UACM
Universidad Autónoma de la Ciudad de México
Nada humano me es ajeno

COLEGIO DE CIENCIA Y TECNOLOGÍA

LICENCIATURA EN INGENIERÍA DE SOFTWARE

Construcción y configuración de clúster con placas únicas de cómputo para sistemas paralelos/distribuidos

T E S I S

QUE PARA OPTAR POR EL TÍTULO DE

LICENCIADOS EN INGENIERÍA DE SOFTWARE

P R E S E N T A N :

CHRISTIAN JESÚS AGUIRRE CORTÉS

LUIS OMAR VÁZQUEZ VIDALES

D I R E C T O R: **M. EN C. ENRIQUE CRUZ MARTÍNEZ**

C O D I R E C T O R: **DR. VÍCTOR HUGO DE LA LUZ RODRÍGUEZ**

Ciudad de México, octubre de 2022.

UACM
Universidad Autónoma de la Ciudad de México
Nada humano me es ajeno

CA Coordinación Académica

SISTEMA BIBLIOTECARIO DE INFORMACIÓN Y DOCUMENTACIÓN



UNIVERSIDAD AUTÓNOMA DE LA CIUDAD DE MÉXICO
COORDINACIÓN ACADÉMICA

RESTRICCIONES DE USO PARA LAS TESIS DIGITALES

DERECHOS RESERVADOS©

La presente obra y cada uno de sus elementos está protegido por la Ley Federal del Derecho de Autor; por la Ley de la Universidad Autónoma de la Ciudad de México, así como lo dispuesto por el Estatuto General Orgánico de la Universidad Autónoma de la Ciudad de México; del mismo modo por lo establecido en el Acuerdo por el cual se aprueba la Norma mediante la que se Modifican, Adicionan y Derogan Diversas Disposiciones del Estatuto Orgánico de la Universidad de la Ciudad de México, aprobado por el Consejo de Gobierno el 29 de enero de 2002, con el objeto de definir las atribuciones de las diferentes unidades que forman la estructura de la Universidad Autónoma de la Ciudad de México como organismo público autónomo y lo establecido en el Reglamento de Titulación de la Universidad Autónoma de la Ciudad de México.

Por lo que el uso de su contenido, así como cada una de las partes que lo integran y que están bajo la tutela de la Ley Federal de Derecho de Autor, obliga a quien haga uso de la presente obra a considerar que solo lo realizará si es para fines educativos, académicos, de investigación o informativos y se compromete a citar esta fuente, así como a su autor ó autores. Por lo tanto, queda prohibida su reproducción total o parcial y cualquier uso diferente a los ya mencionados, los cuales serán reclamados por el titular de los derechos y sancionados conforme a la legislación aplicable.

Índice general

1. Introducción al clúster con placas SBC **1**
1.1. ¿Qué es un clúster? . . . 1
1.1.1. Tipos de clúster . . . 1
1.2. Historia del clúster . . . 2
1.3. ¿Qué es el procesamiento paralelo y para qué se utiliza? . . . 2
1.3.1. Raspberry Pi 3 B . . . 3
1.4. SISTEMAS OPERATIVOS . . . 5
1.4.1. Raspberry Pi OS . . . 5
1.4.2. Kali Linux . . . 6
1.4.3. RetroPie, para convertirlo en una consola de videojuegos . . . 6
1.4.4. Lakka, para jugar más ligero . . . 7
1.4.5. RISC OS Pi, para los menos expertos . . . 7
1.4.6. Eleccción del Sistema Operativo para las placas de nuestro clúster . . . 8

2. Armado y configuración de clúster con Arquitectura Maestro-Workers **9**
2.1. Hardware del clúster. . . . 10
2.1.1. Materiales para construcción del clúster . . . 10
2.1.2. Materiales Adicionales . . . 16
2.1.3. Costos . . . 18
2.2. Configuración General . . . 19
2.2.1. Arquitectura del clúster (almacenamiento y red). . . . 19
2.2.2. Instalación del Sistema Operativo Raspberry Pi OS (anteriormente llamado Raspbian) . . . 20
2.3. Configuración Inicial . . . 21
2.3.1. Conexión a la Red de internet . . . 22
2.3.2. Modificaciones desde el menú de configuración . . . 23
2.3.3. Configuración del nodo maestro . . . 24
2.3.4. Configuración de red . . . 24
2.3.5. Generación de claves . . . 25
2.3.6. DHCP (Protocolo de configuración dinámica de host) . . . 27
2.3.7. Cambiar el nombre del host . . . 28
2.3.8. NFS . . . 30
2.3.9. NAT . . . 30

3. Desarrollo e implementación de un software distribuido básico **31**
3.1. Diagramas de arquitectura de Software . . . 31
3.2. OpenMPI . . . 32
3.2.1. Script de instalación MPI . . . 33
3.3. Configuración de los nodos trabajadores . . . 34
3.4. NFS CLIENTE . . . 35
3.4.1. script para la configuración del NFS cliente . . . 35
3.4.2. PRUEBAS DE NFS . . . 37
3.4.3. HABILITAR EL DHCP EN LOS NODOS . . . 39
3.5. PRUEBAS EN EL CLUSTER . . . 41

3.6. PRUEBAS DE EJECUCIÓN, CÁLCULO DE PI . . . 42
3.6.1. Pasos complementarios de configuración . . . 44

4. Automatización y Monitoreo **47**
4.1. Automatización . . . 47
4.1.1. Manejo de varias sesiones SSH de forma concurrente: ClusterSSH (cssh) . . . 47
4.1.2. Habilitar SSH sin necesidad de teclado o monitor . . . 48
4.1.3. Configurar WIFI sin necesidad de teclado o monitor . . . 50
4.2. Monitoreo . . . 50
4.2.1. Glances como herramienta de monitoreo . . . 50
4.2.2. Instalación de Glances . . . 50
4.2.3. Uso de glances . . . 51
4.2.4. Creación de una página web para monitorear todos los nodos . . . 52
4.3. Pruebas para monitorear la temperatura de los nodos . . . 56

5. Dificultades y Conclusiones **59**
5.1. Dificultades. . . . 59
5.1.1. Alimentación de energía . . . 59
5.1.2. Sobrecalentamiento . . . 59
5.1.3. Directorios de DHCP . . . 59
5.1.4. Llaves SSH . . . 59
5.1.5. Programación paralela . . . 60
5.2. Conclusiones. . . . 60

A. Configuración de clúster con placas Rpi4 **61**
A.1. Scripts de automatización adicionales. . . . 62
A.1.1. Scripts para el nodo maestro . . . 62
A.1.2. Scripts para nodos secundarios . . . 65

B. Infraestructura física del clúster Rpi4 **67**

Índice de figuras

1.1. Esquema de clúster de CPU. . . . 1
1.2. Historia de Internet: Grandes Hitos. . . . 2
1.3. Captura de pantalla de SO Raspbian. . . . 5
1.4. Kali linux, escritorio por defecto. . . . 6
1.5. Retro Pie, escritorio por defecto. . . . 6
1.6. Lakka, escritorio por defecto. . . . 7
1.7. RISC OS Pi, escritorio por defecto. . . . 7

2.1. Diagrama de arquitectura de Hardware para el clúster de RPi. . . . 9
2.2. Raspberry Pi Modelo 3B. . . . 11
2.3. Pantalla de tv o monitor con entrada HDMI. . . . 11
2.4. Hub USB para la corriente eléctrica. . . . 12
2.5. Cables de Ethernet. . . . 12
2.6. Pinzas ponchadoras. . . . 13
2.7. Switch de Ethernet. . . . 13
2.8. Cables v8-usb. . . . 14
2.9. Memoria Micro SD 14
2.10. Computadora portátil (Laptop). . . . 15
2.11. Internet estable mínimo 6 mbps. . . . 15
2.12. Disco de estado sólido de 256 GB. . . . 16
2.13. Ventilador. . . . 16
2.14. Acrílico para apilar las SBC. . . . 17
2.15. Nobreak. . . . 17
2.16. Arquitectura del clúster. . . . 19
2.17. Versiones de Raspberry Pi OS. . . . 20
2.18. Versiones de Raspberry Pi OS. . . . 21
2.19. Pasos de configuración a Wi-Fi. . . . 22
2.20. Menú de configuración. . . . 23
2.21. Salida de la instrucción ssh-keygen -t rsa 26
2.22. Configuración de archivo hosts para establecer la IP fija de los nodos workers 27
2.23. Menú para cambiar el hostname 29

3.1. Diagrama de arquitectura de software del nodo maestro. . . . 31
3.2. Cables v8-usb. . . . 32
3.3. ompi_info 34
3.4. prueba nfs parte1. . . . 37
3.5. prueba nfs parte2. . . . 38
3.6. Instrucción para copiar las llaves a los nodos. . . . 40
3.7. Menú para configurar lenguaje. . . . 44
3.9. Menú para escoger el lenguaje por defecto a español UTF-8. . . . 45
3.8. Menú Change Locale. . . . 45
3.10. Menú para cambiar la zona horaria. . . . 45

4.1. Uso de CSSH. . . . 48
4.2. Uso de CSSH con grupos. . . . 49

4.3. glances-w con cssh. . . . 53
4.4. Dashboard de Monitoreo para todos los nodos del clúster. . . . 54
4.5. Métricas de monitoreo. . . . 54
4.6. Comparativa de temperatura del clúster . . . 57

A.1. Win32 Disk Imager Read. . . . 62
A.2. Win32 Disk Imager Write. . . . 62

B.1. Accesorios de Rpi4. . . . 67
B.2. Tarjeta Rpi4 con ventilador incluido. . . . 68
B.3. Memoria micro SD 32GB. . . . 68
B.4. Cable de corriente con interruptor. . . . 69
B.5. Cable HDMI. . . . 69
B.6. Adaptador mini USB a HDMI. . . . 70
B.7. Carcasas Rpi4 con ventilador y disipadores de calor. . . . 70
B.8. Rpi4. . . . 71
B.9. Clúster rpi3 + rpi4, topografía 1. . . . 72
B.10. Clúster rpi3 + rpi4, topografía 3. . . . 72

Índice de tablas

1.1. Comparativa de las características principales de los modelos de Raspberry Pi. 4

2.1. Comparativa de costos. 18

Resumen

El presente trabajo expone un poco de la historia del clúster y definición del clúster. Se presentan las características de la Raspberry Pi 3B y se compara con otras Placas Únicas de Cómputo (Single Board Computer o SBC por sus siglas en inglés). Las SBC son placas que contienen todos o la mayor parte de los componentes de una computadora en reducidas dimensiones. Brevemente se presentan las características de diferentes sistemas operativos aptos para la Raspberry Pi. También, se describen las piezas de hardware que se requieren para el proceso de montaje de nuestro clúster.

Con respecto al software se describen los pasos de descarga e instalación del sistema operativo en las tarjetas microSD, se eligen los protocolos de comunicación, la configuración inicial del sistema operativo desde la actualización de paquetes y otras configuraciones complementarios como cambiar el idioma, hasta la configuración de red (DHCP), la configuración de un NFS (network file system), y el software requerido para la ejecución de sistemas de procesamiento paralelo.

Se explica el porqué se eligió este sistema operativo sobre otros, así como la justificación de la selección de la Raspberry Pi 3B para este trabajo de tesis. Se ejecuta un sistema de procesamiento paralelo y se implementa una estación de monitoreo vía web en lenguaje html y se integra al sistema de supervisión distribuida para redes, adicionalmente, se realizan diagramas de arquitectura y scripts de automatización.

Se incluye una cotización de costos para dar una idea de cuánto puede costar aproximadamente en un clúster con características similares o mejores que el se expone en este trabajo.

Se realizan pruebas básicas y ejecuciones de un sistema paralelo como el calculo del número “pi”.

Esta tesis presenta el diseño, montaje y configuración de un clúster de bajo costo, escalable, el cual tiene por objeto estimular la enseñanza de cómputo en paralelo y la interacción entre hardware y software para mostrar cómo montar un clúster a gran escala en el futuro. Para ello, no solo se ha prestado especial atención al hardware, sino que también al software específico para el entorno en el que será ejecutado, despachado y monitoreado (middleware).

Introducción

El área de la supercomputación sigue avanzando año tras año y según la célebre lista TOP500 [top] son cada vez más los sistemas que mejoran en potencia y eficiencia. Hasta el momento el primer lugar en High-Performance Conjugate Gradient (HPCG) lo tiene Estados Unidos con su supercomputador Frontier. Por definición, un clúster de computadoras es un conjunto de máquinas que trabajan juntas y que en determinados aspectos podría contemplarse como un único sistema. Sin embargo, al hablar de clústeres es fácil imaginar gigantescas instalaciones con grandes servidores, que además de ocupar un enorme espacio también tienen un consumo de energía sumamente elevado.

También existen los clústers en la nube donde los famosos proveedores son GCS, AWS, Azure, entre otros; que es más caro, pero tiene sus ventajas en comparación de un clúster físico. Por ejemplo, ya no hay que preocuparse de comprar las piezas que se dañen y de instalar las piezas manualmente, cuando es en la nube la mayoría de los nodos se construyen y configuran de manera automática o semiautomática. Sin embargo, también de alguna forma se pueden automatizar varios procesos en un clúster con Raspberry Pi; algunos procesos automatizados también se exponen en este trabajo.

La rápida evolución en los sistemas informáticos, las redes de comunicaciones y la aparición de computadoras del tamaño de una tarjeta de crédito han permitido que el concepto de clúster se pueda aplicar a proyectos más modestos. Por ello, hemos decidido crear nuestro propio clúster con placas únicas de cómputo.

Existen grandes empresas o instituciones que han creado grandes clústers con SBC, por ejemplo, el clúster de Oracle con 1,060 Raspberry Pi. También está el caso del Laboratorio Nacional de Los Alamos (EEUU) con su supercomputadora de 750 Raspberry Pi o sus 3000 núcleos de procesamiento trabajando en conjunto; el sistema brinda a los desarrolladores tiempo exclusivo en una plataforma económica, pero altamente paralelizada para la prueba y validación de tecnologías de software de sistemas escalables.

La ventaja de estos clústers con Raspberry Pi es que son pequeños, baratos, consumen muy poca energía, reducen tiempos para problemas de procesamiento paralelo y se pueden escalar fácilmente, es por ello que lo utilizan en muchas áreas; tal como estaciones meteorológicas, en agricultura, en big data, seguridad, en lo académico y en muchas otras áreas. Además de que una tendencia en los últimos años es usar procesadores tipo ARM que son usados en teléfonos celulares y placas tipo SBC.

Capítulo 1

Introducción al clúster con placas SBC

1.1. ¿Qué es un clúster?

Una breve definición de clúster sería: un conjunto de computadoras independientes, interconectadas entre sí, de tal manera que funcionan como un solo recurso computacional.
En el siguiente diagrama se puede visualizar la arquitectura básica de un clúster.

