_Toc101259432)

[Figure 3 – A demo glass-isc-dhcp dashboard (Miles, 2020) 12](#_Toc101259433)

[Figure 4 - NoTouch Centre (Stratodesk, 2022) 13](#_Toc101259434)

[Figure 5 Mechanical Drawing of Raspberry-Pi 4 to illustrate the size of the device (Raspberry Pi Foundation, 2022) 14](#_Toc101259435)

[Figure 6 A pricing table for Microsoft Azure (Microsoft Azure, 2022) 15](#_Toc101259436)

[Figure 7 - Hardware Requirements for Server OS Installations 16](#_Toc101259437)

[Figure 8 - Table Showing Hardware Requirements of Bitwarden (Bitwarden, Inc, 2022) 16](#_Toc101259438)

[Figure 9 – Cisco UCS M3 Boot Screen 17](#_Toc101259439)

[Figure 10 - docker-compose.yml file to Create a Pi-Hole Container 18](#_Toc101259440)

[Figure 11 – Table from Veritis comparing CSPs (Veritis, n.d.) 20](#_Toc101259441)

[Figure 12 - Average Deviation of Linux Kernel Compile Runs (Open Benchmarking Org, 2022) 21](#_Toc101259442)

[Figure 13 - Time to Complete Linux Kernel Compile (Open Benchmarking Org, 2022) 21](#_Toc101259443)

[Figure 14 - Physical Network Example of How iperf Test Was Performed 21](#_Toc101259444)

[Figure 15 - DNS Benchmark GUI (Gibson Research Corporation, 2018) 22](#_Toc101259445)

[Figure 16 - Successful Connection of Client to AD DC 23](#_Toc101259446)

[Figure 17 - Screenshot of Webserver Main Page 24](#_Toc101259447)

[Figure 18 - Section of the /etc/dhcp/dhcpd.conf File used by isc-dhcp-server 25](#_Toc101259448)

[Figure 19 - Provision Machine Bash Script Used 27](#_Toc101259449)

[Figure 20 - Network Diagram of Home Configuration 28](#_Toc101259450)

[Figure 21 - Network Diagram of DMZ Infrastructure for UCS 29](#_Toc101259451)

[Figure 22 - Portainer Configuration for RPi 29](#_Toc101259452)

[Figure 23 - Diagram of Portainer Config for RPi 29](#_Toc101259453)

[Figure 24 - Portainer Docker Compose Example 30](#_Toc101259454)

[Figure 25 - Pihole Docker Compose Example 31](#_Toc101259455)

[Figure 26 - Table Outlining Iperf Performance of Systems 31](#_Toc101259456)

[Figure 27 - Result of Iperf Test on RPi 4 32](#_Toc101259457)

[Figure 28 - Result of Iperf Test on UCS C220-M3S 32](#_Toc101259458)

[Figure 29 - Visualisation of Iperf Results 32](#_Toc101259459)

[Figure 30 - Linux Kernel Compile Failed Attempt on RPi 4 34](#_Toc101259460)

[Figure 31 - First Run of defconfig Linux Kernel Compile on RPi 4 35](#_Toc101259461)

[Figure 32 - Second Run of defconfig Linux Kernel Compile on RPi 4 35](#_Toc101259462)

[Figure 33 - Third Run of defconfig Linux Kernel Compile on RPi 4 36](#_Toc101259463)

[Figure 34 - A Failed attempt to benchmark compiling the Linux Kernel on the UCS 38](#_Toc101259464)

[Figure 35 - First Run of defconfig Linux Kernel Compile on UCS 38](#_Toc101259465)

[Figure 36 - Second Run of defconfig Linux Kernel Compile on UCS 39](#_Toc101259466)

[Figure 37 - Third Run of defconfig Linux Kernel Compile on UCS 39](#_Toc101259467)

[Figure 38 - Output of top Command from RPi 4 2GB 40](#_Toc101259468)

[Figure 39 - Output of HTOP Command from RPi 4 4GB 40](#_Toc101259469)

[Figure 40 - Table of Power Draw Readings Gathered from USB-C to USB-C Multimeter 40](#_Toc101259470)

[Figure 41 - Image of USB-C to USB-C Multimeter Reading Idle power draw of RPi 4 41](#_Toc101259471)

[Figure 42 - Multimeter Configuration to Check Accuracy of USB-C to USB-C Multimeter 42](#_Toc101259472)

[Figure 43 - Table of Power Draw Readings Gathered from Outlet Multimeter 42](#_Toc101259473)

[Figure 44 - Table of Power Draw Readings Gathered from Panduit G5 Web GUI 43](#_Toc101259474)

[Figure 45 - Power Draw as Measured from Panduit Web GUI 43](#_Toc101259475)

[Figure 46 - Power Draw as Measured from CIMC 43](#_Toc101259476)

[Figure 47 - Graph Illustrating the Power Draw of Systems 44](#_Toc101259477)

[Figure 48 - Table of Pi-Hole Time to Resolve Times 44](#_Toc101259478)

[Figure 49 - Table of Virgin Media’s DNS Time to Resolve Times 45](#_Toc101259479)

[Figure 50 - Table of Google DNS Time to Resolve Times 45](#_Toc101259480)

[Figure 51 - Table of OpenDNS Time to Resolve Times 45](#_Toc101259481)

[Figure 52 - HTOP Output Running Dhammer DHCP Stress Test 46](#_Toc101259482)

[Figure 53 - HTOP Output Whilst 1 User Is Logging onto Client Machine 46](#_Toc101259483)

[Figure 54 - HTOP Output from Webserver Handling 400 Requests 47](#_Toc101259484)

[Figure 55 - Example of CLI Output from Wrk 47](#_Toc101259485)

[Figure 56 - Table of Results For 50 Connections 47](#_Toc101259486)

[Figure 57 - Table of Results For 100 Connections 48](#_Toc101259487)

[Figure 58 - Table of Results For 200 Connections 48](#_Toc101259488)

[Figure 59 - Table of Results For 400 Connections 48](#_Toc101259489)

[Figure 60 - Table of Linux Kernel Compile Times on RPi 4 50](#_Toc101259490)

[Figure 61 - Boxplot Graph of RPi 4 Time to Compile Linux Kernel 50](#_Toc101259491)

[Figure 62 - Table of Linux Kernel Compile Times on RPi 3 50](#_Toc101259492)

[Figure 63 - Boxplot Graph of RPi 3 Time to Compile Linux Kernel 50](#_Toc101259493)

[Figure 64 - Table of Linux Kernel Compile Times on UCS C220-M3S 51](#_Toc101259494)

[Figure 65 - Boxplot Graph of UCS C220-M3S Time to Compile Linux Kernel 51](#_Toc101259495)

[Figure 66 - Bar Chart Highlighting Standard Deviation Between Systems 51](#_Toc101259496)

[Figure 67 - Bar Chart Highlighting the Average Time to Compile Across Systems 52](#_Toc101259497)

[Figure 68 - Table Comparing Average Time to Compile on Systems 53](#_Toc101259498)

[Figure 69 - Table Comparing Idle Power Draw of Systems 54](#_Toc101259499)

[Figure 70 - Table Outlining Energy Usage of RPi 4 in Idle Hours 54](#_Toc101259500)

[Figure 71 - Table Outlining Energy Usage of UCS C220-M3S in Idle Hours 54](#_Toc101259501)

[Figure 72 - Table Outlining Power Usage of Systems in a Working Week 55](#_Toc101259502)

Table of Acronyms

|  |  |
| --- | --- |
| Domain Name Services | DNS |
| Dynamic Host Configuration Protocol | DHCP |
| Small-Medium Enterprises | SMEs |
| Network Attached Storage | NAS |
| Active Directory | AD |
| Customer Relationship Management | CRM |
| Packet Capture/Packet Analysis | PCAP/PA |
| Operating System | OS |
| Cloud Service Provider | CSP |
| Central Processing Unit | CPU |
| Error Correction Code | ECC |
| Random Access Memory | RAM |
| Double Data Rate 3 | DDR3 |
| Small Computer Systems Interface | SCSI |
| Serial Advanced Technology Attachment | SATA |
| Serial Attached SCSI | SAS |
| Redundant Array of Inexpensive Disks | RAID |
| Raspberry-Pi | RPi |
| Intrusion Prevention System | IPS |
| Intrusion Detection System | IDS |
| Not Suitable for Work | NSFW |
| Power Distribution Unit | PDU |
| Demilitarized Zone | DMZ |
| Infrastructure as Code | IaC |
| Google Cloud Platform | GCP |
| Value Added Tax | VAT |
| Virtual Machine | VM |
| Graphical User Interface | GUI |
| Cisco Integrated Management Console | CIMC |
| Unified Computing System | UCS |
| Managed Service Provider | MSP |
| Gigabyte | GB |
| Megabits Per Second | Mbps |
| Active Directory Domain Controller | AD DC |
| Samba | SMB |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# 1 Introduction

The objective of this project is to analyse varying models of RPi’s as low energy consumption and low-cost solution for SMEs and start-ups. This will look at the RPi, analyse its performance and cost to performance when compared to a more traditional server set up and compare the cost of these solutions to what a company could also get within a cloud solution. This will also compare the reasons why these companies may want to or not want to use cloud or traditional server setups over a RPi.

This project aims to address the issue of rising energy costs and hardware costs for businesses by either initially hosting or migrating services from a traditional server or cloud infrastructure to a RPi or a cluster of RPi’s. Both the running costs and the initial hardware costs (if applicable) will be under investigation. Some of the services that will be tested on these systems will be AD, DNS, NAS and a company Webserver.

Although some practical uses of the RPi for business have been covered in the past these have been typically in-depth analysis of a single-use case of the RPi for a business, for example:

* Low-Cost network monitoring system (Maulana & Al-Khowarizmi, 2021)
* Low-Cost Real-Time System monitor (Nguye, et al., 2015)
* Intrusion Detection System (de la Cruz, et al., 2016)
* Low-Cost Small Business Brewing (Acácio de Andrade, et al., 2020)

The systems that will be analysed throughout this project are a RPi 4 Model B 2GB, RPi 4 Model B 4GB and a Cisco UCS C220-M3S.

Depending on the time constraints of the project services such as DHCP, Self-Hosted Company CRM, Honeypot and a PCAP/PA server can also be investigated.

To analyse this there will be a compilation of metrics gathered, raw performance metrics, power draw and performance statistics of the services hosted on these devices. Initial assumptions are that the RPi will be significantly lower power draw but also be significantly lower performance. This may however be ideal for services such as AD which are not as resource intensive as PA.

The project can be broken down into the following requirements:

## Requirement 1

What is the raw performance of the RPi vs server?

## Requirement 2

What is the difference in the power draw of the RPi vs server?

## Requirement 3

Using the data pulled from Requirements 1 & 2 we can then find the comparative performance per watt of the RPi vs server.

## Requirement 4

The performance of the RPi vs server when hosting the key requirement services listed below:

* AD
* DNS
* NAS
* Company Webserver
* DHCP

This project is intended to give a detailed insight into the tools and services that SMEs could utilise a RPi for without causing any impact on regular business operation.

## E-Waste

Electronic waste (E-Waste) is the waste produced in both the production and disposal of electronic equipment. Over the past decade, technology and electrical equipment have become far more ubiquitous than in prior decades (Insert figure of how many devices are in a house now vs 10 years ago). This project hopes to highlight to businesses that firstly any old tech that they have can be utilised for tasks like these and that if they invest in technology like an RPi 4 Model B there are other non-traditional uses for this as they grow from a small into a medium and even large enterprise.

# 2 Background

## Explanation of Services

### Pi-Hole/DNS

Pi-Hole is a free open-source DNS sinkhole (Pi-Hole, 2022), this can also be referred to as a network-wide advertisement blocker and DNS forwarder. This can be utilised by SMEs and Start-ups as a tool to manage the websites that employees can access and filter out any websites that are NSFW. This can be beneficial as it can allow the company to monitor all the devices that are trying to access NSFW sites at work and identify any sites that may need blocking in the future. See the below Pi-Hole dashboard example taken from my home network set up:

Graphical user interface, chart

Description automatically generated

Figure 1 – A Live/Production Pi-Hole dashboard using Pi-Hole’s Dark Theme

*Note: This configuration of Pi-Hole is running on a docker container on a Raspberry-Pi 4 Model B 4GB*

* Does this explain that Pi-Hole isn’t itself a DNS server?
* Should I cover a dedicated DNS server as well or is Pi-Hole sufficient?
* Should I write up why Pi-Hole was used and not something like AdGuard – mention commercial versions
* <https://technitium.com/dns/>
* <https://nextdns.io/>

### NAS

A NAS can be utilised in several ways for the types of businesses studied in this project. This can be utilised as a generally shared network drive for all users, set up as a backup location for user’s documents and folders and even utilised as a shared drive with folders for individual teams that are managed using user groups to allow for the creation of groups for individual teams. This can be done by either setting up a samba (SMB) share for Linux this can be done by installing samba or with Windows this is supported natively the file system must be formatted in a format that is readable by both Windows and Linux for example File Allocation Table (FAT), Extensible File Allocation Table (exFAT)and New Technology File System (NTFS).

### Webserver

A webserver is utilised by businesses to host their customer-facing webpage and/or any custom internal tools that they may have built to streamline their internal processes.

### AD

AD is a Microsoft developed service that has alternatives and applications that allow for Linux servers to be the AD for the Windows clients. Microsoft defines AD as “*Active Directory stores information about objects on the network and makes this information easy for administrators and users to find and use. Active Directory uses a structured data store as the basis for a logical, hierarchical organization of directory information.”* (Microsoft, 2022). This is beneficial for all scales of business as it can allow for the creation of users and user groups. It also allows for the management of user permissions which can help a company to secure their network and devices from end-users installing malicious software. This can all be managed through a single AD server, instead of the local user approach where system administrators would have to go through each system when someone leaves to remove their user profile. AD resolves this by being the central management system for the businesses users and their permissions.

### Honeypot

A honeypot in terms of technology is a server that imitates another server that is of high interest to a malicious attacker. This can be beneficial as these can be configured to send notifications to the IT team of the business. With many of these Honeypots’ companies can also implement what is known as a Honeynet, this is a network of Honeypots that mimic a full company network. See the below diagram of an example Honeynet.

Graphical user interface, diagram

Description automatically generated

Figure 2 – An example Honeynet from imperva.com (Imperva, 2022)

### DHCP

DHCP is a protocol used within networking to provide client devices connecting to that network an Internet Protocol (IP) address and all the additional network configuration information required, for example, subnet mask and default gateway. This is under the optional requirements as a lot of business routers and ISP-provided routers will host their DHCP server making this a non-essential requirement, however, it may be beneficial for businesses to host their DHCP server like Linux’s isc-dhcp-server with a glass-isc-dhcp (Miles, 2020) web management portal as this will allow them to potentially have easier control over their DHCP leases and DHCP configuration than they may get with their ISP provided router. See the below example of a glass-isc-dhcp dashboard:

Graphical user interface, application

Description automatically generated

Figure 3 – A demo glass-isc-dhcp dashboard (Miles, 2020)

### CRM

A CRM is a tool used by businesses to manage their customer relationships and even store information about customers who are potential leads. This tool will typically have a Webserver and Database element to it however a lot of CRM companies are now offering their services as a SaaS solution eliminating the need for a business to host this internally on their own servers.

### IPS and IDS

PCAP/PA is often utilised within systems to perform IPS and IDS for the LAN. This is a useful tool used by security analysts to monitor network traffic and can be used to identify any atypical and/or malicious network traffic. This can be helpful to identify if the business’s internal network has been compromised or if suspicious activity is occurring on the network. This has been previously investigated using a RPi 3 Model B, this worked, however, the researchers noticed a limitation with the RAM of the system (peaking at around 90% capacity) (de la Cruz, et al., 2016). This hardware limitation is where the RPi 4 Model B 4GB or 8GB models can further the throughput capacity as this system was tested with 5 clients and peaked around 29Mbps of network traffic.

### Portainer

Portainer is “A centralised service delivery platform for containerized apps” (Portainer, 2022). This platform can be used as a web management GUI where users who are less experienced can

### Other Services to Note

The below services will most likely not be tested within this project however are noteworthy as they either already have RPi distributions of the services themselves or they have low power requirements that allow them to be run on a RPi.

* Stratodesk – Can be used by MSPs like Country Connect
* Systems monitor (Grafana)
* The use of the RPi as a Desktop – Power consumption difference to SFF PC for web browsing
* Private Cloud using Nextcloud, Own Cloud or similar services
* 3CX VoIP server <https://www.3cx.com/docs/recommended-hardware-specifications-for-3cx/>
* Mail server - <https://mailu.io/1.9/compose/setup.html>
* NTP server

Citrix describes a VDI as “the hosting of desktop environments on a central server.” (Citrix, 2022), VDI Servers are typically used in a variety of use cases as outlined below (VMWare, 2022):

* Remote Work
* Bring Your Own Device
* Task or Shift Work

VDI’s for SMEs can be used by paying an MSP like Country Connect Ltd to host the SMEs hosted desktop infrastructure meaning all the SME has to pay for is the service cost and the cost of electricity for the thin client, whether this is a Dell, HP, or Raspberry Pi. Alternatively, the business could host their own VDI with thin clients or RPi’s.

Stratodesk is a tool that can be utilised by MSPs to manage clients connecting to a VDI with Stratodesk's own NoTouch OS which is Linux based and allows for the MSP to manage all thin clients and their configurations through one browser. NoTouch OS also supports both x86 and ARM CPU architectures meaning Thin Clients and RPi’s can run this OS with Stratodesk offering their own RPi image for the NoTouch OS.

Graphical user interface, website

Description automatically generated

Figure 4 - NoTouch Centre (Stratodesk, 2022)

## RPi

The RPi is a credit card sized computer that can be used for a wide range of applications from Robotics, Desktop computers, Interactive Museum exhibits and government call centres (Ltd, Raspberry-Pi. 2022). The aim of the Raspberry-Pi computers is to drive “down the cost of general-purpose computing…” (Raspberry Pi Foundation, 2022). The base cost of the RPi ranges from £34 for the 1GB Raspberry-Pi 4 Model B to £73.50 8GB Raspberry-Pi 4 Model B (The Pi Hut, 2022). Mechanical drawing of RPi 4 Model B below (Raspberry Pi Foundation, 2022):

Diagram, schematic

Description automatically generated

Figure 5 Mechanical Drawing of Raspberry-Pi 4 to illustrate the size of the device (Raspberry Pi Foundation, 2022)

## Traditional Server Set-up

Within lots of start-ups and small to medium enterprises they would “traditionally” utilise older refurbished enterprise hardware from companies such as Intelligent Servers (Intelligent Servers, 2022) and Bargain Hardware (Bargain Hardware, 2022). Both companies offer business class refurbished IT hardware such as Servers, Workstations, and Desktop PC’s/Laptops. This hardware can be brought at a wide range of prices starting from £125 (including VAT) for a barebones server up to a £69995 for a pre-configured Dell MD3420 (Bargain Hardware, 2022).