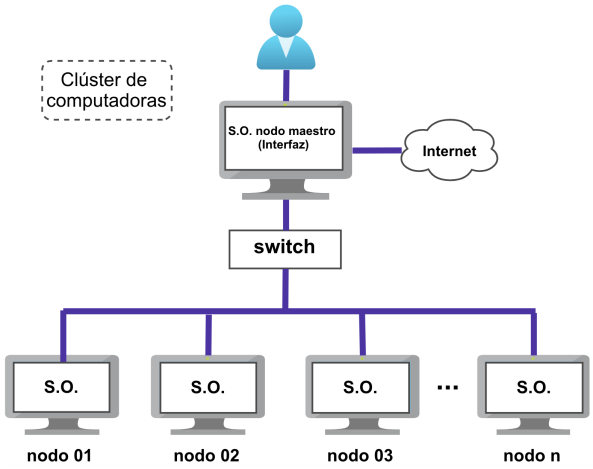


Figura 1.1: Esquema de clúster de CPU.

1.1.1. Tipos de clúster

Existen diferentes tipos de clúster y cada uno tiene diferentes propósitos, los principales tipos de clúster son:

- Clúster de Alta Disponibilidad, (Failover o High Availability Computing Clúster).
- Cluster de Balanceo de Carga (Load Balancing).
- Cluster de Alto Rendimiento (High Performance Computing Cluster).
- Procesamiento paralelo.

Clúster de Alta Disponibilidad:
Garantiza que un servicio permanezca siempre activo. Es por eso que, en caso de que un nodo falle, ya sea por fallas de red, hardware o software, otro nodo mantiene el servicio en funcionamiento.

Clúster de Balanceo de Carga:
Es una estructura en la que todos los nodos son responsables de realizar una determinada tarea. De

este modo, si uno de los nodos presenta un problema, se elimina automáticamente del clúster y la función inicial que le fue asignada se comparte entre los demás nodos.

Clúster de Alto Rendimiento:
Este tipo de clúster se utiliza para realizar tareas de alto rendimiento, con el fin de garantizar al máximo sus altas prestaciones.

Procesamiento paralelo:
Este tipo de clúster transforma una tarea compleja en varias tareas sencillas y las distribuye entre los nodos integrados al clúster.

1.2. Historia del clúster

El primer clúster de la historia fue ARCNET (Attached Resource Computer Network o Red informática de recursos adjuntos), fue desarrollado por el ingeniero John Murphy en Datapoint Corporation en 1976 bajo la dirección de Víctor Poor y anunciado en 1977, con una arquitectura de red de tipo LAN.

ARCNET utiliza cableado coaxial RG / 62 de 93 ohmios, cableado de par trenzado RS485 o cableado de fibra óptica para transmitir datos a 2,5 Mbps y un máximo de 255 nodos. Fue desarrollado originalmente para conectar grupos de sus terminales Datapoint 2200.

Para conectar los nodos en el primer clúster de la historia se utilizó el Adaptador ARCNET, también conocido como tarjeta de interfaz de red o NIC, fue un componente del hardware que sirve para conectar un ordenador a una red informática, habilitando el intercambio de datos y recursos entre éste y otros dispositivos, es decir, una red.
Mediante la implementación de los circuitos electrónicos requeridos, la NIC permitió la comunicación de una red de computadoras mediante cables (actualmente ya se puede hacer mediante otra tecnología).
El primer clúster de la historia nace en el año 1977 públicamente, aunque generalmente se cree que el nacimiento de la red Internet fue primero con el proyecto ARPANET [his]. ARPANET fue creado en el año 1969 y en efecto, fue casi una década antes de ARCNET (el primer clúster), pero ARPANET aún no era pública y únicamente se utilizaba para conectar algunas universidades. Fue hasta el año 1983 que se hizo pública con la adopción de los protocolos TCP/IP, fue entonces que surge Internet en el año 1983 públicamente, entonces, ARCNET fue primero que internet, como se puede ver en la Figura 1.2.

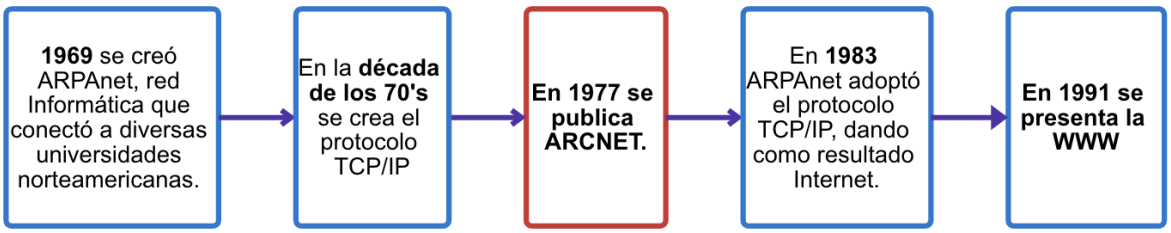


Figura 1.2: Historia de Internet: Grandes Hitos.

La base formal de la ingeniería informática cree que la metodología de proveer un medio de hacer trabajos paralelos de cualquier tipo fue inventado posiblemente por Gene Amdahl de IBM, que en 1967 publicó lo que ha llegado a ser considerado como el "libro blanco" inicial de procesamiento paralelo: la ley de Amdahl que describe matemáticamente el aceleramiento que se puede esperar paralelizando cualquier otra serie de tareas realizadas en una arquitectura paralela.

1.3. ¿Qué es el procesamiento paralelo y para qué se utiliza?

Si nos transportamos un par de décadas atrás, una computadora tradicional contaba con una unidad de procesamiento individual para ejecutar las instrucciones especificadas en un programa. Los problemas eran divididos en series discretas de instrucciones donde eran ejecutadas una a la vez, es decir, unas tras otras. Una manera de incrementar el poder de cómputo es usar más de una unidad de procesamiento

dentro de una única computadora o bien, utilizar varias computadoras, trabajando todas juntas sobre el mismo problema. El problema está dividido en partes, donde cada una de ellas es resuelta por una unidad de procesamiento individual de manera paralela. Se puede definir al procesamiento paralelo como el uso simultáneo de múltiples recursos computacionales para resolver un problema, [Wil06].

La necesidad de la computación paralela se origina por las limitaciones de las computadoras secuenciales, integrando varios procesadores para llevar a cabo la computación es posible resolver problemas que requieren de más memoria o de mayor velocidad de cómputo. El objetivo principal de la computación paralela es, reducir el tiempo de resolución de problemas computacionales, o bien, para resolver problemas más grandes que no podrían ser resueltos por un computador convencional. Para llevar a cabo esto, es necesario emplear sistemas de cómputo de alto desempeño y algoritmos paralelos que utilicen estos sistemas eficientemente. Los problemas habituales en los cuales se aplica la programación paralela son:

- Problemas con alta demanda de cómputo.
- Problemas que requieren procesar una gran cantidad de datos.
- Problemas de tiempo real en los que se necesita la respuesta en un tiempo mínimo.

De esta forma, la comunidad científica usa la computación paralela para resolver problemas que sin el paralelismo serían intratables, o con tiempos de respuesta inaceptables. Algunos campos que se favorecen de la programación paralela son: predicciones y estudios meteorológicos, estudio del genoma humano, modelado de la biosfera, predicciones sísmicas, simulación de moléculas, modelización y simulación financiera, realidad virtual, motores de búsqueda web, exploración de hidrocarburos, diseño de fármacos, entre otros. Aunque tradicionalmente el objetivo primario de HPC (Computación de Alto Rendimiento o High-Performance Computing) fue reducir el tiempo de ejecución de las aplicaciones; en las últimas décadas la eficiencia energética ha cobrado un valor semejante. Esto se debe al elevado consumo energético y generación de calor de las arquitecturas paralelas que afectan al funcionamiento y repercuten en el costo, debido a la adquisición de equipos para refrigeración y al consumo energético que generan los mismos.

1.3.1. Raspberry Pi 3 B

En esta sección se describe la arquitectura de la RPi 3 modelo B, dado que fue la elegida para el despliegue del clúster correspondiente a esta tesis.

La RPi 3 está conformada por un SoC (System on a Chip) especialmente desarrollado para este modelo: el Broadcom BCM2837. Este SoC incluye 4 núcleos de procesamiento ARM Cortex-A53 de alto rendimiento que funcionan a 1.2GHz con 32kb de caché de nivel 1 y 512kb de nivel 2, sumado a un procesador gráfico VideoCore IV que se encuentra conectado a un módulo de memoria LPDDR2 de 1 GB en la parte posterior de la placa. Este modelo comparte el mismo chip SMSC LAN9514 que su predecesor la RPi 2, agregando conectividad Ethernet 10/100 y cuatro canales USB a la placa. El chip SMSC se conecta al SoC a través de un único canal USB, que funciona como un adaptador USB-a-Ethernet y un concentrador USB. También se puede encontrar el chip Broadcom BCM43438 que proporciona conectividad LAN inalámbrica de 2.4GHz 802.11n, y Bluetooth 4.1 de bajo consumo. Es importante destacar que no es necesario conectar una antena externa a la RPi 3, dado que las señales inalámbricas son emitidas y recibidas por una antena tipo chip soldada directamente a la placa, a fin de mantener el tamaño del dispositivo al mínimo. A pesar de su tamaño, esta antena presenta excelente rendimiento. De igual manera se mostrará en el Cuadro 1.1, una comparativa de diferentes Raspberrys y sus características [Esp].

Lista de Raspberry Pi en el mercado							
MODELO	RPi B+	RPi 2 B	RPi 3 B	RPi Zero	RPi Zero W	RPi 3 B+	RPi 4
Fecha de lanzamiento	Feb 2012	Feb 2015	Feb 2016	Nov 2015	Feb 2017	Mar 2017	Jun 2019
Precio (US$)	25	35	35	5	10	47	48
SoC	Broadcom BCM2835	Broadcom BCM2836	Broadcom BCM2837	Broadcom BCM2835	Broadcom BCM2835	Broadcom BCM2837B0	Broadcom BCM2711B0
Procesador	ARM1176JZF-S	Cortex-A7	Cortex-A53 (ARMv8) 64-bit	ARM1176JZF-S	ARM1176JZF-S	Cortex-A53 (ARMv8) 64-bit	quad-core Cortex-A72
Nº Cores o Núcleos	1	4	4	1	1	4	4
GPU	VideoCore IV 400 MHz	VideoCore IV 400 MHz	VideoCore IV 400 MHz	VideoCore IV 400 MHz	VideoCore IV 400 MHz	VideoCore IV 400 MHz	VideoCore VI 500 MHz
CPU Clock	700 MHz	900 MHz	1.2 GHz	1 GHz	1 GHz	1.4 GHz	1.5 GHz
RAM	512 MB	1 GB	1 GB	512 MB	512 MB	1 GB	1/2/4 GB
Memoria	Micro SD	Micro SD	Micro SD	Micro SD	Micro SD	Micro SD	Micro SD
USB 2.0	2	4	4	1 microUSB	1 microUSB	2	2
USB 3.0	0	0	0	0	0	0	2
Ethernet	Sí (10/100)	Sí (10/100)	Sí, Fast Ethernet (10/100) Gbps	No	No	Sí, Gigabit Ethernet over USB 2.0 (300 Mbps de máximo teórico)	Sí, Gigabit Ethernet
Wi-Fi	No	No	Si	No	Si	Si 2,4GHz / 5GHz	Si 2,4GHz / 5GHz
Bluetooth	No	No	Si	No	Si	Bluetooth 4.2	Bluetooth 5.0
HDMI	Si	Si	Si	Mini	Mini	Si	Si, x2
GPIO	8	17	17	17	17	40 pines	40 pines
DSI (LCD)	Si	Si	Si	No	No	Si	Si
Cámara	Si	Si	Si	Si	Si	Si	Si
Altura	85.6mm	85.6mm	85.6mm	65mm	65mm	82mm	82mm
Ancho	53.98mm	56.5mm	56.5mm	30mm	30mm	56mm	56mm
Profundidad	17 mm	17 mm	17 mm	5 mm	5 mm	19,5 mm	19,5 mm
Peso	45 g	45 g	45 g	9 g	9 g	50 g	50 g
Consumo	330 mA, 5V	350 mA, 5V	400 mA, 5V	100 mA, 5V	150 mA, 5V	500mA, 5V	600mA, 5V

Tabla 1.1: Comparativa de las características principales de los modelos de Raspberry Pi.

1.4. SISTEMAS OPERATIVOS

Como ya se ha mencionado, las posibilidades que nos ofrecen las SBC, específicamente la Raspberry Pi son muchas y muy variadas. Por ello, es importante que el sistema operativo que lleve sea el más adecuado a nuestras necesidades. A continuación mostraremos algunos de los más destacados que hay en la actualidad [AGSS19].

1.4.1. Raspberry Pi OS

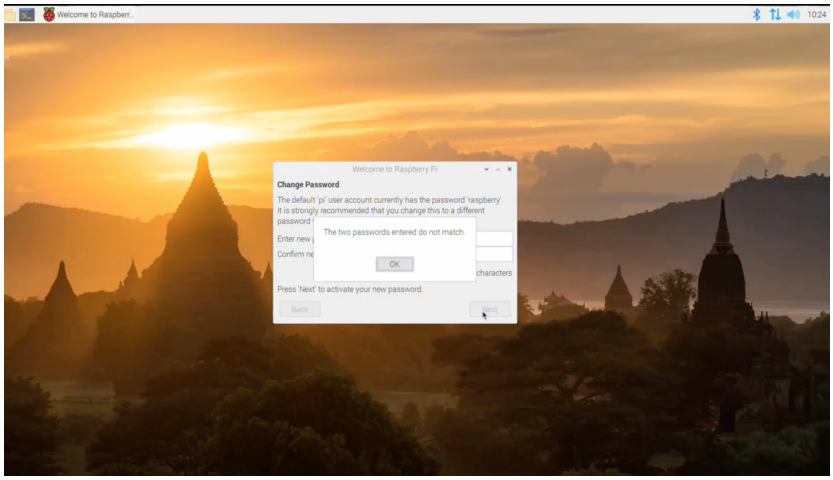


Figura 1.3: Captura de pantalla de SO Raspbian.

Raspberry Pi OS antes conocido como Raspbian, es el sistema operativo oficial para las placas Raspberry Pi.
Para descargarlo podemos encontrar el software disponible en la página oficial de Raspberry Pi: https://www.raspberrypi.com/software/

Existen tres versiones diferentes:

1. **Completa (Raspberry Pi OS with desktop and recommended software):** Esta versión viene con todo lo necesario para comenzar a trabajar sin ningún tipo de problema, como si fuese un ordenador normal.
2. **Standard:** En este caso solamente viene lo básico y no tenemos ningún tipo de programa preinstalado, excepto lo esencial.
3. **Lite (Raspberry Pi OS Lite):** Solamente ocupa 400 MB, eso significa que únicamente tiene lo necesario para arrancar. A partir de ahí, cada uno deberá instalarle aquello que le sea más útil.

Una de las características más interesantes de este sistema operativo es que es más fácil de usar, por la simpleza de su interfaz y que nos da la opción de optimizar los recursos de las placas descargando la versión lite y aprovechándolos para mejorar el poder de procesamiento.

1.4.2. Kali Linux

Es un sistema operativo orientado hacia la seguridad en las redes, no en vano tiene más de 600 herramientas orientadas a tal fin, como un escáner de redes Nmap o un programa de criptografía, su pantalla de escritorio estándar es como se visualiza en la Figura 1.4.