## Cloud

Over the past decade the utilisation of Cloud infrastructures has become more and more prevalent within large enterprise. The subscription-based offering provided by the cloud and its ease of scalability was beneficial to large companies as their hardware costs are higher than SMEs where they can also get exclusive contracts with cloud providers. These can then however be costly through the subscription, for example see the below pricing for Microsoft Azure’s mid-range Virtual Machine (VM) offering (Microsoft Azure, 2022):

Graphical user interface

Description automatically generated

Figure 6 A pricing table for Microsoft Azure (Microsoft Azure, 2022)

## Hardware Costs of Each System

### Raspberry Pi

Currently when in stock the RPi 4 Model B can be found for the below prices:

* 1GB model £34
* 2GB model £43.50
* 4GB model £54
* 8GB model £73.50

The above prices all include VAT and reflect the price of the RPi 4 Model B as of February 2022 on the reseller sit [The Pi Hut](https://thepihut.com) (The Pi Hut, 2022). Although the RPi 3 is no longer commercially available the starter kits to these can be typically found for £40-60 on resale sites such as eBay, Facebook Marketplace, and Gumtree.

### Cisco UCS C220-M3

A similar model of Cisco UCS C220-M3S as the one used for this project can be seen as refurbished units for sale on sites like [IT in Stock](https://www.itinstock.com/cisco-ucs-c220-m3-ucsc-c220-m3s-2x-quad-core-e5-2643-330ghz-600gb-24gb-server-48623-p.asp) for £760 Ex VAT or £912 including VAT (IT in Stock, 2022).

### GCP

Although there are no initial hardware costs associated with the GCP the substantial subscription cost associated with this negates the hardware cost in a lot of instances.

# 3 Approach

This section will outline the justification of choices for Hardware, Software and Platforms used in the project.

## Hardware Decisions

The RPi 4 B was the first edition of RPi to offer varying RAM sizes with the RPi. The RPi 4 was also the first iteration of the RPi to include separate lanes to the CPU for network and USB. Prior to the RPi 4 the USB and network shared one CPU lane.

The RPi 4 has the following hardware outlined on their RPi 4 datasheet (Raspberry Pi LTD, 2019):

* Quad core 64-bit ARM-Cortex A72 @ 1.5GHz
* 1, 2 and 4 Gigabyte LPDDR4 RAM options
* H.265 (HEVC) hardware decode (up to 4Kp60)
* H.264 hardware decode (up to 1080p60)
* Supports dual HDMI display output up to 4Kp60
* IEEE 802.11 b/g/n/ac Wireless LAN
* Gigabit Ethernet port (supports PoE with add-on PoE HAT)

Comparatively the RPi 3 has the following hardware outlined on their RPi 4 datasheet (Raspberry Pi LTD, n.d.):

* Quadcore Broadcom Cortex-A53 @ 1.4GHz
* 1GB LPDDR2 SDRAM
* IEEE 802.11.b/g/n/ac wireless LAN
* Gigabit Ethernet over USB 2.0 (maximum throughput 300Mbps)
* H.264, MPEG-4 decode (1080p30)
* H.264 encode (1080p30)

The main differences between these systems are the RAM, CPU and network. The CPU processing power according to [Pass Mark](https://www.cpubenchmark.net/) almost doubles from the RPi 3 to the RPi 4. The score of the RPi 3’s Cortex A53 gets a CPU score of 357 compared to the Cortex A72 of the RPi 4’s CPU score of 666 (PassMark Software, 2022). These scores are calculated from other user’s submissions after running the Pass Mark benchmarking software.

Although it is already known that the RPi will be less powerful than the traditional server however it is also known that Linux has lower hardware requirements than a Windows server instance (see the below table).

|  |  |  |
| --- | --- | --- |
| Hardware Requirements | Debian 11 Server | Windows Server |
| CPU | 1GHz | 1.4GHz |
| RAM | 512MB | 512MB (2GB with Desktop Experience installed) |
| HDD | 10GB | 32GB |

Figure 7 - Hardware Requirements for Server OS Installations

The above table must also be considered alongside the hardware requirements of the services you wish to run on the server, see the below example hardware requirements for Bitwarden a password manager that can be hosted on a local server:

|  |  |  |
| --- | --- | --- |
|  | Minimum | Recommended |
| Processor | x64, 1.4GHz | x64, 2GHz dual core |
| Memory | 2GB RAM | 4GB RAM |
| Storage | 10GB | 25GB |
| Docker Version | Engine 19+ and Compose 1.24+ | Engine 19+ and Compose 1.24+ |

Figure 8 - Table Showing Hardware Requirements of Bitwarden (Bitwarden, Inc, 2022)

For this project the “Traditional” server instance will be demonstrated using a Cisco UCS C220-M3S This is a 1U rackmount server with 2x X79 CPU sockets, up to 512GB DDR3 ECC RAM, up to four 3.5” or up to eight 2.5” SAS/SATA drives, 2 PCIe Gen 3 slots and 2 1GE LAN interfaces on the motherboard (Cisco Systems Inc, 2017). The 2 LAN interfaces will typically be connected with one allowing access to the servers CIMC platform, that can allow for remote configuration and management of the server. The second LAN port will then typically be utilised by the OS/Hypervisor installed on the system. For the instance of this project the hypervisor used will be ESXi installed onto the UCS C220-M3S. The hardware configuration of the UCS C220-M3S in this project is as follows:

* 2x Intel Xeon E5-2643 8 Core 16 thread CPUs @ 3.3GHz
* 16x 8GB 1600MHz DDR3 ECC RAM
* 1x LSI 9271-8i MegaRAID SAS Host Bus Adapter
* 4x 280GB 7200RPM 6Gb/s Toshiba HDD
* 4x 280GB 7200RPM 6Gb/s Seagate HDD

Text

Description automatically generated

Figure 9 – Cisco UCS M3 Boot Screen

The Cisco UCS C220-M3S, outlined above, was used in the place of the traditional server as it was available for me to have one racked in the Cisco Systems, Green Park, Reading Lab DMZ. This allows me to remotely access this UCS. With this available to me, the use of Panduit G5 IP managed PDUs gives greater insight into the power drawn by the server at the wall. Using the Cisco labs DMZ, I was able to access my UCS without access to Cisco’s internal corporate network. My former colleagues were also available to provide me with layer 1 support should I experience any issues.

## OS/Platform

This section will explain the platforms/OS used and why these platforms were chosen.

* Debian
* Raspian
* Proxmox/ESXi
* GCP

I will be using a hypervisor such as Proxmox/ESXi as this accurately represents both what I have seen being utilised in business and what I have read is best practice for servers. A hypervisor, also referred to as a Virtual Machine Monitor, is a piece of software that is used to create, manage, and run VMs (VMWare, 2022). Hypervisors can either be run on top of the host OS, these are classified as Type 2 hypervisors, the other one often behaves like a lightweight OS, and these are Type 1 hypervisors. For this project we Using a hypervisor has become common practice as it allows for the server’s resources to be split up into VMs that are easier to increase the CPU cores, RAM, and HDD space than on a barebones installation. This allows for Systems Administrators and IT team to increase the power of a server with the click of a few buttons as opposed to a barebones installation where the server could be down for hours whilst old CPUs, RAM modules and SSD/HDDs are removed, and new models added. The VM approach on the other hand can take a Systems Administrators and IT team 10-20 Minutes of downtime for the service which is a lot less impactful to everyday business use than the hours that may be needed for a barebones upgrade (Jackson, et al., 2020).

Following the advent of VMs and subsequentially DevOps another approach to hosting applications became more widespread. This technology is Containerization and was popularised by Docker. Docker became widely used and popular to remove the argument between Developers and Operations where applications wouldn’t work on the client machines/servers but would work on the developer’s machine. Docker provided a platform for Developers to build applications and run within a Docker container knowing that this will be platform independent so thus remove the former issue where applications would run on the Developers machine and not run in the production environment. Docker Compose was then developed as a tool to enable multi container applications, for example a developer could write one docker compose file and deploy a Linux-Apache-MySQL-PHP stack for a web application. See the below example of the Docker Compose file used to deploy Pi-Hole in the project:

version: "3"

# More info at https://github.com/pi-hole/docker-pi-hole/ and https://docs.pi-hole.net/

services:

  pihole:

    container\_name: pihole

    image: pihole/pihole:latest

    ports:

      - "53:53/tcp"

      - "53:53/udp"

      - "67:67/udp"

      - "80:80/tcp"

      - "443:443/tcp"

    environment:

      TZ: 'Europe/London'

      WEBPASSWORD: 'INSERT-PASSWORD'

    networks:

      your-network:

        ipv4\_address: 192.168.1.3

# Volumes store your data between container upgrades

    volumes:

      - './etc-pihole/:/etc/pihole/'

      - './etc-dnsmasq.d/:/etc/dnsmasq.d/'

    # Recommended but not required (DHCP needs NET\_ADMIN)

    #   https://github.com/pi-hole/docker-pi-hole#note-on-capabilities

    cap\_add:

      - NET\_ADMIN

    restart: unless-stopped

networks:

  your-network:

    external:

      name: name-of-your-docker-network

Figure 10 - docker-compose.yml file to Create a Pi-Hole Container

### RPi OS

For this project it has been decided that the OS to be used for the RPi will be RPi OS Lite (64-bit) for the RPi 4 and used for the RPi 3 to test the raw performance of this system. This is the chosen OS as this is developed and maintained by the RPi Foundation, the version used for the testing is as follows (Raspberry Pi Foundation, 2022):

* Release date: January 28th, 2022
* System: 64-bit
* Kernel version: 5.10
* Debian version: 11 (bullseye)
* Size: 435MB
* SHA256 file integrity hash: d694d2838018cf0d152fe81031dba83182cee79f785c033844b520d222ac12f5

It was decided that for this testing the Lite version would be used as this share’s similarities to the Server releases of other Linux Distributions like Debian Server, Ubuntu Server etc.

### Debian

For the traditional server the operating system of choice is Debian 11 because this distro of Linux is what the RPi OS is built upon and derived from. As shown above in the RPi OS outline. With the RPi OS also being Debian 64bit and Debian based means that the commands run on a Debian machine will also be the same as the commands run on a RPi OS machine. With the exception of the leading sudo as this is not installed by default with a barebones Debian 11 Server install. Details of the version used is listed below (Debian Org, 2021):

* Release Date: December 18th, 2021
* Kernel Version: 5.10
* Size: 378MB
* SHA512 file integrity hash: c685b85cf9f248633ba3cd2b9f9e781fa03225587e0c332aef2063f6877a1f0622f56d44cf0690087b0ca36883147ecb5593e3da6f965968402cdbdf12f6dd74

### Proxmox vs ESXi

* Why did I choose to use a Hypervisor?
  + Most industry use them
  + Allow for better resource scaling to services and VMs on these
  + Cite (Jackson, et al., 2020) for Bare Metal vs Hypervisor installations
* Why ESXi and not Proxmox or Hyper-V?
  + Industry experience setting up and configuring ESXi
* Should I use Proxmox as it is free and open source? Especially as this is a cost-based report and analysis

### GCP

GCP is one of the big 3 cloud providers, these outlined below:

* Amazon Web Services
* Microsoft Azure
* Google Cloud Platform

Comparing the offerings of these three providers in the table below (Figure 11) , It can be seen that GCP has a cheaper but similar offering to AWS at the entry level and offers higher CPU counts but less RAM in the high end at similar costs to both AWS and Azure.

|  |  |  |  |
| --- | --- | --- | --- |
| Machine Type | AWS | Azure | GCP |
| Smallest Instance | An instance with 2 virtual CPUs and 8 GB RAM will cost you around USD69/month. | An instance with 2 virtual CPUs and 8 GB RAM will cost you around USD70/month. | Instance with 2 virtual CPUs and 8 GB RAM will cost you around USD52/month. |
| Largest Instance | Largest instance that includes 3.84 TB RAM and 128 vCPUs will cost you around USD 3.97/hour. | Largest instance that includes 3.89 TB RAM and 128 vCPUs will cost you around USD 6.79/hour. | Largest instance that includes 3.75 TB RAM and 160 vCPUs will cost you around USD 5.32/hour. |

Figure 11 – Table from Veritis comparing CSPs (Veritis, n.d.)

These differences mean that GCP is good to use for the comparisons in this project. As the small businesses will likely use the systems on the lower end of the scale where GCP is more cost effective.

## Raw Performance

This section will outline how the raw performance benchmarks for each system will be achieved.

### CPU/Memory

To stress-test the CPU and memory in the system the following tools can be used to test these:

* GeekBench
* Sysbench
* Hard Info
* Phoronix Test Suite

The test suite that will be used for this project will be Sysbench/GeekBench. Testing will also be performed using Phoronix Test Suite to compile Firefox and timing how long this takes to compile. Compiling an application such as Firefox or the Linux Kernel is a good raw performance test of a system as this process puts a lot of strain on both CPU and Memory in the system, this is also very close to the real-world use of a software build server that is utilised in a development environment. Compiling software such as the Linux Kernel, Google Chrome and Firefox is often also used when technology reviewers like LinusTechTips, Level1Techs and Gamers Nexus.

The below commands were used to install and run the raw performance benchmarks:

* wget http://phoronix-test-suite.com/releases/repo/pts.debian/files/phoronix-test-suite\_7.8.0\_all.deb
* sudo apt install gdebi-core
* sudo gdebi phoronix-test-suite\_7.8.0\_all.deb
* phoronix-test-suite --version
* phoronix-test-suite benchmark build-linux-kernel

Initially start with the sysbench benchmarks as these are the least time consuming and can give an immediate insight into the raw performance of individual components of each system. Then move onto the Linux Kernel compile as it is a real-world use and stresses the whole system. As highlighted by Passmark Software’s online comparison we can expect that the RPi 4 will perform about 1/10th of the performance of the UCS C220-M3S in CPU heavy workloads (Passmark Software, 2022), The below link gives a more detailed comparison of the RPi 4 CPU vs the CPU in the UCS used:

* <https://www.cpubenchmark.net/compare/ARM-Cortex-A72-4-Core-1800-MHz-vs-Intel-Xeon-E5-2643/4078vs1217>

The timed Linux kernel build provides 2 options of how to run the test:

* Defconfig
* Allmodconfig

The defconfig option builds the config file with the default settings based on the systems architecture, where the allmodconfig builds a config file that makes as many parts of the Linux kernel as possible a module. The below graphs pulled from Open Benchmarking show the spread of data from all uploaded results of this Linux kernel build.

Chart, scatter chart

Description automatically generated

Figure 12 - Average Deviation of Linux Kernel Compile Runs (Open Benchmarking Org, 2022)

Chart

Description automatically generated

Figure 13 - Time to Complete Linux Kernel Compile (Open Benchmarking Org, 2022)

### Network

To test the network performance of the devices the below physical network was used:

Diagram, timeline

Description automatically generated

Figure 14 - Physical Network Example of How iperf Test Was Performed

To test the NIC on the devices the following commands were run:

Host A

* iperf -s

Host B

* iperf -c <IP ADDRESS OF HOST A>

In all instances the hardware of each device was connected into a gigabit switch over Cat6 cable which is rated up to gigabit speeds. This was used over Cat6a as none of the NICs used on any of the devices were more than gigabit NICs.

## How to Benchmark Performance of Different Services

This section will outline how each individual service is intended to be benchmarked.

### DNS/Pi-Hole

The performance of the DNS/Pi-Hole service will be tested using the Gibson Research Corporation’s DNS benchmark software. This software allows the testing of multiple input DNS servers, This allowed for testing of both local DNS servers, ISP DNS servers and public DNS servers such as Google DNS, OpenDNS and Cloudflare DNS.

A picture containing graphical user interface

Description automatically generated

Figure - DNS Benchmark GUI (Gibson Research Corporation, 2018)

The GRC DNS benchmark software provides detailed feedback on the performance of each DNS server tested with statistics such as Cached Name, Uncached Name and DotCom Lookup results even being granular with these providing the minimum, maximum, average and reliability results for each of the types of queries.

### AD

To test the feasibility of running the RPi as an AD DC, a RPi 4 will be set up using SMB which is an open-source technology which is compatible with Windows AD. Although SMB is compatible it does not come with the full feature set of a Windows AD DC, this can be limiting as the company grows however for a lot of SMEs SMB is suitable as they wouldn’t use the full feature set of a dedicated Windows AD DC.

The testing will involve configuring the AD DC server to then run 5-20 Windows 10 VMs and link this to the DC and see the HTOP output whilst these VMs are joining the domain.

Graphical user interface, application

Description automatically generated

Figure - Successful Connection of Client to AD DC

* How do you even benchmark such a service?
* This is lightweight and typically a thousand requests can be handled per CPU core – Can’t find documentation to confirm though
* Can this be handled by the RPi? Could I set up a server and get hundreds of machines running joined to the AD?
* <https://wiki.samba.org/index.php/Setting_up_Samba_as_an_Active_Directory_Domain_Controller>
* You can run Windows 10 on ARM – Possibility for server to support this

### Webserver

To test the webserver running on the systems, the below technologies will be used:

* Python
* Flask
* Nginx
* Wrk

The webserver will be running using docker compose, the docker compose file will create both the Flask container containing the static website details and the Nginx container that acts as the web proxy for the Flask container mapping port 80 on the host to port 8080 on the flask container. This website will be tested over HTTP, provided there is time in the project HTTPS may also be tested using a self-signed certificate or free certificate given from LetsEncrypt.

The below is the example website used for testing the webserver:

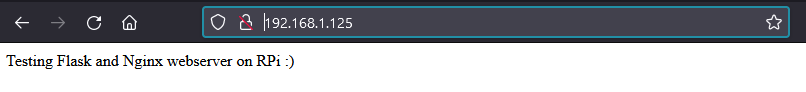


Figure 17 - Screenshot of Webserver Main Page

To utilise wrk and test the webserver the following commands were used:

* wrk -t12 -c50 -d30s --latency http://<IP-OF-HOST>
* wrk -t12 -c100 -d30s --latency http://<IP-OF-HOST>
* wrk -t12 -c200 -d30s --latency http://<IP-OF-HOST>
* wrk -t12 -c400 -d30s --latency http://<IP-OF-HOST>

The above commands generate 50, 100, 200 and 400 concurrent connections using the -c command.