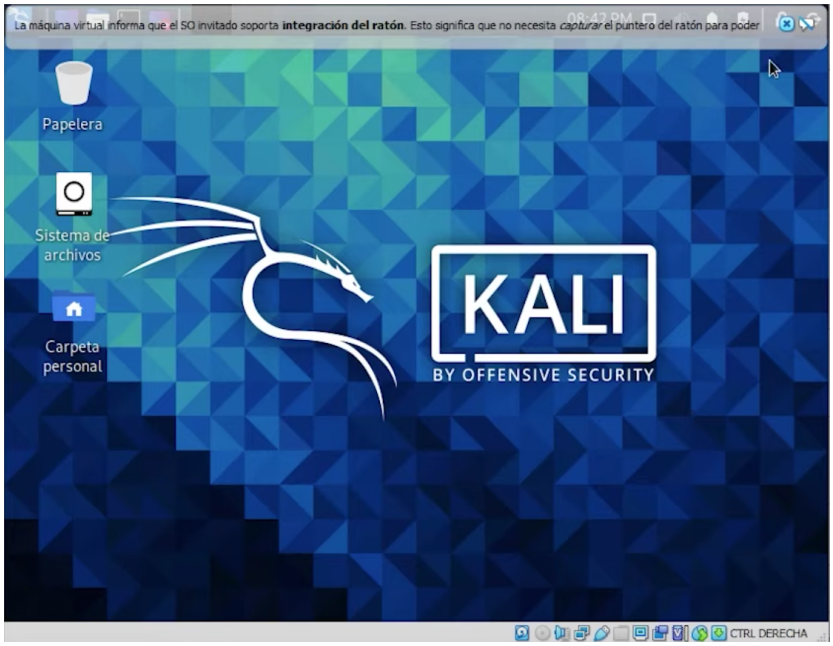


Figura 1.4: Kali linux, escritorio por defecto.

Las actualizaciones de software y seguridad son bastante habituales, el diseño y el propio funcionamiento es sencillo y un tanto austero, puesto que todo el potencial está en la seguridad que aporta al usuario. Tiene una comunidad muy amplia que usa esta versión, lo que le convierte en una posibilidad realmente buena.

1.4.3. RetroPie, para convertirlo en una consola de videojuegos

Gracias a RetroPie convertiremos nuestro Raspberry Pi en una consola retro. Realmente lo que estamos ejecutando es Raspberry Pi OS, pero con la apariencia que le han dado sus creadores. RetroPie ejecuta Emulation Station por lo que podremos usar nuestro dispositivo como si fuera una consola, viendo los juegos con suficiente calidad como para disfrutar de ellos; su pantalla de escritorio estándar es como se visualiza en la Figura 1.5.



Figura 1.5: Retro Pie, escritorio por defecto.

1.4.4. Lakka, para jugar más ligero

Lakka también es un sistema operativo para jugar, pero se diferencia fundamentalmente en que es muy ligero en su funcionamiento y sencillo en su interfaz. El diseño de Lakka se puede asemejar al de la PlayStation, puesto que la estructura, el colorido y el funcionamiento es el mismo que tiene la consola de Sony. Su pantalla de escritorio estándar es como se visualiza en la Figura 1.6.

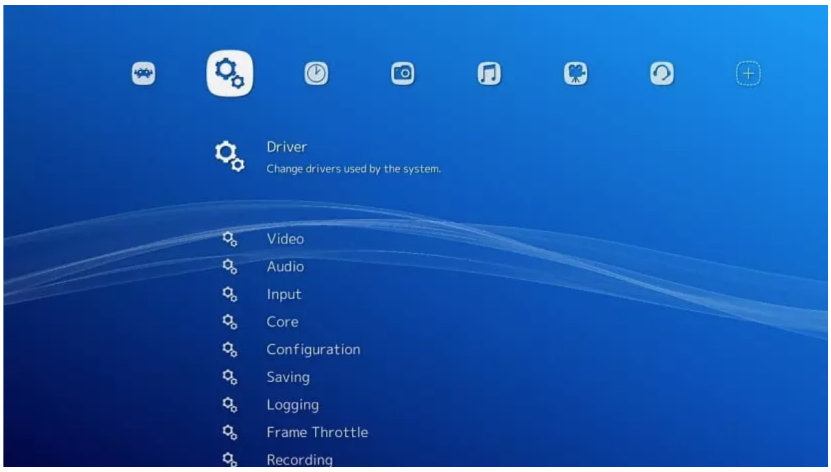


Figura 1.6: Lakka, escritorio por defecto.

Este es uno de los mejores sistemas operativos que podemos instalar en la Raspberry Pi si lo que queremos es utilizarla como una consola de video juegos retro, puesto que el buen funcionamiento se combina con una interfaz que nos resulta muy familiar.

1.4.5. RISC OS Pi, para los menos expertos

Este es un sistema operativo pensado para aquellos usuarios que no son muy expertos en Raspberry Pi. Se abren aplicaciones una a una, aunque parezca una limitación, es algo que ayuda a mucha gente que comienza en este mundo. Su pantalla de escritorio estándar es como se visualiza en la Figura 1.7.

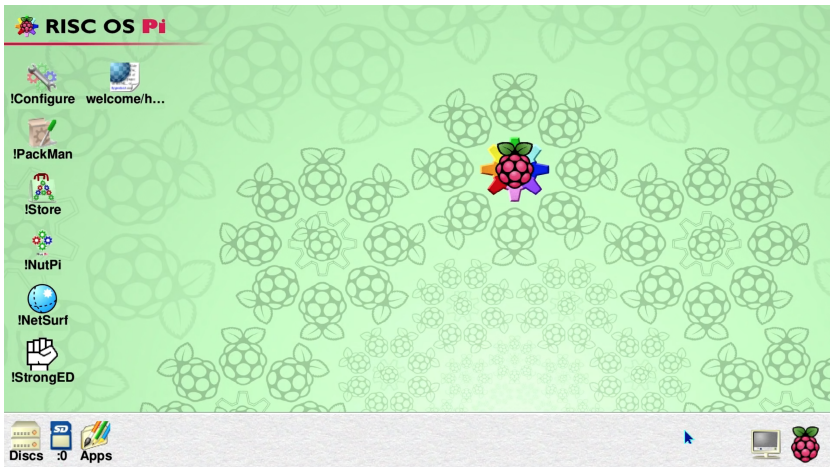


Figura 1.7: RISC OS Pi, escritorio por defecto.

Es el sistema operativo más conveniente si lo que pretendemos es aprender a funcionar con este dispositivo. Además de ser muy liviano (ocupa solo 16 MB de memoria), su nivel es el ideal para usuarios principiantes.

1.4.6. Eleccción del Sistema Operativo para las placas de nuestro clúster

Elegimos el sistema operativo **Raspberry Pi OS** por que además de ser el oficial, es una de las mejores distribuciones de uso general para el Raspberry Pi que podemos encontrar [Pon16].
Algunas de las principales características convincentes en su utilización son:

- Raspberry Pi OS es una distribución Linux basada en Debian
- La principal diferencia de Raspberry Pi OS respecto a otras distribuciones Linux es que está **optimizada para funcionar en procesadores ARM**, concretamente en el del Raspberry Pi, no generando fallos ni errores y pudiendo sacar todo el potencial a estas CPUs.
- Otorga la posibilidad de elegir entre las tres diferentes versiones que existen y se adecuan a nuestras necesidades de optimizar los recursos en los nodos trabajadores, instalándoles la versión LITE, e instalando la versión estándar o de escritorio al nodo maestro para tener una interfaz más gráfica y fácil de usar.

Capítulo 2

Armado y configuración de clúster con Arquitectura Maestro-Workers

En este capítulo se explican los materiales requeridos y configuraciones que se deben llevar a cabo para el armado y configuración inicial del clúster con placas únicas de cómputo, específicamente con el modelo Raspberry Pi 3 B.

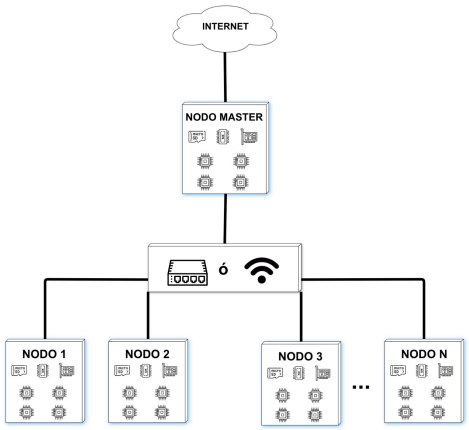


Figura 2.1: Diagrama de arquitectura de Hardware para el clúster de RPi.

En el diagrama anterior (Figura 2.1) se muestra el hardware esencial que utilizamos para nuestro clúster. Como podemos observar tenemos dos opciones para conectar nuestros nodos a internet y que se puedan comunicar entre ellos: Conexión por cables de ethernet, utilizando un switch de 8 puertos y conectándolo a nuestro modem y la conexión inalámbrica por medio de Wi-Fi conectándo nuestros nodos, utilizando el nombre de nuestra red Wi-Fi y la contraseña [Mon16].

También podemos ver las principales características del hardware de nuestras Placas Raspberry Pi 3B, así como el tipo de almacenamiento que utilizan, algunas de ellas son:

- Almacenamiento: **Memoria Micro SD de mínimo 8GB.**
- Memoria RAM: **1GB LPDDR2 SDRAM.**
- Procesador: **ARM Cortex-A53 (64Bit) a 1.4GHz Quad-Core (4 núcleos).**
- Conectividad:
  - Wi-fi: **802.11 b/g/n Wireless LAN and Bluetooth 4.1**
  - Ethernet: **Gigabit Ethernet sobre USB 2.0 (rendimiento máximo de 300 Mbps)**

2.1. Hardware del clúster.

La mayoría de personas piensan, que para construir un clúster se requiere el uso de hardware comercial muy costoso y de gran tamaño, como computadoras de escritorio, servidores, o la implementación de configuraciones complejas en máquinas virtuales. Debido a las razones anteriores, no era muy usual el hecho de que alguien ensamblara y configurara un clúster, pero gracias al bajo costo y al reducido tamaño de las placas SBC, en éste caso, la Raspberry Pi (RPi), construir un clúster para explorar el mundo de la computación paralela hace que esto sea accesible y fácil de llevar a cabo.

Si bien, las RPi no son una opción posible para un sistema complejo de producción, resulta interesante el conjunto de herramientas que brindan para aprender las tecnologías que conforman los clústers profesionales. Éstas permiten conocer, instalar y configurar diferentes herramientas como: sistemas de cómputo paralelo y distribuido, monitores de procesos, balanceadores de carga, compiladores u otros proyectos de vanguardia que en su mayoría son de código abierto [Ste02].

Un clúster está formado por un conjunto de dispositivos conectados entre sí para que se puedan comunicar. Para realizar el despliegue del mismo, se requieren una serie de elementos, principalmente de los nodos que son los dispositivos encargados de realizar el cómputo y el control; un dispositivo que brinde un medio de comunicación que permita interconectar los mismos.
Para que los dispositivos se comuniquen entre sí se debe configurar el hardware de red y realizar las conexiones de las placas al mismo, por medio de cables o conexiones inalámbricas. También se necesita contar con hardware complementario el cual facilita la tarea de configuración de los dispositivos. A continuación se proporciona un listado de los elementos que se necesitan para realizar el armado del clúster y que posteriormente se describirán más a detalle:

- Cinco RPi 3 Modelo B.
- Una TV o Monitor con entrada HDMI
- Un teclado USB.
- Un HUB usb que permita conectar al menos 5 dispositivos USB a la corriente eléctrica o bien una fuente de PC de al menos 250W.
- Cinco cables Ethernet/RJ45.
- Un Switch de ethernet con al menos 6 puertos.
- Cinco cables tipo V8-usb (mini USB).
- Cinco tarjetas Micro SD Kingston de mínimo 8Gb (Es preferible que sean de mayor capacidad).
- Una Laptop o PC de escritorio para realizar las conexiones remotas.
- Una conexión de Internet.

2.1.1. Materiales para construcción del clúster

Lo más importante y de mayor valor económico fueron las tarjetas Raspberry Pi (también conocidas como placas RPi), que fueron proporcionadas por el laboratorio LACECI (Laboratorio de Cómputo para la Enseñanza de la Ciencia) ubicado en el plantel San Lorenzo Tezonco de la Universidad Autónoma de la Ciudad de México (UACM); el resto de los materiales fueron comprados con nuestros propios recursos. Sin embargo, al finalizar el proyecto, éste quedará a disposición del Laboratorio LACECI para su modificación o estudio.

Placas Únicas de Cómputo (SBC por sus siglas en inglés)

Para armar nuestro clúster de Placas Únicas de Cómputo elegimos las tarjetas Raspberry Pi 3 Modelo B.
Entre las características más destacadas podemos remarcar su tamaño reducido y bajo costo, lo cual hace que sea muy apropiada para el diseño de un clúster con fines de experimentación, Figura 2.2.

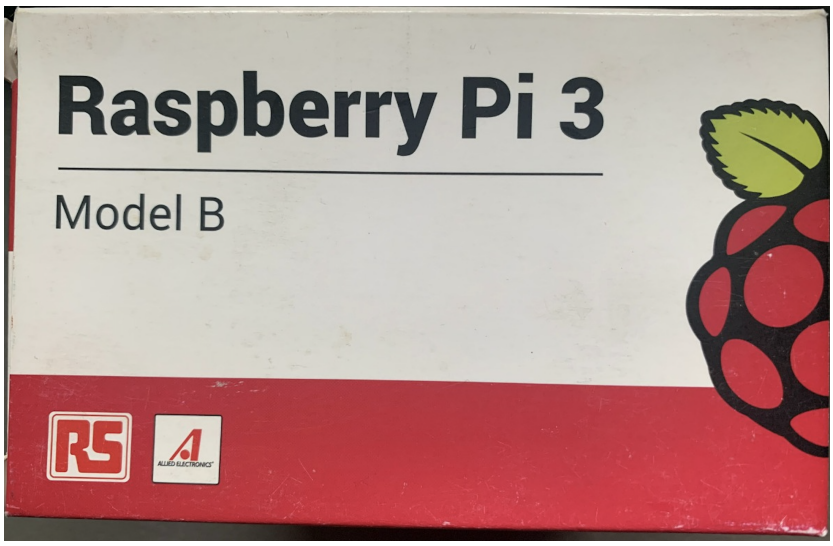


Figura 2.2: Raspberry Pi Modelo 3B.

TV o Monitor con interfaz HDMI

Utilizamos un monitor con entrada HDMI, (Figura 2.3) y un cable HDMI para conectarlo al nodo maestro y así poder visualizar las instrucciones a ejecutar, configurar y administrar los recursos del clúster.



Figura 2.3: Pantalla de tv o monitor con entrada HDMI.

HUB USB para la corriente eléctrica.

Se necesitan adaptadores de corriente para cada cable que alimenta cada RPi 3, (para que estén conectadas a la corriente eléctrica), (Figura 2.4), No obstante, existen hubs de alimentación USB que serían más prácticos en lugar de que cada cable v8-USB tenga su propio adaptador.