### DHCP

# dhcpd.conf

#

# Sample configuration file for ISC dhcpd

#

# option definitions common to all supported networks...

#option domain-name "example.org";

option domain-name-servers 8.8.8.8, 8.8.4.4;

default-lease-time 600;

max-lease-time 7200;

# The ddns-updates-style parameter controls whether or not the server will

# attempt to do a DNS update when a lease is confirmed. We default to the

# behavior of the version 2 packages ('none', since DHCP v2 didn't

# have support for DDNS.)

ddns-update-style none;

# If this DHCP server is the official DHCP server for the local

# network, the authoritative directive should be uncommented.

authoritative;

# Use this to send dhcp log messages to a different log file (you also

# have to hack syslog.conf to complete the redirection).

#log-facility local7;

# No service will be given on this subnet, but declaring it helps the

# DHCP server to understand the network topology.

#subnet 10.152.187.0 netmask 255.255.255.0 {

#}

subnet 192.168.1.0 netmask 255.255.255.0 {

}

subnet 192.168.2.0 netmask 255.255.255.0 {

    range 192.168.2.64 192.168.2.126;

    option routers 192.168.2.1;

    default-lease-time 7200;

    max-lease-time 28400;

    option domain-name-servers 8.8.8.8, 8.8.4.4;

}

Figure 18 - Section of the /etc/dhcp/dhcpd.conf File used by isc-dhcp-server

Although it was initially thought that the time to receive a DHCP lease would be used to test the performance of the DHCP server the ipchama/dhammer (ipchama, 2021) DHCP stress testing tool was used and a HTOP output from the VM/RPi taken, the below commands were used to initiate the stress test. The below command was used for the RPi 4:

* sudo ./dhammer dhcpv4 --interface eth0 --mac-count 10000 --rps 1000 --maxlife 0 --relay-target-server-ip 192.168.1.125 --relay-source-ip 192.168.1.1

The below command was used for the traditional server set up:

* sudo ./dhammer dhcpv4 --interface ens0 --mac-count 10000 --rps 1000 --maxlife 0 --relay-target-server-ip 7.5.17.236 --relay-source-ip 7.5.17.254

By utilising the HTOP output it can be seen the strain that DHCP puts on the hardware for each of the systems.

### NAS

To test running the NAS the below commands are used on a Windows machine to generate test files of 1GB, 10GB, 100GB and 1TB:

* fsutil file createnew G:\1tb.test 1099511627776
* fsutil file createnew G:\100gb.test 107374182400
* fsutil file createnew G:\10gb.test 10737418240
* fsutil file createnew G:\1gb.test 1073741824

The time to transfer each of these files will then be recorded and from this the average transfer speed can be calculated. Both the time to transfer and the average transfer speed can then be used to evaluate the performance of the RPi 4 vs the UCS C220-M3S.

## IaC

This section outlines the IaC scripts and processes used for each of these.

### Docker

Docker was used to run some of the services outlined above due to the declarative approach meaning that the state wanted from the system is declared within the docker-compose.yml file. This is often hardware agnostic provided the service to be run is supported on both. RedHat describe the declarative approach to IaC as the following “A declarative approach defines the desired state of the system, including what resources you need and any properties they should have, and an IaC tool will configure it for you. “ (RedHat, 2020). This ensures that the service state is the same on all systems tested.

### Portainer

To manage the Docker hosts and the containers running on them Portainer was used with one central Portainer Host and Portainer agents being installed on each host. The below command was used to install the Portainer agent on these (Portainer, 2022):

* sudo docker run -d -p 9001:9001 --name portainer\_agent --restart=always -v /var/run/docker.sock:/var/run/docker.sock -v /var/lib/docker/volumes:/var/lib/docker/volumes portainer/agent:2.9.3

The Portainer host was set up using the below command (Portainer, 2022):

* sudo docker run -d -p 8000:8000 -p 9443:9443 --name portainer \
* --restart=always \
* -v /var/run/docker.sock:/var/run/docker.sock \
* -v portainer\_data:/data \
* portainer/portainer-ce:2.9.3

Using Portainer you also gain access to a whole repository of application templates, there is also the opportunity to create and store custom templates. The custom templates allow for granular control of the containers through a web GUI as opposed to the CLI like Docker is traditionally managed through, using these templates will also ensure easy reproducibility

### Bash

In terms of IaC a Bash script would be an imperative approach to IaC. The imperative approach involves the script/code specifying the steps required to achieve the desired state of the system. This means that all the steps required need to be clear and executable by the system, for example if the script/code specified yum as the package manager and this was being executed on a basic install of Debian this would not work as basic installs of Debian use apt as the default package manager. For the purpose of this project a Bash script was used to provision the machines, setting these up with Docker, Docker Compose and adding the Portainer agent to the system.

#!/bin/bash

sudo apt-get update

sudo apt-get install pip -y

sudo apt-get install ca-certificates curl gnupg lsb-release wget -y

#install docker

curl -fsSL https://get.docker.com -o get-docker.sh

sudo sh get-docker.sh

#Install docker-compose using pip

sudo pip install --upgrade pip

sudo pip install docker-compose

#Check docker is installed and get the Portainer agent

d=$(docker --version)

if [[ $? != 0 ]]; then

    echo "Command failed."

elif [[ $d ]]; then

    echo "Docker is installed"

    #Comment out the below line to not run the portainer agent

    sudo docker run -d -p 9001:9001 --name portainer\_agent --restart=always -v /var/run/docker.sock:/var/run/docker.sock -v /var/lib/docker/volumes:/var/lib/docker/volumes portainer/agent:2.9.3

    #Uncomment the below to add a MacVLAN to the docker config - Change the subnets to match your use case

    #sudo docker network create -d macvlan --subnet=192.168.1.0/24 --gateway=192.168.1.1  -o parent=eth0 sc-net

else

    echo "Docker is not installed"

fi

#Check that docker compose is installed

dc=$(docker-compose --version)

if [[ $? != 0 ]]; then

    echo "Command failed."

elif [[ $dc ]]; then

    echo "Docker Compose is installed"

else

    echo "Docker Compose is not installed"

fi

sudo apt upgrade -y

reboot

Figure 19 - Provision Machine Bash Script Used

# 4 Implementation

## Network Infrastructure

### Home

Diagram

Description automatically generated

Figure 20 - Network Diagram of Home Configuration

The network configuration used for the RPi’s included network segregation with the home devices, IoT devices and Servers all sitting on 3 separate VLANs outlined below:

* VLAN 1 – 192.168.1.x/24
* VLAN 2 – 192.168.2.x/24
* VLAN 3 – 192.168.3.x/24

The RPi’s will operate on VLAN 3, with Docker configured with a MAC VLAN to allow for separate services to operate on different IP addresses.

### DMZ

Diagram

Description automatically generated

Figure 21 - Network Diagram of DMZ Infrastructure for UCS

## Docker Infrastructure

For the services tested these will be run using docker compose and Portainer to manage the containers running.

A picture containing text

Description automatically generated

Figure 22 - Portainer Configuration for RPi

A picture containing icon

Description automatically generated

Figure - Diagram of Portainer Config for RPi

The instance of Portainer is running within a Docker container on sc-pi-1, As well as Portainer sc-pi-1 is also running Pi-Hole in a container.

The below command was used to configure the MacVLAN network to allow different containers to use different IP addresses to the host:

* sudo docker network create -d macvlan --subnet=192.168.1.0/24 --gateway=192.168.1.1 -o parent=eth0 sc-net

## VM Infrastructure

For the traditional server there will be

* Outline the VM
  + The CPU cores (going to match the Pi)
  + Memory
  + Storage

## IaC Examples

### Portainer

version: '3'

services:

  portainer:

    image: portainer/portainer-ce:latest

    container\_name: portainer

    restart: unless-stopped

    security\_opt:

      - no-new-privileges:true

    volumes:

      - /etc/localtime:/etc/localtime:ro

      - /var/run/docker.sock:/var/run/docker.sock:ro

      - ./portainer-data:/data

    networks:

      steep-corner-net:

        ipv4\_address: 192.168.1.2

    ports:

      - 9000:9000

networks:

  steep-corner-net:

    external:

      name: sc-net

Figure - Portainer Docker Compose Example

### Pi Hole

version: "3"

# Script taken from the below github repo

# More info at https://github.com/pi-hole/docker-pi-hole/ and https://docs.pi-hole.net/

services:

  pihole:

    container\_name: pihole

    image: pihole/pihole:latest

    ports:

      - "53:53/tcp"

      - "53:53/udp"

      - "67:67/udp"

      - "80:80/tcp"

      - "443:443/tcp"

    environment:

      TZ: 'Europe/London'

      WEBPASSWORD: '@R9a=+v(Xkg9%,[W'

    # Volumes store your data between container upgrades

    networks:

      steep-corner-net:

        ipv4\_address: 192.168.1.3

    volumes:

      - './etc-pihole/:/etc/pihole/'

      - './etc-dnsmasq.d/:/etc/dnsmasq.d/'

    # Recommended but not required (DHCP needs NET\_ADMIN)

    #   https://github.com/pi-hole/docker-pi-hole#note-on-capabilities

    cap\_add:

      - NET\_ADMIN

    restart: unless-stopped

networks:

  steep-corner-net:

    external:

      name: sc-net

Figure - Pihole Docker Compose Example

# 5 Results

## Raw Performance Results

This section will analyse the results acquired by the raw performance tests below:

* Timed Linux Kernel Compile
* Iperf

### Iperf Network Performance Test

The below table shows the network performance of the systems analysed:

|  |  |  |
| --- | --- | --- |
| System | Transfer | Bandwidth |
| RPi 4 | 1.07GB | 921Mbps |
| RPi 3 | 113MB | 94.7Mbps |
| UCS C220-M3S | 1.09GB | 934Mbps |