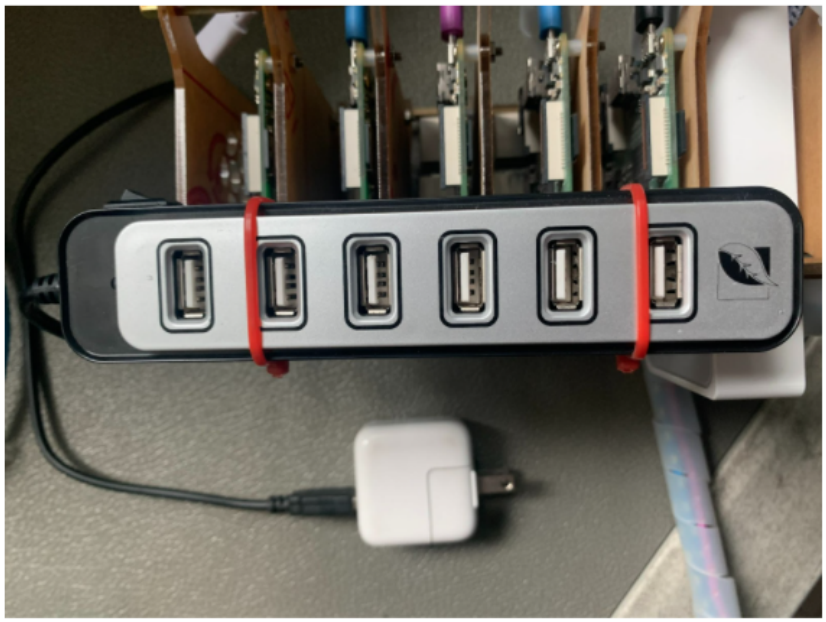


Figura 2.4: Hub USB para la corriente eléctrica.

Debemos tener cuidado al adquirir el hub USB, ya que algunos reducen el voltaje o no soportan tener muchos dispositivos conectados, en esos casos pueden dejar sin alimentación algunas Raspberry Pi. Para más detalles de estos problemas en la sección de **Dificultades del capítulo 1.**

Cables de Ethernet/RJ45.

Las RPi 3 B cuentan con antenas Wi-Fi así que tenemos la opción de conectarlas a la red inalámbricamente o por medio de cables de ethernet. Lo más recomendable es configurar el clúster a una red de internet alámbrica, (Figura 2.5), las ventajas de esto entre otras cosas, es que baja la pérdida de paquetes, tiene menor latencia, mayor estabilidad, y por último mayor velocidad y eficiencia.



Figura 2.5: Cables de Ethernet.

La RPi 3 cuenta con un puerto RJ45 para conexiones Ethernet estos cables de Ethernet se recomienda que sean armados por uno mismo, para tener mejor organización en el cableado ya que en muchos establecimientos es difícil encontrar las medidas necesarias de cable. Basta con comprar el cable RJ45 de Ethernet, cortarlo a la medida que necesitemos y colocar los plugs de acuerdo con la combinación de colores estándar de red. La herramienta que se necesita es unas pinzas ponchadoras que se explican a continuación.

Teclado.

Se debe utilizar un teclado USB estándar, para realizar las configuraciones iniciales de las placas. Una vez que se configura el servicio SSH en el sistema operativo para aceptar conexiones de acceso remoto, se puede prescindir de este dispositivo [RE18].

Pinzas ponchadoras.

Las pinzas ponchadoras o crimpeadoras, (Figura 2.6), son muy útiles para adaptar nuestros cables de red Ethernet, pueden ser de cualquier calidad siempre y cuando tengan entrada de plugs RJ45 que son los que vamos a usar para construir nuestros cables de red Ethernet.



Figura 2.6: Pinzas ponchadoras.

Switch de Ethernet.

El switch de Ethernet (Figura 2.7), es muy importante, debido a que este es el que conectará cada nodo a la red, aunque también pueden estar conectadas a través de Wi-Fi, pero la transmisión de datos puede ser más lenta y con mayor inestabilidad.
El laboratorio LACECI también nos proporcionó un switch para trabajar; más adelante se presentará una tabla con los costos de cada uno de los materiales para tener una estimación de costos para un proyecto similar, (en la **sección de costos** de este mismo capítulo).



Figura 2.7: Switch de Ethernet.

Las características de nuestro switch es de 8 puertos Ethernet RJ45 que nos permitirá conectar hasta 7 nodos (1 maestro, 6 workers y uno de alimentación a nuestro servicio de internet). Este modelo permite alcanzar una velocidad de hasta 1000 Mbps. Sin embargo, el modelo de Raspberry Pi utilizado está limitado a 100 Mbps. En versiones más recientes de Raspberry Pi este límite puede ser mejorado.

Cables v8-usb.

Este cable, (Figura 2.8), nos servirá para alimentar de corriente las RPi 3 (para el modelo de RPi que elegimos es necesario V8-USB, pero para versiones más nuevas ya aceptan la entrada tipo C).

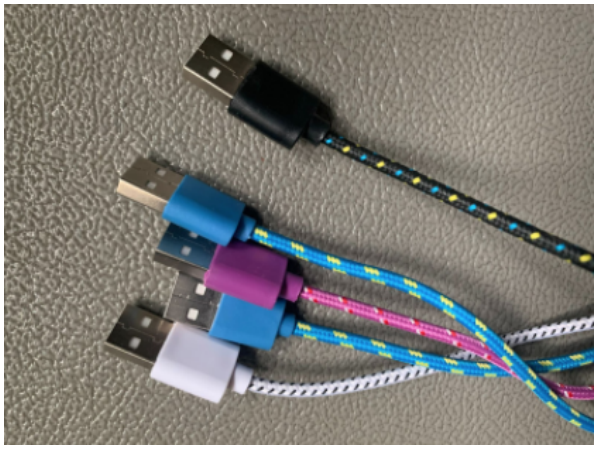


Figura 2.8: Cables v8-usb.

Podemos usar los cargadores de celulares Android que creímos que ya no usaríamos jamás, o que algunos todavía utilizan, ya que en su mayoría los celulares actuales ya cuentan con entrada tipo C o carga inalámbrica.

Memorias Micro SD.

Raspberry Pi no lleva incorporado almacenamiento en disco. Como alternativa, dispone de una ranura microSD en la cual se instala el sistema operativo. La capacidad de las tarjetas es un factor a tener en cuenta, (Figura 2.9), puesto que la instalación del sistema operativo "Raspberry Pi OS" requiere una capacidad mínima de 4GB para la versión con interfaz gráfica recomendada o 2GB para la versión LITE, pero recomendamos utilizar memorias de mínimo 8GB para poder instalar las demás herramientas y programas que utilizaremos.



Figura 2.9: Memoria Micro SD

Computadora de escritorio o Laptop (opcional)

Se necesitará una computadora inicialmente de escritorio o portátil, (ver Figura 2.10), para descargar el sistema operativo y poder instalarlo en las placas. Además, al conectar una computadora al mismo switch del clúster, no se necesita que las placas estén conectadas a un monitor/TV y un teclado para ejecutar comandos en la terminal. Para hacer esto, vía SSH (Secure SHell) se realizarán las conexiones a las diferentes placas.



Figura 2.10: Computadora portátil (Laptop).

Conexión a internet estable

Para revisar la velocidad de carga y descarga de Internet, se puede utilizar el programa speedtest, desde la página oficial [spe], donde se mostrará un dashboard como el que se ve en la Figura 2.11, con una medida de descarga y carga de datos.



Figura 2.11: Internet estable mínimo 6 mbps.

La conexión a Internet permite descargar los paquetes de las herramientas que se utilizan para configurar los servicios del clúster. Se recomienda una conexión de mínimo 6mbps.

2.1.2. Materiales Adicionales

Los materiales que se muestran a continuación, son opcionales para un mejor diseño y funcionamiento de nuestro clúster, pero no son necesarios para llevar a cabo la construcción del mismo.

Disco de estado sólido de 256 GB (opcional).

Debido a que las SBC no incluyen su propio almacenamiento puede incluirse un disco duro (HDD) o un disco de estado sólido (SSD), véase Figura 2.12, para el almacenamiento, lectura y escritura de la información de todo el clúster.



Figura 2.12: Disco de estado sólido de 256 GB.

La adquisición y configuración del disco de estado sólido es opcional, es decir, no es indispensable ya que por eso contamos con las tarjetas micro SD. Tampoco es necesario que sea de estado sólido, aunque sí se recomienda para hacer más eficientes las transacciones, y la capacidad de almacenamiento es de acuerdo a nuestras necesidades.

Ventiladores.

El uso de ventiladores es esencial para la refrigeración del clúster, estos tienen su propia fuente de alimentación USB y pueden tener un tamaño de 120mm, el cual se adapta perfectamente a la carcasa de la Raspberry Pi de cada nodo. El tamaño del ventilador depende del diseño del clúster, con el objetivo de que se adapte y se puede usar más de uno. Figura 2.13

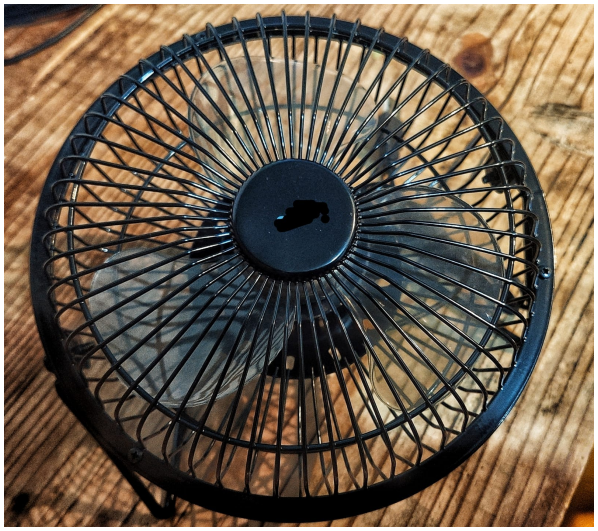


Figura 2.13: Ventilador.

Adicionalmente se pueden utilizar disipadores de calor para garantizar una buena ventilación. No obstante, por el tamaño de nuestro clúster será suficiente con los ventiladores, además nuestras carcasas son abiertas, en comparación a otras carcasas que son cerradas e impiden la ventilación.

Bases para apilar las placas SBC.

Para poder darle una mejor presentación a nuestro clúster decidimos utilizar unas bases para poder apilar y acomodar mejor nuestras placas Raspberry Pi, (ver Figura 2.14), nos decidimos por éste diseño principalmente porque es abierto y permite la ventilación, además de la elegancia, en comparación a otras carcasas cerradas que existen en el mercado. También se puede elaborar de otros materiales de forma casera, por ejemplo, piezas de LEGO, cartón, madera, acrílico, etc.
Existen carcasas cerradas con ranuras para conectar los dispositivos periféricos, las cuales no son recomendables, ya que no permiten la ventilación o en su caso usar un sistema de ventilación diferente que se adapte a las carcasas cerradas.



Figura 2.14: Acrílico para apilar las SBC.

Nobreak.

Se necesitará un nobreak, (ver Figura 2.15), para regular el voltaje y proteger las tarjetas SBC de descargas eléctricas, las Raspberry Pi 3 utilizan una fuente de alimentación micro USB de 5,1v.



Figura 2.15: Nobreak.

La RPi 3 utiliza un poco más de corriente que sus predecesoras, probablemente a causa de la CPU más robusta. Debemos tener en cuenta que ejecutar cargas pesadas o conectar múltiples artículos USB en la Raspberry Pi 3, sin su propia fuente de alimentación, puede consumir los amperios y es posible que la Pi colapse temporalmente. Por eso es importante invertir en una buena fuente de alimentación.

2.1.3. Costos

A continuación se muestra una cotización, considerando todos los materiales que se necesitan para un diseño y funcionamiento aceptable de un clúster con SBC considerando el modelo de las Raspberry Pi 4, Tabla 2.1.

COTIZACIÓN DE COSTOS PARA RASPBERRY PI 4			
Cantidad	**Descripción**	**Precio**	**Importe**
5	RASPBERRY PI 4 MODELO B	$1,470.00	$7,350.00
5	CABLE USB 2.0 DE 1.2m A MACHO Tipo C	$71.00	$355.00
1	SWITCH LINKSYS GIGABIT ETHERNET 24 PUERTOS METALICO (opcional)	$300.00	$300.00
1	HUB CONCENTRADOR 7 PUERTOS USB 3.0	$254.32	$254.32
5	Tarjeta de MEMORIA MICRO SD 16GB	$94.00	$470.00
1	DISCO DURO EXTERNO ESTADO SOLIDO SSD 1T (opcional)	$1,456.90	$1,456.90
4	CABLE DE RED POR METRO BLANCO CAT5 UTP ETHERNET GIGABIT RJ45	$12.07	$48.28
1	PINZA CRIMPEADORA INTELLINET PARA PLUGS RJ11/ RJ12 / RJ45 CON TRINQUETE	$254.32	$254.32
2	VENTILADOR MANHATTAN FUENTE DE PODER 80MM 3 PINES 80X80X25 GABINETE CPU	$140.00	$280.00
1	GABINETES DE PLASTICO PARA RASPBERRYY PI (5 pisos)	$100.00	$100.00
1	NOBREAK KOBLENZ 9022-USB/R 900VA/450W LCD 10 Contactos; Pantalla LCD; Hasta 76 min. De respaldo; Garantía 3 años (opcional)	$2,318.07	$2,318.07
1	BOLSA 100 CINCHOS NEGROS 3.6X150MM (opcional)	$25.00	$25.00
12	CONECTOR (PLUGS) RJ45	$4.31	$51.72
		SUBTOTAL	$13,263.61
		IVA	2122.1776
		TOTAL	$15,385.79

Tabla 2.1: Comparativa de costos.

Cabe mencionar que algunos materiales que estamos contemplando pueden sustituirse por otros de diferente calidad e incluso pueden no adquirirse y eso podría llegar a modificar el tiempo de vida del clúster, pero también a reducir considerablemente los costos, y continuar con un buen rendimiento. Es como un automóvil, si se invierten recursos, puede mejorar su funcionamiento, apariencia, seguridad, comodidad, etcétera. Pero también se puede tener un auto austero que te funcione para lo que necesitas y realice su función principal que es trasladarse.

La tabla de costos que se muestra en la Tabla 2.1 puede variar dependiendo del proveedor, marcas, cantidades, ofertas, equipamiento y fechas de compra, pero consideramos importante mostrar la cotización para tener una idea de lo que puede llegar a costar aproximadamente.

También se está considerando lo último en tecnología; por ejemplo, en lugar de usar RPi 3 se cotizó la RPi 4, disco de estado sólido de 1 TB (que podría ser opcional), pinzas ponchadoras que en su alternativa sería comprar el cable ya ponchado y así sucesivamente; los materiales no son estrictamente exclusivos.

2.2. Configuración General

En esta sección se detalla como instalar el sistema operativo en la RPi y cómo configurar los nodos del clúster.

2.2.1. Arquitectura del clúster (almacenamiento y red).

La arquitectura del clúster se basa en una "arquitectura maestro-workers", donde básicamente, una tarjeta Raspberry Pi funciona como el nodo maestro y el resto funcionan como nodos workers.
En el nodo maestro se instala el sistema operativo Raspberry Pi OS Estándar porque es la versión de software recomendada en la documentación del OS, la versión completa contiene funcionalidades y aplicaciones que no usaremos, y la versión Lite nos queda pequeña en caso de que necesitemos la interfaz gráfica para hacer monitoreo de los nodos por medio de un middleware [Sch13].

¿Qué es un middleware?

Es el nombre que recibe el software que se encuentra entre el sistema operativo y las aplicaciones. Su objetivo es que el usuario del cluster tenga la sensación de estar frente a una única supercomputadora, ya que provee de una interfaz única de acceso al sistema.
Mediante este software se consigue optimizar el uso del sistema y realizar operaciones de balanceo de carga, tolerancia de fallos, monitoreo de procesos, etc [AW15].

El diagrama arquitectónico se visualiza en la Figura 2.16.

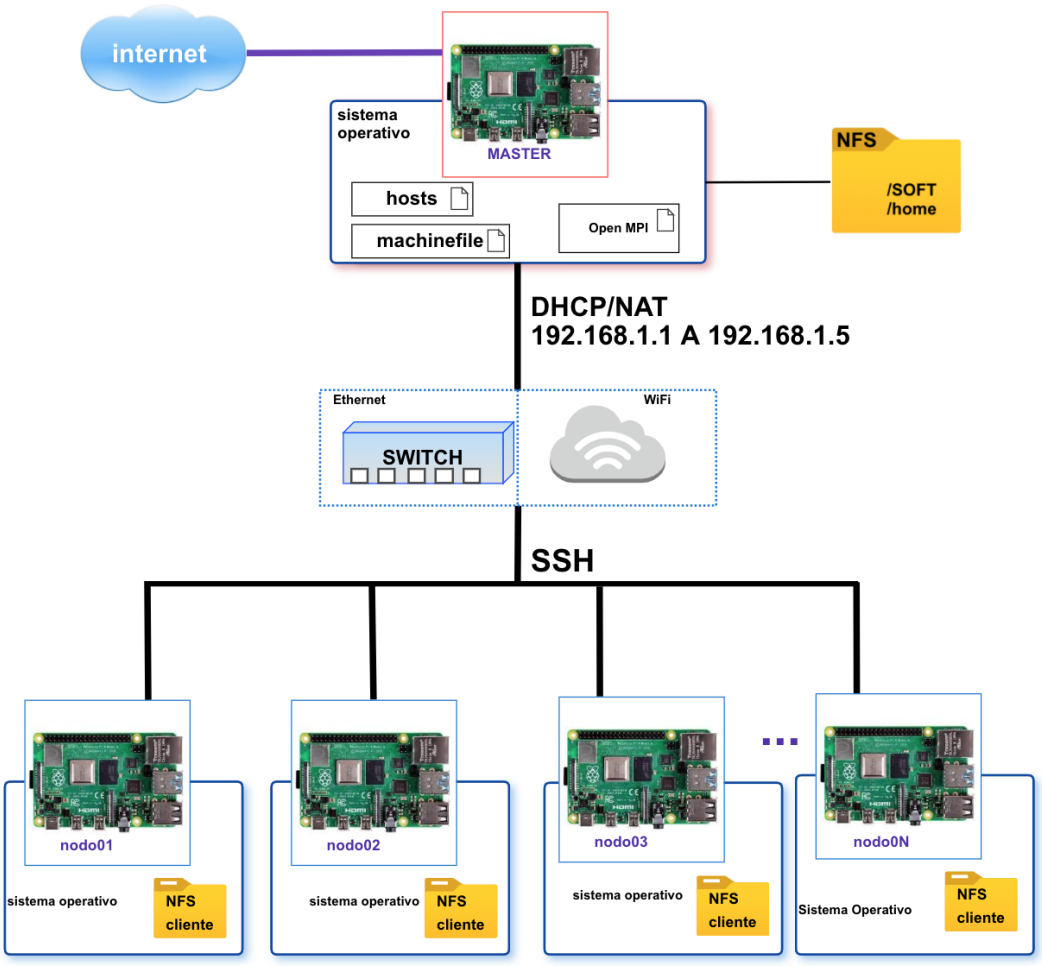


Figura 2.16: Arquitectura del clúster.

En el nodo maestro se configura la red por medio del protocolo SSH para administrar los recursos de los nodos workers, por medio de un switch de ethernet o Wi-Fi, donde la red de internet es interna, es decir, son derivados de la misma IP. El nodo maestro está conectado a internet para abastecer de internet a los demás nodos por medio del switch de ethernet o bien la conexión inalámbrica de Wi-Fi y descargar los paquetes necesarios para su funcionamiento.

En los nodos workers se instala el sistema operativo Raspberry Pi OS Lite, ya que debemos optimizar los recursos y no necesitamos de la interfaz gráfica, únicamente los utilizaremos para distribuir cargas y realizar procesamiento de datos de los algoritmos distribuidos que se ejecuten desde el nodo maestro (el cual es el orquestador).
La cantidad de nodos workers pueden ser desde uno hasta "n" nodos, pero en éste proyecto comenzaremos con un nodo maestro y 4 nodos workers.

2.2.2. Instalación del Sistema Operativo Raspberry Pi OS (anteriormente llamado Raspbian)

Existen tres diferentes imágenes disponibles para descargar del sistema operativo Raspberry Pi OS, las cuales se muestran junto con sus características en la Figura 2.17

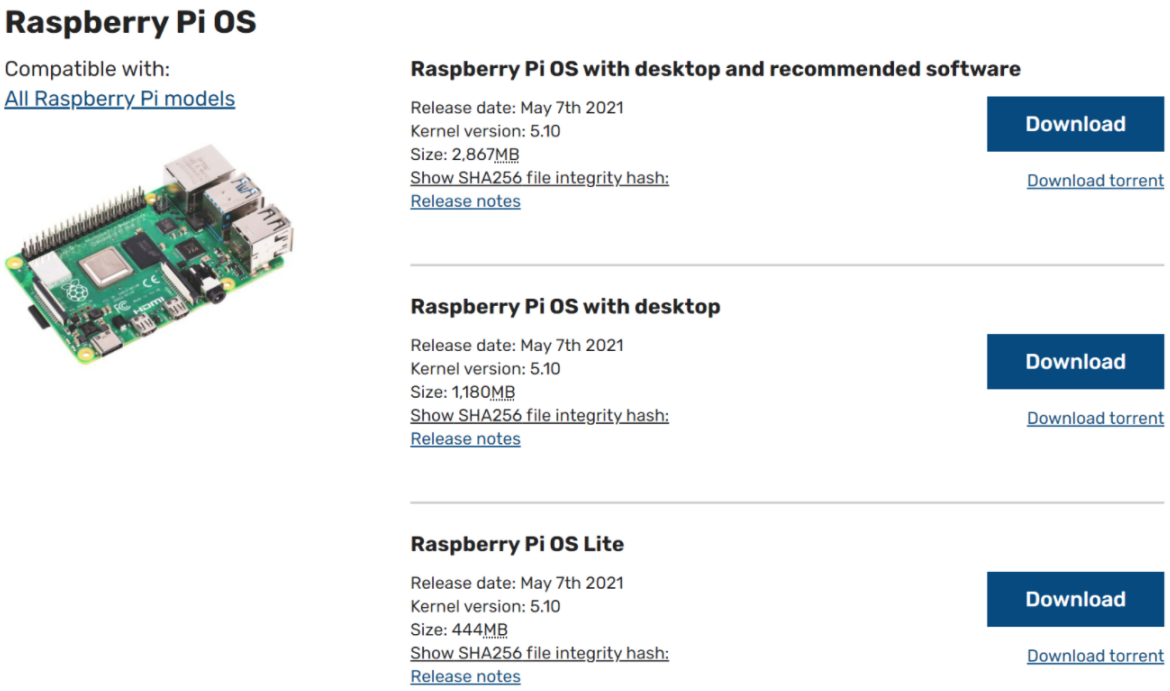


Figura 2.17: Versiones de Raspberry Pi OS.

Se utilizó "Raspberry Pi OS with desktop" para el nodo maestro y Raspberry Pi OS en su versión lite para los nodos workers.

El primer paso para la instalación del sistema operativo, es descargar la imagen de la versión de sistema operativo que deseemos instalar desde la página oficial de Raspberry [Ras].
Se eligió esta distribución porque es práctica, liviana, y recomendada por los creadores de las placas RPi; tiene una comunidad de usuarios amplia y, principalmente, porque cuenta solamente con los servicios básicos del sistema. De esta forma, se tiene la mayor parte de los recursos del sistema disponibles y permite hacer el SO totalmente personalizable.

Una vez finalizada la descarga, se obtiene un archivo con extensión zip del cual se descomprime un

archivo con extensión img. Este archivo contiene una imagen de disco el cual debe ser grabada dentro de la MicroSD.

Para escribir el archivo img dentro de la MicroSD se utilizará el software Etcher, que es recomendado en el sitio oficial de Raspberry [etc]. Etcher es una aplicación gratuita y de código abierto desarrollada para facilitar la grabación de imágenes ISO, IMG y otros formatos comprimidos directamente y de forma segura a memorias USB y tarjetas de memoria. Las principales características de esta aplicación son:

- Tiene una interfaz sencilla, agradable e intuitiva.
- Cuenta con sistemas de validación para la detección de errores.
- Es multiplataforma. (Es compatible con diferentes Sistemas Operativos).

Como se observa en la Figura 2.18, el proceso de copiado consta de 3 pasos:

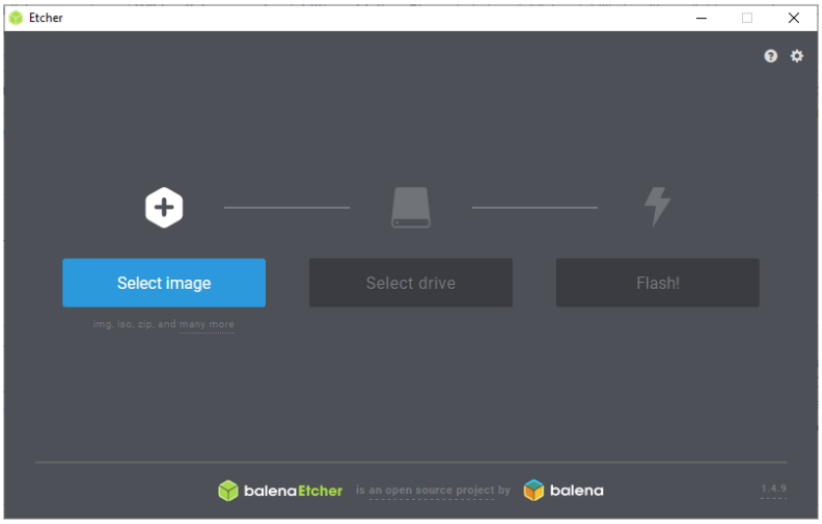


Figura 2.18: Versiones de Raspberry Pi OS.

1. **Select Image** Seleccionar la imagen ISO, IMG o cualquier otro formato compatible.
2. **Select Drive** Elegir la unidad de destino donde se grabará la imagen seleccionada en el paso 1.
3. **Flash!** Si todo está correcto, se inicia la grabación.

El proceso de grabación de la imagen puede tardar varios minutos, esto depende del tamaño del archivo img y la velocidad de la tarjeta. Una vez que finalice la copia de los datos, Etcher verifica que todo está correcto. Finalizada la grabación, la tarjeta MicroSD se encuentra en condiciones de ser insertada en la RPi.

2.3. Configuración Inicial

Existe un conjunto de configuraciones iniciales que debemos realizar antes de empezar a configurar nuestros nodos (maestro y workers). Al arrancar el sistema operativo por primera vez lo primero que se necesita hacer en la Raspberry es autenticarse; para eso se deben ingresar las credenciales de acceso provistas por el fabricante que son:

usuario: **pi**
contraseña: **raspberry**

Una vez que se tenga el control de la terminal, ya se podrán ingresar las instrucciones para configurar la RPi. A continuación, se deben elevar los privilegios del usuario pi (o cambiar a usuario root), lo cual es necesario debido a que el usuario pi no cuenta con los permisos requeridos para realizar modificaciones

en la configuración del sistema operativo. Para eso se ejecuta la instrucción 'sudo su', logrando obtener los privilegios de usuario root o superusuario.

2.3.1. Conexión a la Red de internet

La RPi se pueden conectar inalámbricamente a la red, debido a la antena de Wi-Fi que tiene integrada e independientemente de la versión de Raspberry Pi OS, podemos configurar la conexión de Wi-Fi de dos formas:
1.- Editando el archivo wpa_supplicant.conf
de la siguiente forma:

```
cd /etc/wpa_supplicant
wpa_passphrase "nombre_SSID" "contraseña_SSID" >> wpa_supplicant.conf
```

2.- Usando las instrucción raspiconfig y seleccionando la opción 1, luego S1, e ingresando los datos de SSID y PASSWORD, como se visualiza en la figura 2.19.



Figura 2.19: Pasos de configuración a Wi-Fi.

3.- Existe una tercera forma, pero está disponible en la versión de sistema operativo con interfaz gráfica, consiste en dar click en la esquina superior derecha en el símbolo de Wi-Fi, y ahí se visualizarán las redes de Wi-Fi que están disponibles, solo tenemos que seleccionar el nombre de SSDI a la que nos queremos conectar e ingresar su contraseña.

En caso de que contemos con conexión alámbrica, basta con conectar todos los nodos incluyendo el nodo maestro al switch de Ethernet y el switch debe estar conectado a nuestro módem, como a la alimentación de corriente para tener internet en todas las placas y puedan comunicarse entre ellas.

2.3.2. Modificaciones desde el menú de configuración

Al contar con una conexión a Internet, el siguiente paso será ejecutar la instrucción **raspi-config**. Con este comando se visualiza un menú interactivo ver Figura 2.20 que simplifica la tarea de hacer modificaciones del sistema, sin tener que editar archivos manualmente como se hizo anteriormente con el archivo wpa_supplicant.conf.

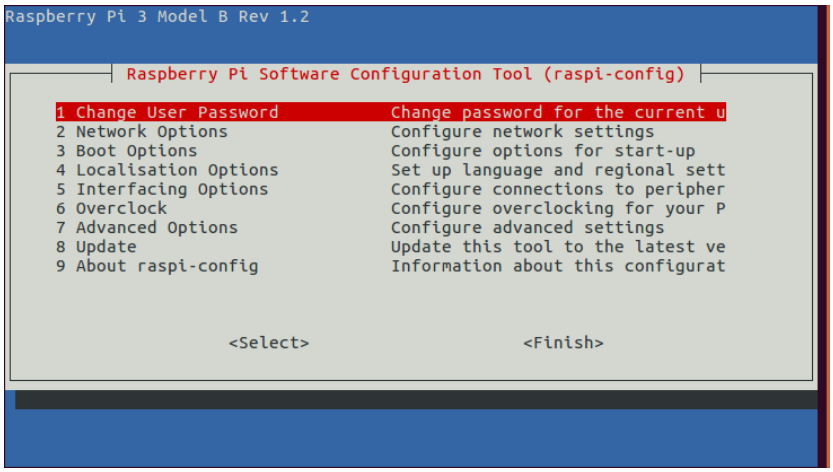


Figura 2.20: Menú de configuración.

Actualización de los paquetes del sistema

Lo primero es actualizar los paquetes del sistema. Para esto se debe seleccionar la opción 8 **Update** del menú mencionado en la Figura 2.20, lo que inicia la actualización de los paquetes del sistema.