Figure - Table Outlining Iperf Performance of Systems

The below is a sample output from the iperf test:Text

Description automatically generated with medium confidence

Figure 27 - Result of Iperf Test on RPi 4

Text

Description automatically generated

Figure 28 - Result of Iperf Test on UCS C220-M3S

Figure - Visualisation of Iperf Results

### Timed Linux Kernel Compile

#### RPi 3

Text

Description automatically generated

<https://openbenchmarking.org/result/2203196-FO-RPI3RUN1804>

Text

Description automatically generated

<https://openbenchmarking.org/result/2203205-FO-RPI3RUN2568>

Text

Description automatically generated

<https://openbenchmarking.org/result/2203201-FO-RPI3RUN3300>

#### RPi4 – 2Gb

Text

Description automatically generated

Figure 30 - Linux Kernel Compile Failed Attempt on RPi 4

<https://openbenchmarking.org/result/2203072-FO-FIRSTRUNR33>

Text

Description automatically generated

Figure 31 - First Run of defconfig Linux Kernel Compile on RPi 4

<https://openbenchmarking.org/result/2203070-FO-SECONDRUN71>

Text

Description automatically generated

Figure 32 - Second Run of defconfig Linux Kernel Compile on RPi 4

Text

Description automatically generated

Figure 33 - Third Run of defconfig Linux Kernel Compile on RPi 4

#### RPi4 – No FLIRC Case

Text

Description automatically generated

<https://openbenchmarking.org/result/2203152-FO-RPI44GBRU67>

<https://openbenchmarking.org/result/2203150-FO-RPI44GBRU48>

Text

Description automatically generated

<https://openbenchmarking.org/result/2203168-FO-RPI44GBRU89>

Text

Description automatically generated

<https://openbenchmarking.org/result/2203168-FO-RPI44GBRU88>

<https://openbenchmarking.org/result/2203173-FO-FIRSTRUNR50>

#### UCS C220-M3S

Text

Description automatically generated

Figure 34 - A Failed attempt to benchmark compiling the Linux Kernel on the UCS

Text

Description automatically generated

Figure 35 - First Run of defconfig Linux Kernel Compile on UCS

Text

Description automatically generated

Figure 36 - Second Run of defconfig Linux Kernel Compile on UCS





Text

Description automatically generated

Figure 37 - Third Run of defconfig Linux Kernel Compile on UCS

#### RPi4 vs UCS C220-M3S

Compare the TTC of the Linux kernel compile

### Power Draw of Systems

#### RPi 4

Output of top command whilst running Linux kernel compile.

Text

Description automatically generated with medium confidence

Figure 38 - Output of top Command from RPi 4 2GB

A picture containing graphical user interface

Description automatically generated

Figure 39 - Output of HTOP Command from RPi 4 4GB

The USB-C to USB-C Multimeter only highlights the volts and amps used by the RPi 4 at the time of measurement. This however means that the Power Law must be applied to calculate the watts. To do this the average voltage and current was taken to perform the calculations.

|  |  |  |  |
| --- | --- | --- | --- |
| Workload | V | A | W |
| Idle | 5.25 | 0.3 | 1.575 |
| Linux Compile 1 | 5.25 | 0.8 | 4.2 |
| Linux Compile 2 | 5.25 | 0.9 | 4.725 |
| Linux Compile 3 | 5.25 | 0.7 | 3.675 |

Figure 40 - Table of Power Draw Readings Gathered from USB-C to USB-C Multimeter

When this data was initially collected it was believed that the figures for power usage from the below Multimeter were wrong. Following this a different variety of Multimeter was ordered. This new Multimeter sits between the power brick and outlet on the wall. This confirmed that there was a slight variation to these figures however not as large a deviation as was initially suspected.

A picture containing USB-C to USB-C multi meter readings

Description automatically generated

Figure 41 - Image of USB-C to USB-C Multimeter Reading Idle power draw of RPi 4

The above image shows the original Multimeter used that sits between the power brick and power cable for the RPi 4. This was then changed in favour of the below configuration that better matched the traditional server setup and how power draw was measured for this.

A picture containing text, wall, kitchen, indoor

Description automatically generated

Figure 42 - Multimeter Configuration to Check Accuracy of USB-C to USB-C Multimeter

The table below shows the readings gathered from the outlet Multimeter. The average from the USB-C to USB-C Multimeter is 4.2W vs the average from the outlet Multimeter of 5.2W. This shows a 1W difference, this could be due to inefficiencies with the power brick.

|  |  |
| --- | --- |
| Test | Average Power Draw in Watts |
| Idle | 2.1 |
| Linux Kernel Compile 1 | 5.2 |
| Linux Kernel Compile 2 | 5.3 |
| Linux Kernel Compile 3 | 5.1 |

Figure 43 - Table of Power Draw Readings Gathered from Outlet Multimeter

#### UCS C220-M3S

|  |  |
| --- | --- |
| Test | Average Power Draw in Watts |
| Idle | 115 |
| Linux Kernel Compile 1 | 300 |
| Linux Kernel Compile 2 | 298 |
| Linux Kernel Compile 3 | 299 |

Figure 44 - Table of Power Draw Readings Gathered from Panduit G5 Web GUI

Background pattern

Description automatically generated with low confidence

Figure 45 - Power Draw as Measured from Panduit Web GUI

Table

Description automatically generated

Figure 46 - Power Draw as Measured from CIMC

## Comparison of RPi 4 vs UCS C220-M3S Power Draw

Using Figure 29 and Figure 30 tables and the data gathered from both the Multimeter and Panduit G5 PDU, a full comparison of the energy cost to complete a Linux kernel compile, the cost of the idle systems outside business hours and consequently the average cost to run each system over business hours.

The graph below (Figure 43) illustrates the power draw of each system in 3 given scenarios bullet pointed below:

* Power draw of the system in an idle state - this is the average power drawn in watts from the outlet.
* Power draw running Linux kernel compile – this is the average power drawn in watts from the outlet.
* Max power draw seen – This is the upper power draw seen from each system during stress tests and benchmarking.

Figure 47 - Graph Illustrating the Power Draw of Systems

## Results of Performing Services on RPi 4

### DNS

#### Time to Resolve

The following sub sections will outline the time to resolve DNS queries for each type of DNS server outlined below:

* RPi 4 – Pi-Hole
* UCS C200-M3S – Pi-Hole
* Virgin Media DNS
* BT DNS
* Google DNS
* OpenDNS

##### RPi 4 – Pi-Hole

Table

Description automatically generated

Figure - Table of Pi-Hole Time to Resolve Times

##### UCS C220-M3S – Pi-Hole

##### Virgin Media DNS

Table

Description automatically generated

Figure - Table of Virgin Media’s DNS Time to Resolve Times

##### BT DNS

62.6.40.178

62.6.40.162

194.72.9.38

194.72.9.34

194.72.0.98

194.72.0.114

194.74.65.68

194.74.65.69

##### Google DNS

Table

Description automatically generated

Figure - Table of Google DNS Time to Resolve Times

##### OpenDNS

Table

Description automatically generated

Figure - Table of OpenDNS Time to Resolve Times

### DHCP

#### RPi 4

Graphical user interface

Description automatically generated

Figure - HTOP Output Running Dhammer DHCP Stress Test

### AD

#### RPi 4

##### HTOP

Graphical user interface

Description automatically generated with medium confidence

Figure 53 - HTOP Output Whilst 1 User Is Logging onto Client Machine

##### Results Found

[2022/04/18 16:02:35.390940, 5] ../../source3/auth/auth.c:51(smb\_register\_auth)

Attempting to register auth backend winbind

[2022/04/18 16:02:35.391022, 5] ../../source3/auth/auth.c:63(smb\_register\_auth)

Successfully added auth method 'winbind'

[2022/04/18 16:02:35.415384, 3] ../../source4/auth/ntlm/auth.c:797(auth\_register)

AUTH backend 'winbind' registered

#### UCS C220-M3S

### NAS

#### RPi 4

#### UCS C220-M3S

### Web Server

#### RPi 4

##### HTOP

Graphical user interface

Description automatically generated

Figure - HTOP Output from Webserver Handling 400 Requests

##### Wrk results

Text

Description automatically generated

Figure - Example of CLI Output from Wrk

12 threads and 50 connections

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Thread Stats | Average | Stdev | Max | +/- Stdev |
| Latency | 44.92ms | 5.15ms | 102.82ms | 80.66 |
| Req/Sec | 89.09 | 9.57 | 121 | 73.67 |

Figure - Table of Results For 50 Connections

Latency Distribution

50% 43.70ms

75% 46.65ms

90% 51.22ms

99% 63.51ms

32037 requests in 30.03s, 6.26MB read

Requests/sec: 1066.96

Transfer/sec: 213.60KB

12 threads and 100 connections

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Thread Stats | Average | Stdev | Max | +/- Stdev |
| Latency | 89.99ms | 6.88ms | 158.59ms | 81.53 |
| Req/Sec | 88.94 | 10.12 | 131 | 73.32 |

Figure - Table of Results For 100 Connections

Latency Distribution

50% 88.89ms

75% 92.53ms

90% 97.69ms

99% 112.33ms

31959 requests in 30.03s, 6.25MB read

Requests/sec: 1064.35

Transfer/sec: 213.08KB

12 threads and 200 connections

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Thread Stats | Average | Stdev | Max | +/- Stdev |
| Latency | 264.15 | 333.49ms | 2.00s | 85.17 |
| Req/Sec | 88.32 | 23.22 | 180 | 73.71 |