Finalizada la actualización, se volverá a visualizar el menú. Lo siguiente es configurar el acceso remoto y maximizar el tamaño de la memoria RAM, para llevar esto a cabo se deben seguir los siguientes pasos:

- **Acceso remoto:** Seleccionar la opción 5 Interfacing Options. En la siguiente pantalla, seleccionar la opción P2 SSH y por último confirmar la operación seleccionando la opción "Yes". A continuación se visualiza un mensaje confirmando la habilitación del servicio SSH, el cual permitirá iniciar conexiones remotas a la placa RPi.
- **Maximizar la memoria RAM:** Configurar el tamaño de la memoria asignada a la GPU, lo que permite tener la mayor cantidad de memoria RAM para el momento de realizar las pruebas. Para realizar esta configuración se debe seleccionar la opción 7 Advanced Options, y a continuación seleccionar la opción A3 Memory Split. En la siguiente ventana se solicita ingresar el nuevo tamaño de la memoria el cual será 16, que es el mínimo de memoria asignable para gráficos. Aceptar para confirmar la operación.
- **Analizar el asistente:** Para garantizar que las configuraciones realizadas surtan efecto, se debe analizar el asistente de configuración. Esto se consigue al seleccionar la opción “Finish” y confirmando el reinicio del sistema.

Con este último paso se completa la configuración básica de la primera RPi, la cual servirá como punto de partida para configurar los nodos del clúster. Para evitar repetir este proceso para cada placa restante, es recomendable generar una imagen del sistema almacenado en tarjeta microSD.

2.3.3. Configuración del nodo maestro

Como se mencionó anteriormente, el nodo maestro es el nodo desde el que se administra el clúster. En él se llevan a cabo tareas como lanzamiento de trabajos, instalación de librerías y programas, obtención de los resultados de la ejecución de las aplicaciones, etc. Además, es el punto de acceso al resto de los equipos que conforman el clúster. A continuación se realiza una breve descripción de los elementos mínimos con los que deberá contar el nodo maestro para administrar el clúster (los mismos serán desarrollados posteriormente en detalle en esta sección):

- **Generación de claves:** para permitir la autenticación por clave pública y evitar la autenticación con usuario y contraseña.
- **Configuración de red:** necesaria para permitir la interconexión con el resto de los nodos.
- **Protocolo de configuración dinámica de host (DHCP, por sus siglas en inglés):** es necesario para el aprovisionamiento de los parámetros de red a cada uno de los nodos workers.
- **Sistema de archivos de red (NFS, por sus siglas en inglés):** permite compartir cualquier archivo o programa que se desee a través de la red, para dejarlo accesible desde todos los nodos del clúster.
- **Traducción de direcciones de red (NAT, de sus siglas en inglés):** permite utilizar el nodo maestro como un punto de acceso a Internet para cada uno de los nodos.
- **Librerías Open MPI:** necesarias para compilar los algoritmos paralelos que serán ejecutados en el clúster con MPI y/o Open MPI.
  MPI (Message Passing Interface) es un estándar que define la sintaxis y la semántica de las funciones contenidas en una biblioteca de paso de mensajes diseñada para ser usada en programas que exploten la existencia de múltiples procesadores.

A continuación se inicia la configuración del nodo maestro partiendo de la configuración básica descrita anteriormente:

Conectarse al nodo maestro

Debemos estar conectados al nodo maestro para ejecutar los comandos. Para conectarse al nodo maestro existen muchas opciones, de las cuales mencionaremos tres formas diferentes:

1. **El flujo ideal:** Sería conectar un teclado y una pantalla a la placa RPi, encender la placa, escribir el usuario y contraseña (pi, raspberry), y listo podemos hacer uso de la terminal de Raspbian.

2. **El flujo alterno:** Es cuando no tenemos una pantalla ni un teclado, pero tenemos una computadora con algún sistema operativo basado en Linux, en ese caso se puede conectar la placa a través de un cable Ethernet y ejecutar la instrucción SSH <ipDeLaMaquina>y listo estarán conectados al nodo maestro, o bien, podemos conectar el nodo maestro a una conexión Wi-Fi, los pasos a la conexión Wi-Fi se describen en la sección de **Conexión de Raspberry Pi OS a Wi-Fi** muy útil para conectarse desde un sistema operativo privativo.

3. **Otro caso alterno:** En caso de usar sistema operativo privativo se puede hacer uso del programa Putty para conectarse a través de SSH. Para este caso tenemos que instalar el programa "Putty" o algún otro programa que permita conexiones a través de SSH, una vez que "Putty" esté instalado es posible conectarse con la IP del nodo maestro, el usuario y contraseña, (para obtener la IP del nodo maestro usamos la instrucción **ifconfig**, por lo que este caso sirve cuando ya conocemos la IP), de esa forma estaremos conectados al nodo maestro [put].

2.3.4. Configuración de red

Para que todos los nodos puedan funcionar como un clúster, deben estar interconectados entre sí, en la Figura 2.16 podemos identificar al nodo maestro, con sus servicios y los 4 nodos workers.

2.3.5. Generación de claves

Desde una PC conectada a la misma red Wi-Fi o ETHERNET configurada anteriormente, conectarse a la RPi utilizando la IP por medio de un cliente SSH. Notar que el sistema operativo solicita las credenciales de acceso, utilizar las mismas que se utilizaron para hacer el primer ingreso al sistema:

usuario: **pi**
contraseña: **raspberry**

SSH permite utilizar una amplia variedad de mecanismos de autenticación diferentes entre los que se incluyen usuario y contraseña. Posiblemente este sea el mecanismo más utilizado por muchos para autenticar una sesión SSH, pero SSH también soporta la autenticación mediante clave pública. La autenticación con clave pública funciona con dos claves: una pública y otra privada. Para entender el funcionamiento imaginemos que se cuenta con un candado y su llave. La clave pública funciona como un candado y la privada como la llave. El candado se colocará en el dispositivo remoto al que se quiere acceder; cuando se intenta acceder se comprobará que la máquina que intenta conectar tiene la llave, que en este caso resulta ser la clave privada [BBSS01]. Para configurar el acceso SSH con clave pública se debe:

- Generar el par de claves pública/privada.
- Copiar la clave pública a cada uno de los nodos.

Para generar las claves ejecutaremos el siguiente comando en la terminal: **ssh-keygen -t rsa** Como resultado de la ejecución dla instrucción **ssh-keygen** se generan dos archivos: **id_rsa** es la clave privada, la que permanecerá en la máquina local. **id_rsa.pub** es la clave pública, la que se tiene que copiar a cada nodo worker. Ambos archivos generalmente se guardan en la ruta /home/pi/.ssh, sin embargo, al ejecutar la instrucción te dirá la ruta exacta en la línea "your identification has been saved in.º en la línea "Enter file in which to save the key". Durante la generación de claves, el sistema solicita asignar una contraseña para proteger la clave privada. Sin embargo, como OpenMPI requiere un passphrase vacío, no se asignará ninguno. Una vez generado el par de claves, hay que copiar la clave pública en cada nodo remoto, a continuación se muestran los pasos a seguir, Figura 2.21

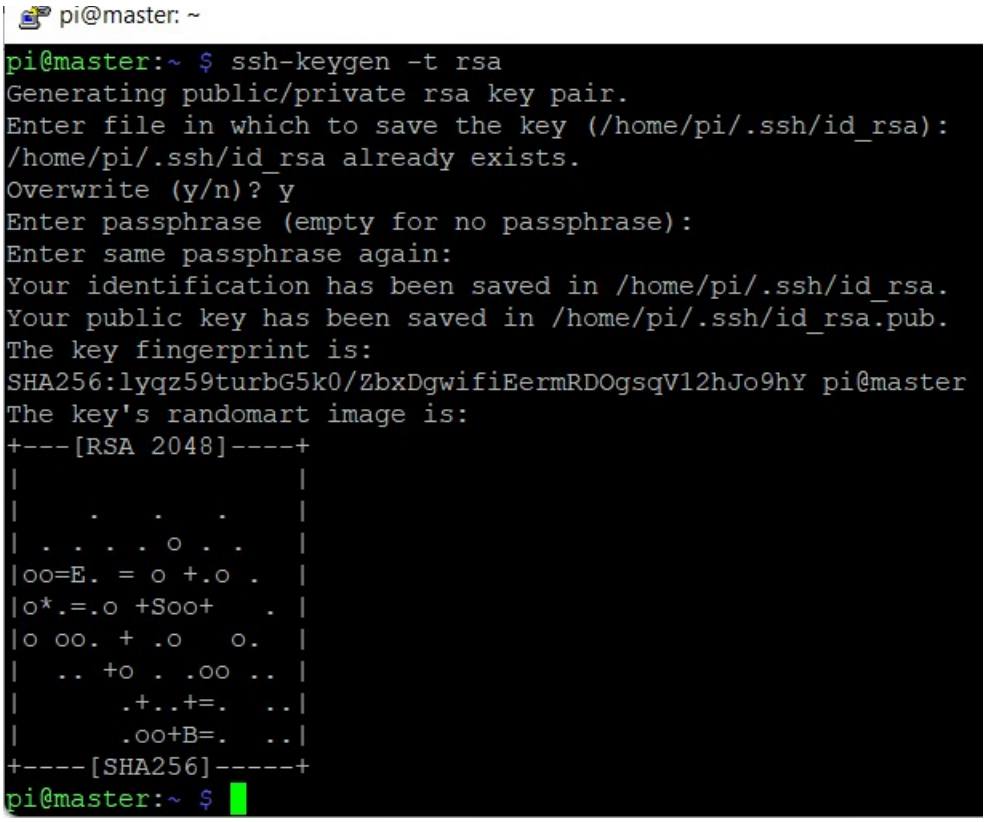


Figura 2.21: Salida de la instrucción ssh-keygen -t rsa

Pasos para copiar la llave pública a los nodos workers

Para conectarnos a cada nodo worker nos solicita un usuario y contraseña, para evitar eso debemos copiar la clave pública a los nodos workers en el directorio .ssh, eso nos permitirá conectarnos sin necesidad de que nos pida usuario y contraseña.

Anteriormente habíamos visto que la generación de claves se genera con la instrucción **ssh-keygen -t rsa**, ahora vamos a copiar únicamente la llave pública a cada uno de los nodos workers.

1. Nos dirigimos al directorio donde se generó la key pública con la instrucción: cd /home/pi/.ssh/ presionamos enter y nos aparecerán dos archivos:
   id_rsa
   id_rsa.pub
   Uno es la llave privada y el otro con terminación .pub es la llave pública.

2. Copiamos y pegamos la key al primer nodo con la instrucción:

   scp id_rsa.pub pi@nodo01:/home/pi/.ssh

   Inicialmente generamos las llaves ssh desde la configuración inicial para poder tener acceso a los nodos y por proceso ortodoxo copiamos las llaves públicas a los nodos con el comando scp archivo.txt usuario@dominio.com:/home/usuario/.ssh, pero al momento de montar el sistema de archivos NFS en los nodos, estas llaves no funcionaron para poder acceder a los nodos sin password, por lo que tuvimos que utilizar un comando especial:

   ssh-copy-id nodo01

   que resolvió el problema del password e incluso le permite a los nodos acceder entre ellos debido

a que comparten el mismo directorio /home/pi/.ssh.

3. Para poner la IP fija a cada nodo, debemos estar en el nodo maestro y dirigirnos a la ruta donde se encuentre nuestro archivo hosts que generalmente se encuentra en **/home/etc** en caso de que el archivo no tenga permisos de edición se los agregamos con la instrucción **sudo chmod +wr nombreDelArchivo** posteriormente podemos editar el archivo con la instrucción: **nano** y nos quedaría de la siguiente forma, donde la primer IP es el nodo maestro y el resto son los nodos workers, como se visualiza en la Figura 2.22

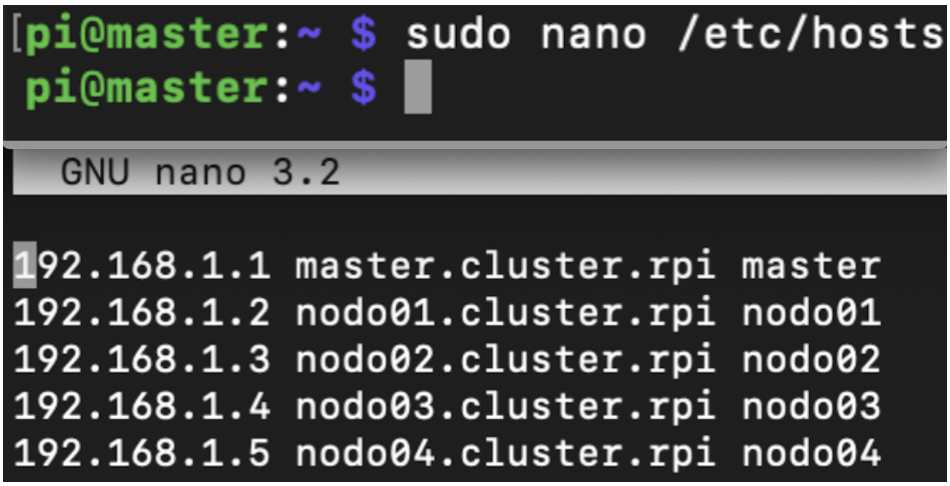


Figura 2.22: Configuración de archivo hosts para establecer la IP fija de los nodos workers

Una vez creado el nuevo documento hosts, el nodo maestro será capaz de detectar los distintos nodos del clúster mediante su nombre de equipo.

2.3.6. DHCP (Protocolo de configuración dinámica de host)

DHCP permite que la configuración IP de los nodos se realice de manera automática, evitando así la necesidad de tener que realizar manualmente uno por uno la configuración de la interfaz de red de cada nodo, la instrucción para instalar DHCP es **apt install isc - dhcp - server -y**.

El siguiente paso será modificar el archivo /etc/hosts. Este archivo se utiliza para obtener una relación entre un nombre de equipo y una dirección IP [Smi02]. Los comandos a ejecutar son los siguientes:

Se elimina el archivo hosts actual.

rm /etc/hosts
Presionamos enter y creamos las línea correspondientes a los nodos del cluster.