Figure - Table of Results For 200 Connections

Latency Distribution

50% 101.57ms

75% 212.75ms

90% 818.47ms

99% 1.38s

31657 requests in 30.04s, 6.19MB read

Socket errors: connect 0, read 0, write 0, timeout 152

Non-2xx or 3xx responses: 5

Requests/sec: 1053.95

Transfer/sec: 211.01KB

12 threads and 400 connections

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Thread Stats | Average | Stdev | Max | +/- Stdev |
| Latency | 263.25ms | 362.33ms | 2.00s | 86.15 |
| Req/Sec | 88.59 | 30.79 | 303 | 67.29 |

Figure - Table of Results For 400 Connections

Latency Distribution

50% 100.60ms

75% 114.26ms

90% 825.65ms

99% 1.57s

31668 requests in 30.07s, 6.19MB read

Socket errors: connect 0, read 0, write 0, timeout 681

Non-2xx or 3xx responses: 12

Requests/sec: 1053.30

Transfer/sec: 210.91KB

#### UCS C220-M3S

##### HTOP

##### Wrk Results

# 6 Evaluation and Analysis

## Raw Performance

### Sysbench

* Unsure why RPi 3 doing better than RPi 4

### Timed Linux Kernel Compile

The below calculations were used to work out the Standard deviation for these runs:

Diagram, schematic

Description automatically generated

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| σ2 = | |  | | --- | | Σ(xi - μ)2 | |  | | N | |

σ = Standard Deviation

#### RPi 4

|  |  |
| --- | --- |
| Run | Time To Compile (Rounded to nearest second) |
| 1 | 2665 |
| 2 | 2684 |
| 3 | 2681 |
| 4 | 2665 |
| 5 | 2682 |
| 6 | 2684 |
| 7 | 2626 |
| 8 | 2651 |
| 9 | 2651 |
| Average | 2665 |
| Standard Deviation | 17.8 |

Figure 60 - Table of Linux Kernel Compile Times on RPi 4



Figure - Boxplot Graph of RPi 4 Time to Compile Linux Kernel

#### RPi 3

|  |  |
| --- | --- |
| Run | Time To Compile |
| 1 | 11932 |
| 2 | 13656 |
| 3 | 13601 |
| 4 | 13268 |
| 5 | 13453 |
| 6 | 13502 |
| 7 | 13003 |
| 8 | 13560 |
| 9 | 13846 |
| Average | 13313 |
| Standard Deviation | 571 |

Figure 62 - Table of Linux Kernel Compile Times on RPi 3



Figure - Boxplot Graph of RPi 3 Time to Compile Linux Kernel

#### UCS C220-M3S

|  |  |
| --- | --- |
| Run | Time To Compile |
| 1 | 167 |
| 2 | 167 |
| 3 | 167 |
| 4 | 168 |
| 5 | 167 |
| 6 | 167 |
| 7 | 169 |
| 8 | 167 |
| 9 | 167 |
| Average | 167 |
| Standard Deviation | 0.71 |

Figure 64 - Table of Linux Kernel Compile Times on UCS C220-M3S



Figure 65 - Boxplot Graph of UCS C220-M3S Time to Compile Linux Kernel

#### Comparison Of Systems

Figure 66 - Bar Chart Highlighting Standard Deviation Between Systems

The above box plot charts for each system (Figure 50, Figure 52 and Figure 54) highlight the consistency of each system. The following points can be taken from the data found:

* UCS and RPi 4 perform more consistently
* RPi 3 has up to 10 mins of variation
* RPi 4 was the only system with no outlier

Although visually the RPi 3 looks more consistent in the box plot it was actually the least consistent in terms of variance with a standard deviation of

Figure 67 - Bar Chart Highlighting the Average Time to Compile Across Systems

For the average time to compile of each system it can clearly be seen from the above graph that going from left to right each system gets significantly faster than the last. The below conclusions can be drawn:

* RPi 4 takes less time to compile than RPi 3
* UCS takes less time to compile than RPi 4

On average the RPi 4 takes roughly 10600s less than the RPi 3. The difference in this raw processing power can be crucial to key business services. The faster time to compile of the RPi 4 along with the more consistent results for the time to compile show more consistent performance of the RPi 4 over the RPi 3. Further to this the UCS C220-M3S performed the compile almost 2500s faster on average, the UCS C220-M3S also performed almost 13200s faster than the RPi 3. The UCS C220-M3S when performing the kernel compile does however use 50kW to compile the Linux kernel where the RPi 4 uses 13.9kW to perform the same task. Even though the UCS C220-M3S is quicker it uses more energy to provide the same result.

## Service Testing

### DNS/Pi-Hole

#### Comparison of Time to Resolve

* Pi-Hole typically slower for non-cached entries
* For cached entries pi hole significantly quicker
* The above is due to it being a DNS forwarder not a dedicated DNS server do I need to test a DNS? Will that be helpful?

### DHCP

* DHCP uses 29.9% CPU (according to HTOP) when dhammer stress test is initiated on RPi 4

### AD

HOW AM I GONNA TEST THIS OMG

* Set up server
* Run 5-20VMs
* Grab HTOP output

### NAS

### Webserver

#### RPi 4

From the results gathered in the below Figures:

* Figure 54
* Figure 55
* Figure 56
* Figure 57

The above Figures show detailed results for how many requests were handled per second by the webserver and also metrics for the latency. It can be seen from these results that the RPi 4 can handle up to 100 concurrent requests with no errors, even jumping to 200 concurrent requests the RPi 4 drops 152 requests over a 30s period. Although this is not ideal that requests time out for a business webserver this will be suitable for most SMEs as the likelihood of them receiving over 100 concurrent requests will likely be small, however once they notice requests hitting over 100 concurrent this could then be migrated to the cloud or even a traditional server.

#### UCS C220-M3S

## Energy Usage of Systems

This section will take the figures for power draw found earlier and apply calculations to these to calculate the Cost/kWh usage of each system. To convert the Watt figures found earlier to kWh’s the below calculation was used:

### Energy Cost of Linux Kernel Compile

The below calculation is used to determine the total power draw for full duration of the test. This calculation can be used for both the RPi 4 and for the UCS C220-M3S.

|  |  |  |  |
| --- | --- | --- | --- |
| System | Average Time to Compile | Average Power Draw | Total Power Used (kWh) Rounded to 3sf |
| RPi 4 | 2665 | 5.2 | 0.00385 |
| UCS C220-M3S | 167 | 299 | 0.01387 |

Figure 68 - Table Comparing Average Time to Compile on Systems

Although the UCS C220-M3S compiles in less than 1/10th of the time taken by the RPi4, the RPi 4 is more energy efficient as it takes less kWh to achieve same result. This for services that the RPi 4 can run and perform on par with the

### Energy Cost of Idle Time Between Business Hours

The below calculation is used to determine the total power draw for the Idle hours between standard business operating hours. This calculation can be used for both the RPi 4 and for the UCS C220-M3S.

|  |  |  |  |
| --- | --- | --- | --- |
| System | Idle Power Draw (W) | Idle Time (Hours) | Total Power Used (kWh) |
| RPi 4 | 0.0021 | 16 | 0.0336 |
| UCS C220-M3S | 0.1150 | 16 | 1.8400 |

Figure 69 - Table Comparing Idle Power Draw of Systems

As shown in the above table RPi 4 uses less energy on idle between 5PM and 9AM the next day.

RPi 4

* Mon 9am – Fri 5PM = 0.1344kWh
* Weekend Fri 5PM – Mon 9am = 0.1344kWh
* Week with idle over weekend = 0.2688kWh
* Assuming power off over weekend = 0.1344kWh

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| RPi 4 | Idle Time Mon - Fri | Idle Time Weekend | No Weekend Power Down | Over 52 Week period | Weekend Power Down | Over 52 Week period |
| Energy kWh | 0.1344 | 0.1344 | 0.2688 | 13.9776 | 0.1344 | 6.9888 |

Figure - Table Outlining Energy Usage of RPi 4 in Idle Hours

UCS

* Mon 9am – Fri 5PM = 7.36kWh
* Weekend Fri 5PM – Mon 9am = 7.36kWh
* Week with idle over weekend = 14.72kWh
* Assuming power off over weekend = 7.36kWh

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| UCS C220-M3S | Idle Time Mon - Fri | Idle Time Weekend | No Weekend Power Down | Over 52 Week period | Weekend Power Down | Over 52 Week period |
| Energy kWh | 7.36 | 7.36 | 14.72 | 765.44 | 7.36 | 382.72 |

Figure - Table Outlining Energy Usage of UCS C220-M3S in Idle Hours

In total over the space of a 52-week year the business could be saving between 375 and 751 kWh by migrating their core services from a traditional server to the RPi 4.

### Energy Cost to Run Systems

The below calculation is used to determine the total power draw for full duration of business hours assuming a lower bound of 50% utilisation and 90% upper bound. This calculation can be used for both the RPi 4 and for the UCS C220-M3S. As this project only covers SME’s MNE’s will not be considered for this calculation as their power utilisation would be covered on a 24/5 or 24/7 basis dependant on the company.

|  |  |  |  |
| --- | --- | --- | --- |
| System | Power Draw (kWh) | Business Hours (Hours Per Week) | Total Power Used (kWh) |
| RPi 4 – 50% | 0.0026 | 40 | 0.1040 |
| RPi 4 – 90% | 0.0047 | 40 | 0.1880 |
| UCS C220-M3S – 50% | 0.0575 | 40 | 2.3000 |
| UCS C220-M3S – 90% | 0.1035 | 40 | 4.1400 |

Figure - Table Outlining Power Usage of Systems in a Working Week

## Cost To Run Analysis

To calculate the cost to run each of the systems is extremely situational and cannot be exactly determined as what may be the cost to run for one business could be significantly lower or higher dependant on the usage of each system.

With the current rise in cost of electricity in the UK this could then equate to £105 – 210 in savings to the business per year per server assuming a 28p/kWh electricity cost. This energy saving could buy 2-4 RPi’s or even be used to reinvest into other elements of the business.

# 7 Future Work

Testing of more tools

Deeper Analysis into the stability

Business use cases of home assistant is this a valid argument?

Research into the business cost associated with downtime

# 8 Conclusions

* RPi 4 uses significantly less power
* The potential applications are limitless

# 9 Reflection on Learning

* Became more familiar with the Linux OS

## Challenges Faced

* Blindly followed tutorials
  + Led to breaking the Pi OS
* Could do separate section on Challenges Faced

# References

Abu Sharkh, M., Jammal, M., Shami, A. & Ouda, A., 2013. *Resource allocation in a network-based cloud computing environment: design challenges,* s.l.: IEEE.

Acácio de Andrade, A., Batista Dietrich, Á., Francisco Blumetti Facó, J. & Reolon Jorge, R., 2020. *Low Cost Solution for Home Brewing and Small Brewing Business Using Raspberry Pi,* s.l.: Springer Nature.

Bargain Hardware, 2022. *Refurbished Servers, PCs, Workstations & Parts | Bargain.* [Online]   
Available at: https://www.bargainhardware.co.uk  
[Accessed 6 February 2022].

Bitwarden, Inc, 2022. *Install and Deploy - Linux | Bitwarden.* [Online]   
Available at: https://bitwarden.com/help/install-on-premise-linux/  
[Accessed 18 March 2022].

Cisco Systems Inc, 2017. *Cisco UCS C220 M3 Rack Server Data Sheet.* [Online]   
Available at: https://www.cisco.com/c/en/us/products/collateral/servers-unified-computing/ucs-c220-m3-rack-server/data\_sheet\_c78-700626.html  
[Accessed 11 February 2022].

Citrix, 2022. *What is VDI? Virtual Desktop Infrastructure.* [Online]   
Available at: https://www.citrix.com/en-gb/solutions/vdi-and-daas/what-is-vdi-virtual-desktop-infrastructure.html  
[Accessed 11 March 2022].

de la Cruz, J. E. C., Goyzueta, C. A. R. & Cahuana, C. D., 2016. *Intrusion Detection and Prevention System for Production Supervision in Small Businesses Based on Raspberry Pi and Snort,* s.l.: IEEE.

Debian Org, 2021. *Debian -- News -- Updated Debian 11: 11.2 released.* [Online]   
Available at: https://www.debian.org/News/2021/20211218  
[Accessed 15 March 2022].

Gibson Research Corporation, 2018. *GRC's | DNS Nameserver Performance Benchmark.* [Online]   
Available at: https://www.grc.com/dns/benchmark.htm  
[Accessed 10 April 2022].

Imperva, 2022. *What is a Honeypot | Honeynets, Spam Traps & more | Imperva.* [Online]   
Available at: https://www.imperva.com/learn/application-security/honeypot-honeynet/  
[Accessed 6 February 2022].

Intelligent Servers, 2022. *Intelligent Servers.* [Online]   
Available at: https://intelligentservers.co.uk  
[Accessed 1 February 2022].

ipchama, 2021. *ipchama/dhammer: DHCP stress tester and benchmark tool.* [Online]   
Available at: https://github.com/ipchama/dhammer  
[Accessed 9 April 2022].

IT in Stock, 2022. *Cisco UCS C220 M3 UCSC-C220-M3S 2x Quad Core E5-2643 3.30GHz.* [Online]   
Available at: https://www.itinstock.com/cisco-ucs-c220-m3-ucsc-c220-m3s-2x-quad-core-e5-2643-330ghz-600gb-24gb-server-48623-p.asp  
[Accessed 21 February 2022].

Jackson, C., Gooley, J., Iliesiu, A. & Malegaonkar, A., 2020. *Cisco Certified DevNet Associate DEVASC 200-901 Official Cert Guide.* s.l.:Cisco Press.

Maulana, H. & Al-Khowarizmi, 2021. *Analyze and Designing Low-Cost Network Monitoring System,* Medan, Indonesia: IOP.

Microsoft Azure, 2022. *Cloud Computing Services | Microsoft Azure.* [Online]   
Available at: https://azure.microsoft.com/en-us/  
[Accessed 2 February 2022].

Microsoft, 2022. *Active Directory Domain Services Overview.* [Online]   
Available at: https://docs.microsoft.com/en-us/windows-server/identity/ad-ds/get-started/virtual-dc/active-directory-domain-services-overview  
[Accessed 3 February 2022].

Miles, C., 2020. *GitHub - Akkadius/glass-isc-dhcp: Glass - ISC DHCP Server Interface.* [Online]   
Available at: https://github.com/Akkadius/glass-isc-dhcp  
[Accessed 4 February 2022].

Nguye, H.-Q., Loan, T. T. K., Mao, B. D. & Huh, E.-N., 2015. *Low Cost Real-Time System Monitoring,* s.l.: IEEE.

Open Benchmarking Org, 2022. *Timed Linux Kernel Compilation Benchmark - OpenBenchmarking.org.* [Online]   
Available at: https://openbenchmarking.org/test/pts/build-linux-kernel-1.13.0  
[Accessed 08 March 2022].

PassMark Software, 2022. *ARM Cortex A53 4 Core 1400 MHz vs ARM Cortex A72 4 Core 1500 MHz.* [Online]   
Available at: https://www.cpubenchmark.net/compare/ARM-Cortex-A53-4-Core-1400-MHz-vs-ARM-Cortex-A72-4-Core-1500-MHz/4143vs3917  
[Accessed 2 March 2022].

Passmark Software, 2022. *ARM Cortex-A72 4 Core 1800 MHz vs Intel Xeon E5-2643 @ 3.30GHz [cpubenchmark.net] by PassMark Software.* [Online]   
Available at: ARM Cortex-A72 4 Core 1800 MHz vs Intel Xeon E5-2643 @ 3.30GHz [cpubenchmark.net] by PassMark Software  
[Accessed 12 March 2022].

Pi-Hole, 2022. *Overview of Pi-hole - Pi-hole documentation.* [Online]   
Available at: https://docs.pi-hole.net/  
[Accessed 2 February 2022].

Portainer, 2022. *Container Management | Kubernetes GUI | Docker Swarm GUI | Portainer.* [Online]   
Available at: https://www.portainer.io/  
[Accessed 29 March 2022].

Portainer, 2022. *Install Portainer Agent with Docker on Linux - Portainer Documentation.* [Online]   
Available at: https://docs.portainer.io/v/ce-2.9/start/install/agent/docker/linux  
[Accessed 10 April 2022].

Portainer, 2022. *Install Portainer with Docker on Linux - Portainer Documentation.* [Online]   
Available at: https://docs.portainer.io/v/ce-2.9/start/install/server/docker/linux  
[Accessed 10 April 2022].

Raspberry Pi Foundation, 2022. *Operating system images - Raspberry Pi.* [Online]   
Available at: https://www.raspberrypi.com/software/operating-systems/  
[Accessed 15 March 2022].

Raspberry Pi Foundation, 2022. *Raspberry Pi.* [Online]   
Available at: https://www.raspberrypi.com  
[Accessed 1 February 2022].

Raspberry Pi Foundation, 2022. *Raspberry Pi 4 Datasheet.* [Online]   
Available at: https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-mechanical-drawing.pdf  
[Accessed 3 February 2022].

Raspberry Pi LTD, 2019. *raspberry-pi-4-datasheet.pdf.* [Online]   
Available at: https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-datasheet.pdf  
[Accessed 1 March 2022].

Raspberry Pi LTD, n.d. *raspberry-pi-3-b-plus-product-brief.* [Online]   
Available at: https://datasheets.raspberrypi.com/rpi3/raspberry-pi-3-b-plus-product-brief.pdf  
[Accessed 1 March 2022].

RedHat, 2020. *What is Infrastructure as Code (IaC).* [Online]   
Available at: https://www.redhat.com/en/topics/automation/what-is-infrastructure-as-code-iac  
[Accessed 5 March 2022].

Stratodesk, 2022. *Stratodesk NoTouch – Thin Client Software.* [Online]   
Available at: https://www.stratodesk.com/products/notouch/#1587497476663-a3983ba6-cdf3  
[Accessed 1 March 2022].

The Pi Hut, 2022. *The Ultimate Raspberry Pi & Maker Store.* [Online]   
Available at: https://www.thepihut.com  
[Accessed 1 February 2022].

Veritis, n.d. *AWS Vs Azure Vs GCP – The Cloud Platform of Your Choice?.* [Online]   
Available at: https://www.veritis.com/blog/aws-vs-azure-vs-gcp-the-cloud-platform-of-your-choice/  
[Accessed 13 February 2022].

VMWare, 2022. *What is a hypervisor?.* [Online]   
Available at: https://www.vmware.com/topics/glossary/content/hypervisor.html  
[Accessed 21 February 2022].

VMWare, 2022. *What is VDI? | Virtual Desktop Infrastructure | VMware Glossary.* [Online]   
Available at: https://www.vmware.com/topics/glossary/content/virtual-desktop-infrastructure-vdi.html  
[Accessed 11 March 2022].