En el caso de nuestro clúster decidimos ponerle las siguientes ips porque son fáciles de recordar, debido a que en México son ips muy comunes que asignan los routers automáticamente a los dispositivos conectados y en nuestro caso llevan una numeración consecutiva e incremental, reservando la **ip 192.168.1.1** para el nodo maestro y sus consecutivas para los nodos workers.

```
cat << EOF >> /etc/hosts
192.168.1.1 master.cluster.RPi master
192.168.1.2 nodo01.cluster.RPi nodo01
192.168.1.3 nodo02.cluster.RPi nodo02
192.168.1.4 nodo03.cluster.RPi nodo03
192.168.1.5 nodo04.cluster.RPi nodo04
EOF
```

Una vez generado el nuevo archivo hosts, el nodo maestro será capaz de identificar los distintos nodos del clúster por medio de su nombre de equipo.

```
# Hacemos un backup del archivo de configuración original.
cp /etc/dhcp/dhcpd.conf ,.backup
# Eliminamos el contenido del archivo de configuración.
cat /dev/null >/etc/dhcp/dhcpd.conf
# Editamos el archivo del servidor DHCP con nuestros parámetros.
```

```
cat << EOF >> /etc/dhcp/dhcpd.conf
option domain-name "cluster.RPi";
option domain-name-servers 8.8.8.8,8.8.4.4;
default-lease-time 600;
max-lease-time 7200;
ddns-update-style none;
authoritative;
subnet 192.168.1.0 netmask 255.255.255.0 {
range 192.168.1.1 192.168.1.100;
option subnet-mask 255.255.255.0;
option routers 192.168.1.254;
option broadcast-address 192.168.1.255;
host nodo01{
hardware ethernet b8:27:eb:8f:e1:45;
fixed-address 192.168.1.1;
option host-name "nodo01";
}
host nodo02{
hardware ethernet b8:27:eb:39:51:97;
fixed-address 192.168.1.2;
option host-name "nodo02";
}
host nodo03{
hardware ethernet b8:27:eb:dc:12:20;
fixed-address 192.168.1.3;
option host-name "nodo03";
}
host nodo04{
hardware ethernet b8:27:eb:e5:f7:32;
fixed-address 192.168.1.4;
option host-name "nodo04";
}
}
EOF
```

2.3.7. Cambiar el nombre del host

Para cambiar el nombre del host lo podemos hacer de la siguiente manera: con la instrucción **sudo raspi-config** nos abrirá una ventana donde seleccionaremos System Options ->Hostname ->Aceptar ->Escribir hostname ->Aceptar. Figura 2.23.

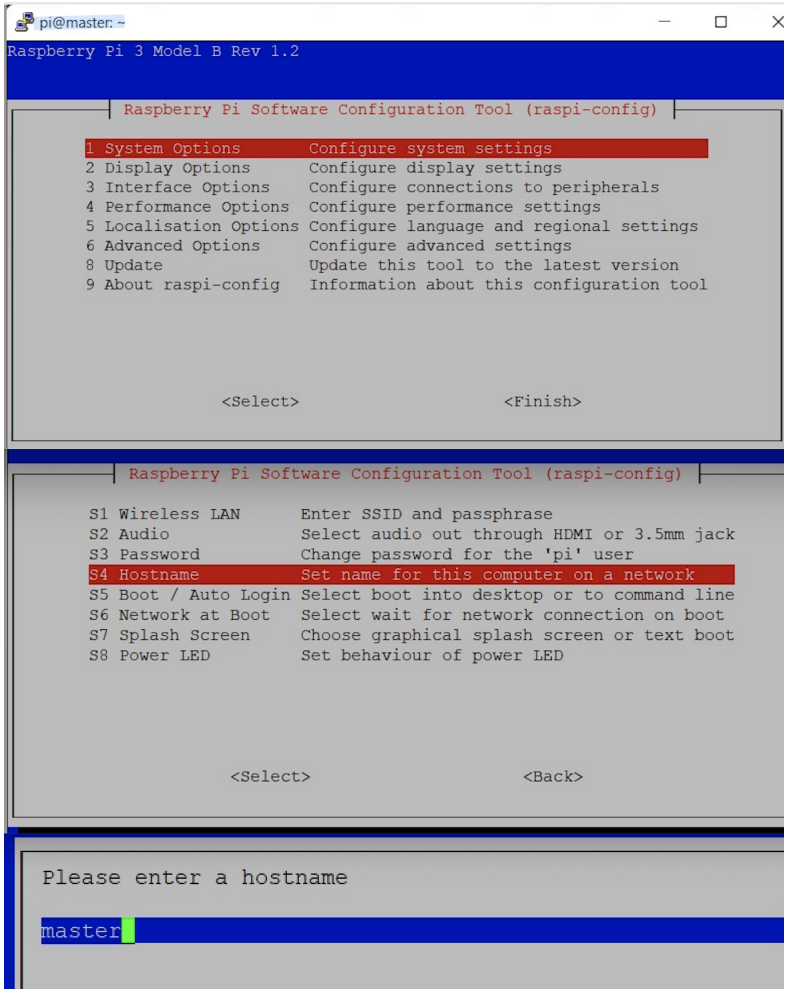


Figura 2.23: Menú para cambiar el hostname

2.3.8. NFS

Para compartir directorios desde el nodo maestro con los nodos workers, utilizamos el sistema de archivos de red NFS (Network File System, NFS por sus siglas en inglés) [Cal99].

Para su configuración, realizamos las siguientes instrucciones:

```
# Instalamos los paquetes
apt install nfs-kernel-server nfs-common -y
# Creamos el directorio donde se alojan las librerías de compilación
mkdir /SOFT
# Cambiamos sus permisos
chown pi:pi /SOFT
# En el archivo /etc/export configuramos los directorios que se compartirán en la red
cat « EOF » /etc/exports
/SOFT 192.168.1.0/255.255.255.0(rw,sync,no_subtree_check,no_root_squash)
/home 192.168.1.0/255.255.255.0(rw,sync,no_subtree_check,no_root_squash)
EOF


# Crea en el servidor NFS la tabla de directorios compartidos.
exportfs -a


# Reiniciamos el servicio
systemctl restart nfs-kernel-server
```

2.3.9. NAT

Se debe modificar el parámetro **net.ipv4.ip_forward=0** en el archivo /etc/sysctl.conf, y cambiarlo por **net.ipv4.ip_forward=1**.

El siguiente comando permite al nodo maestro compartir la conexión a Internet configurando NAT entre la interfaz wlan0 y la interfaz eth0:

```
iptables -t nat -A POSTROUTING -s 192.168.1.0/24 -o wlan0 -j MASQUERADE
```

Capítulo 3

Desarrollo e implementación de un software distribuido básico

3.1. Diagramas de arquitectura de Software

En esta sección se describe el software que se utiliza en el nodo maestro y en los nodos trabajadores y cómo interactúa el software.

Comenzaremos a describir las piezas de software que componen al nodo maestro como se visualiza en la Figura 3.1:

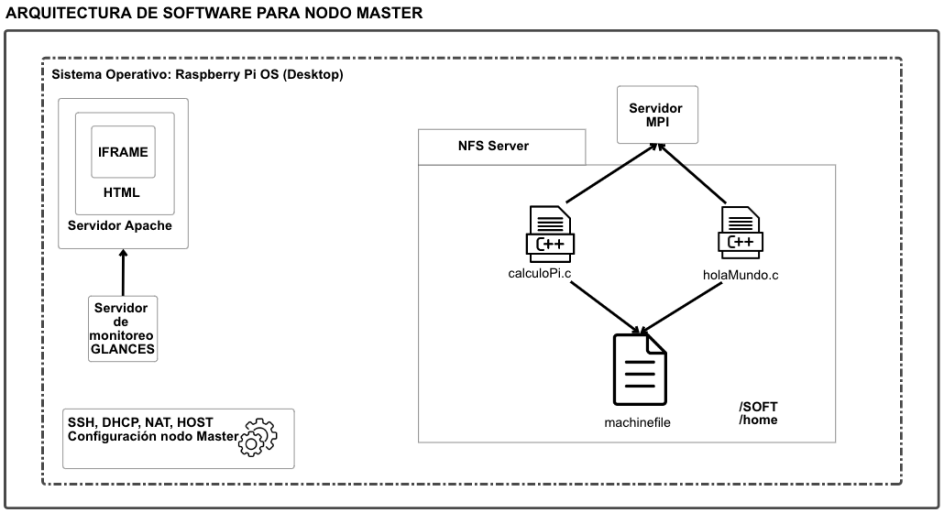


Figura 3.1: Diagrama de arquitectura de software del nodo maestro.

Como base se encuentra el sistema operativo Raspberry Pi OS Desktop, (esta versión incluye interfaz gráfica, pero solo con la paquetería indispensable) que nos servirá para visualizar el monitoreo de los nodos trabajadores.

Se instala y configura un NFS Server(Network File System) con dos directorios que se compartirán a los nodos trabajadores: /SOFT y /home.

Dentro de estos directorios compartidos se encontrarán los programas en C que se desean compilar como el 'holaMundo.c' y el 'calculoPi.c', así como el 'machinefile' que es el que hace referencia a cada uno de los núcleos de cada nodo trabajador configurado dentro del host.

Estos programas en C implementan una biblioteca OpenMPI para el envío de mensajes, útiles para el procesamiento paralelo.

También se requiere una configuración especial de SSH (para las conexiones), DHCP (para la red), HOST (para identificar las IP) y NAT (para tener acceso a internet a través del nodo maestro).

Adicionalmente se instala un servidor de monitoreo (Glances) que nos ayudará a obtener los datos de cada uno de los sensores, memoria y procesador de las SBC (Single Board Computer), integrado a un servidor Apache para desplegar una página con HTML y Iframe para visualizar los datos obtenidos con Glances y poder monitorear el status de cada uno de los nodos incluyendo al nodo maestro.

La configuración de software de los nodos trabajadores es similar a la del nodo maestro, solo cambia en algunas cosas que se pueden visualizar en la Figura 3.2 y se describen a continuación:

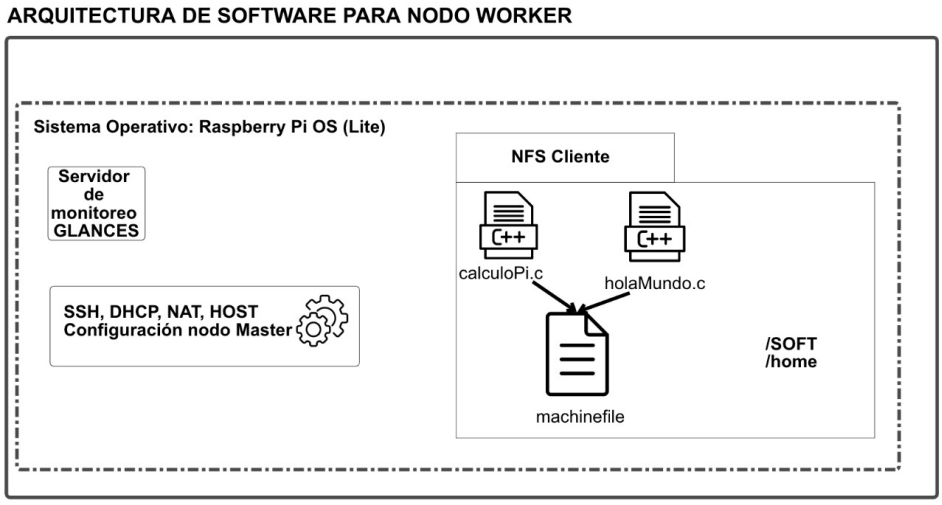


Figura 3.2: Cables v8-usb.

El sistema operativo de los nodos trabajadores no necesitan la interfaz gráfica ya que únicamente realizarán cálculos u otros procesos, es por eso que tendrán instala la versión Raspberry Pi OS (Lite).

También deben tener instalada una versión de NFS Cliente para que compartan los archivos con el nodo maestro, es decir, todos los archivos que estén dentro de los directorios /SOFT y /home en el nodo maestro, también estarán en los nodos trabajadores y si se hace algún cambio dentro de los directorios se replicará el cambio en todos los nodos incluido el nodo maestro.

Requiere una configuración de SSH DHCP, NAT y HOST explicada anteriormente en el nodo maestro.

Adicionalmente se instala un servidor de monitoreo Glances, para obtener los datos de cada sensor, memoria y procesador para mostrar esa información simplemente al hacer referencia a su IP o nombre de host ya sea desde el nodo maestro o de manera independiente, es decir, se puede monitorear individualmente sin necesidad del nodo maestro.

3.2. OpenMPI

OpenMPI es la interfaz de envío de mensajes de código abierto desarrollada y mantenida por un consorcio de socios académicos, de investigación y de la industria. Por lo tanto, OpenMPI puede combinar la experiencia, las tecnologías y los recursos de toda la comunidad de informática de alto rendimiento para crear la mejor biblioteca MPI disponible. OpenMPI ofrece ventajas para los proveedores de sistemas y software, desarrolladores de aplicaciones e investigadores de informática[doc].

En esta sección se instalará OpenMPI y se adaptará a la arquitectura que se está utilizando, también usaremos el lenguaje C (que ya viene precargado en esta distribución de Linux), para ejecutar algunos

programas en paralelo haciendo uso de los múltiples nodos, aunque para el cómputo de alto rendimiento también se pueden usar otros lenguajes de programación como por ejemplo fortran [OC14].

Para comenzar con la instalación de OpenMPI podemos hacer uso del siguiente script:

3.2.1. Script de instalación MPI

```
#!/ bin / bash
#Nos posicionamos en el directorio /tmp con el instrucción:
cd /tmp
#Descargamos los archivos fuentes de OpenMPI con el instrucción wget:
wget https://download.open-mpi.org/release/open-mpi/v3.0/openmpi-3.0.0.tar.bz2
#Descomprimimos el archivo descargado con el instrucción tar:
tar -xf openmpi-3.0.0.tar.bz2
#Accedemos al directorio que se descomprimió.
cd openmpi-3.0.0
#Creamos el directorio "build" donde se irán compilando los proyectos
mkdir -p build
#Creamos el directorio donde se instalará OpenMPI.
mkdir /SOFT/openmpi
#Le damos permisos al usuario Pi sobre el directorio
chown pi:pi /SOFT/openmpi
#Ejecutamos la configuración y preparamos las directivas para Make, eso hará que se adapte
#a nuestra arquitectura (este paso puede tardar varios minutos)
../configure CFLAGS=-march=armv7-a CCASFLAGS=-march=armv7-a
--prefix=/SOFT/openmpi/ --enable-mpi-fortran --disable-mpi-cxx
--disable-mpi-cxx-seek --disable-java --disable-ipv6
#Compilamos e instalamos (también tardará varios minutos).
make -j 3 && make install
```

Es necesario definir 3 variables de entorno para poder usar MPI, (Una variable de entorno es un valor dinámico cargado en memoria, que puede ser utilizado por varios procesos simultáneamente), para eso editamos el archivo /home/pi/.bashrc agregando al final del mismo las siguientes lineas, (si no se cuenta con ninguna variable de entorno se pueden agregar al principio del archivo), y guardar los cambios:

```
export OPENMPI_PATH=/SOFT/openmpi
export PATH=\$OPENMPI_PATH/bin:\$PATH
export LD_LIBRARY_PATH=\$OPENMPI_PATH/lib:\$LD_LIBRARY_PATH
```

Se requiere hacer un cambio más en el archivo .bashrc, en el comienzo del archivo aparece un conjunto de líneas:

```
    case \$- in
    *i*) ;;
    *) return;;
    esac
```

Es necesario comentar estas cuatro líneas, para que no tengan efecto cuando un usuario inicia sesión. Esto se debe a que cuando iniciemos sesión remotamente a través de SSH desde el nodo maestro a cualquiera de los nodos trabajadores podría generar problemas cada vez que MPI envíe un mensaje

a un nodo. Entonces, anulando estas líneas, se evita que se solicite algún dato al momento de la autenticación e iniciar sesión de forma no interactiva.

Para probar la versión de MPI se ejecuta el instrucción:
ompi_info
y de esa manera se visualizan todas las bibliotecas instaladas, Figura 3.3

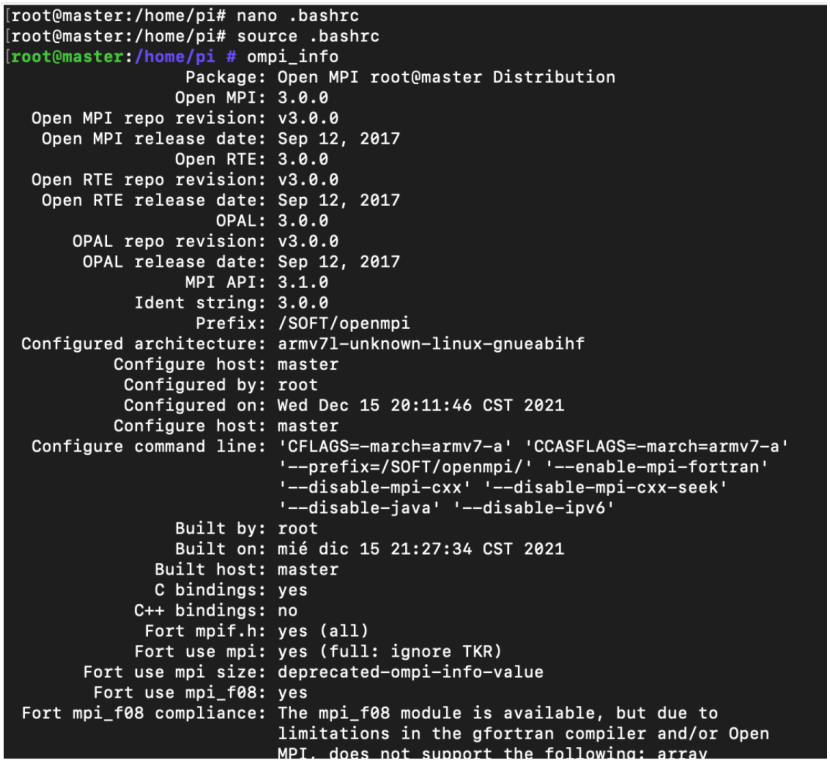


Figura 3.3: ompi_info

3.3. Configuración de los nodos trabajadores

Esa fue la última configuración que se requiere en el nodo maestro y ahora podemos pasar a los nodos trabajadores, cada nodo debe tener su configuración inicial, comenzando por una conexión wifi (o Ethernet).

El proceso de configuración se inicia ingresando los parámetros para establecer una conexión WIFI.

```
1 cd /etc/wpa_supplicant
2 wpa_passphrase "nombre-de-SSID" "contraseña-SSID" >> wpa_supplicant.conf
```

Para ver la configuración de wifi se puede consultar el archivo wpa_supplicant.conf con el editor de texto “nano” o con el editor de texto “vi”, por ejemplo:

nano /etc/wpa_supplicant/wpa_supplicant.conf

Los siguientes pasos de configuración serán actualizar los paquetes, habilitar la conexión SSH y nombrar el nodo desde el menú gráfico con el instrucción **sudo raspi-config**, (la configuración wifi también se

puede realizar desde ese mismo menú), esos pasos son similares a los que se realizaron el nodo maestro en la subsection 2.3.2 del capitulo 2.

Para que los nodos tengan acceso a Internet únicamente desde el nodo maestro se debe deshabilitar la interfaz wifi. Se debe agregar la línea **dtoverlay=pi3-disable-wifi** al final del archivo /boot/config.txtz para que la configuración tenga efecto, es necesario reiniciar la RPi con el instrucción:

sudo shutdown -r now
o
sudo reboot now

Dejar en blanco el archivo hostname porque el nombre del host será asignada a partir del DHCP, (se puede hacer uso de algún editor de texto como vi o nano), por ejemplo:
nano /etc/hostname

Se borra todo con la tecla delete usamos la combinación de teclas CTRL+O luego, CTRL+X

En cada uno de los nodos trabajadores se debe eliminar el archivo hosts y volver a generar de la siguiente manera como en el nodo maestro:

rm /etc/hosts

#Creamos las líneas correspondientes a los nodos del cluster.

```
#!/ bin / bash
cat << EOF >> /etc/hosts
192.168.1.1 maestro.cluster.rpi maestro
192.168.1.2 nodo01.cluster.rpi nodo01
192.168.1.3 nodo02.cluster.rpi nodo02
192.168.1.4 nodo03.cluster.rpi nodo03
192.168.1.5 nodo04.cluster.rpi nodo04
EOF
```

3.4. NFS CLIENTE

Para evitar la duplicidad de archivos en cada uno de los nodos que conforman el cluster, el nodo maestro comparte un conjunto de directorios que contendrán los archivos necesarios para la ejecución de aplicaciones a través del sistema de archivos de red cliente (NFS). De esta manera, se garantiza el acceso de todos los nodos trabajadores a los mismos archivos. El siguiente script muestra la configuración del NFS cliente:

3.4.1. script para la configuración del NFS cliente

```
#!/ bin / bash
#Instalamos el cliente NFS con el siguiente instrucción:
sudo apt-get install nfs-common -y
#Creamos el directorio /SOFT donde se montará el directorio remoto,
#(el directorio /home ya lo tiene creado por default)
```

```
mkdir /SOFT
# Modificamos los permisos del directorio al usuario pi
#para que pueda hacer uso de ellos.
chown pi:pi /SOFT
# Editamos el archivo /etc/fstab para montar los directorios
#compartidos de manera automática desde el arranque de los nodos.
cat << EOF >> /etc/fstab
maestro:/SOFT /SOFT nfs defaults 0 0
maestro:/home /home nfs defaults 0 0
EOF
```

En el script anterior describe la configuración para montar los directorios con su contenido, es importante que cada sistema de archivos se indique en una línea para que sea convertido con systemd en unidades montadas automáticamente desde el arranque o cuando se recargue la configuración del administrador del sistema.

El fstab es leído por la orden mount, que es la que encuentra cualquiera de los directorios o dispositivos.

Las líneas para montar los directorios presentan el siguiente formato:

<file system> /<dir> <type> <options> <dump> <pass>
donde cada parámetro se describe a continuación:

<file system>: Define la partición o dispositivo de almacenamiento para ser montado.

<dir>: Especifica a la orden mount el punto de montaje donde la partición (<file system>) será montada.

<type>: Indica el tipo de sistema de archivos de la partición o dispositivo de almacenamiento para ser montado. Hay diferentes sistemas de archivos que son compatibles como por ejemplo: ext2, ext3, ext4, reiserfs, xfs, jfs, smbfs, iso9660, vfat, ntfs, swap y auto. El type auto permite a la orden mount determinar qué tipo de sistema de archivos se utiliza. Esta opción es útil para proporcionar soporte a unidades ópticas (CD/DVD).

<options>: Indica las opciones de montaje que la orden mount utilizará para montar el sistema de archivos. Tener en cuenta que algunas opciones de montaje son para sistema de archivos específicos. Algunas de las opciones más comunes son:

- **auto** : El sistema de archivos será montado automáticamente durante el arranque, o cuando la orden mount -a se invoque.
- **noauto** : El sistema de archivos no será montado automáticamente, sólo cuando se le ordene manualmente.
- **ro** : Monta el sistema de archivos en modo sólo lectura.
- **rw** : Monta el sistema de archivos en modo lectura-escritura.
- **nouser** : Sólo el usuario root puede montar el sistema de archivos.
- **users** : Permite a cualquier usuario perteneciente al grupo users montar el sistema de archivos.
- **owner** : Permite al propietario del dispositivo montarlo.
- **sync** : Toda E/S se debe hacer de forma sincrónica.
- **defaults** : Asigna las opciones de montaje predeterminadas que serán utilizadas para el sistema de archivos. Las opciones predeterminadas para ext4 son: rw, suid, dev, exec, auto, nouser, async.

Se debe reiniciar el nodo y se puede ejecutar la instrucción “df -h” para obtener un listado de los directorios montados en el sistema.

3.4.2. PRUEBAS DE NFS

El sistema de archivos de red (Network File System, NFS), es un protocolo para almacenar y compartir archivos y/o directorios en una red. Es un sistema de archivos distribuido que permite a los nodos acceder a los archivos y directorios ubicados en sistemas remotos y tratar dichos archivos y directorios como si fueran locales.

Por ejemplo, si creamos un archivo en cualquier nodo, todos los nodos que estén configurados en el NFS, podrán acceder al archivo, siempre y cuando estos archivos se encuentren dentro de los directorios compartidos predeterminados en este caso /SOFT y /home.

Un ejemplo práctico sería, entrar a cualquier nodo e ir a alguna ruta de /SOFT o /home, en este caso nos posicionamos en la ruta por defecto que es /home/pi, ahí podemos visualizar los archivos que tenemos creados con el instrucción **ls**, y nos podemos cambiar a otros nodos por ejemplo **ssh nodo01** y revisar en la misma ruta /home/pi que archivos se encuentra ahí con el instrucción **ls**, efectivamente coincidirán, pero si creamos un archivo ahí mismo por ejemplo, crear el archivo **archivoprueba.txt**, y agregarle un mensaje para identificar que el archivo es único (con el instrucción vi), se guardará en ese nodo, pero también se compartirá con los demás nodos, figura 3.4. Entonces, si nos cambiamos a cualquier otro nodo y nos posicionamos en la misma ruta podemos revisar sus archivos con **ls** y efectivamente ahí estará el archivo **archivoprueba.txt** que acabamos de crear con el mensaje de texto que le agregamos para identificar que el archivo es único, figura 3.5

```
pi@master:~ $ pwd
/home/pi
pi@master:~ $ ls
 2022-04-23-034805_656x416_scrot.png   Bookshelf        mpi_hola_
 2022-04-23-034817_656x416_scrot.png   calcula_PI.cpp   mpi_hola_
 2022-04-23-035232_656x416_scrot.png   config.log       Music
 2022-04-23-035357_656x416_scrot.png   config.system    openmpi
 2022-04-23-035358_656x416_scrot.png   Desktop          Pictures
 2022-04-23-035402_656x416_scrot.png   dhcpd.conf       Public
 2022-04-23-035413_656x416_scrot.png   Documents        raspi-con
 2022-04-23-035414_656x416_scrot.png   Downloads        Templates
 2022-04-25-223556_656x416_scrot.png   helloWorld       Videos
 2022-04-25-223557_656x416_scrot.png   hello-world.c    wifi_ente
 3_Calculo_de_PI.cpp                   infinity.sh      wifi_pass
 activar_ssh.png                       machinefile     'Wireless
 actualizacion_paquetes.png            mpi-build
 apuntes.txt                           mpich
pi@master:~ $ ssh  nodo01
/home/pi
pi@nodo01:~ $ ls
 2022-04-23-034805_656x416_scrot.png   Bookshelf        mpi_hola_
 2022-04-23-034817_656x416_scrot.png   calcula_PI.cpp   mpi_hola_
 2022-04-23-035232_656x416_scrot.png   config.log       Music
 2022-04-23-035357_656x416_scrot.png   config.system    openmpi
 2022-04-23-035358_656x416_scrot.png   Desktop          Pictures
 2022-04-23-035402_656x416_scrot.png   dhcpd.conf       Public
 2022-04-23-035413_656x416_scrot.png   Documents        raspi-con
 2022-04-23-035414_656x416_scrot.png   Downloads        Templates
 2022-04-25-223556_656x416_scrot.png   helloWorld       Videos
 2022-04-25-223557_656x416_scrot.png   hello-world.c    wifi_ente
 3_Calculo_de_PI.cpp                   infinity.sh      wifi_pass
 activar_ssh.png                       machinefile     'Wireless
 actualizacion_paquetes.png            mpi-build
 apuntes.txt                           mpich
pi@nodo01:~ $ vi archivoprueba.txt
pi@nodo01:~ $ cat archivoprueba.txt
este es un mensaje de prueba de NFS
```

Figura 3.4: prueba nfs parte1.

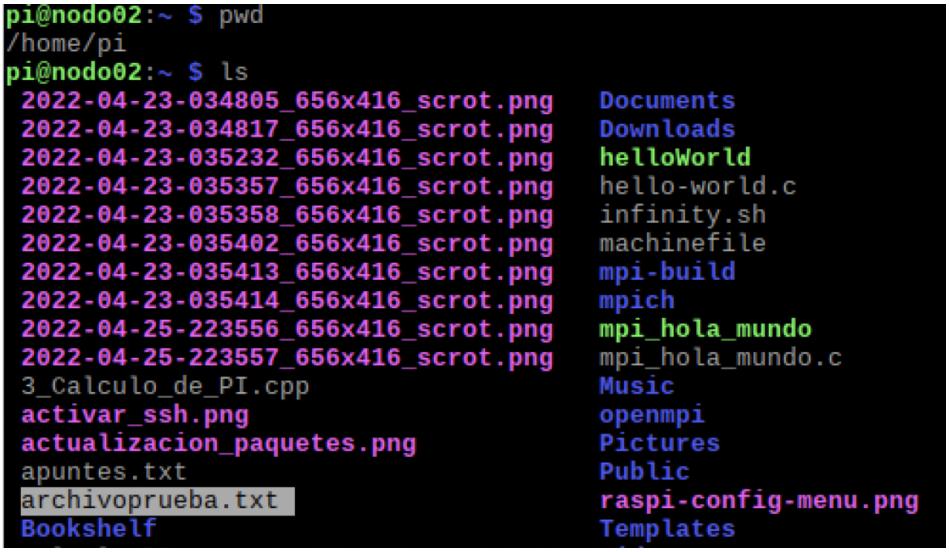


Figura 3.5: prueba nfs parte2.

3.4.3. HABILITAR EL DHCP EN LOS NODOS

Se necesita instalar un servidor DHCP que permita que la configuración IP de los nodos se haga de forma automática, evitando así la necesidad de tener que realizar manualmente uno por uno la configuración de la interfaz de red, como se describe en el siguiente script:

```
#Instalar dhcp server
sudo apt-get install isc-dhcp-server -y
sudo cat /dev/null > /etc/dhcpcd.conf
#Para borrar las líneas de nano se usa CTRL + K
sudo cat <<EOF>> /etc/dhcpcd.conf
option domain-name "cluster.rpi";
option domain-name-servers 8.8.8.8,8.8.4.4;
default-lease-time 600;
max-lease-time 7200;
ddns-update-style none;
authoritative;

subnet 192.168.1.0 netmask 255.255.255.0 {
        range 192.168.1.2 192.168.1.254;
        option subnet-mask 255.255.255.0;
        option routers 192.168.1.254;
        option broadcast-address 192.168.1.255;

        host nodo01 {
        hardware ethernet b8:27:eb:de:6a:7d;
        fixed-address 192.168.1.2;
        option host-name "nodo01";
        }
        host nodo02 {
        hardware ethernet  b8:27:eb:63:ef:48;
        fixed-address 192.168.1.3;
        option host-name "nodo02";
        }
        host nodo03 {
        hardware ethernet b8:27:eb:6b:03:43;
        fixed-address 192.168.1.4;
        option host-name "nodo03";
        }
        host nodo04 {
        hardware ethernet b8:27:eb:98:07:1b;
        fixed-address 192.168.1.5;
        option host-name "nodo04";
        }
        }

#Configuración de ips estáticas para cada nodo
interface eth0
static ip_address=192.168.1.5
static routers=192.168.1.0
static domain-name ''cluster.rp''
static domain_name_servers=192.168.1.1 8.8.8.8

interface wlan0
static ip_address=192.168.1.5
static routers=192.168.1.0
option domain-name ''cluster.rpi"
static domain_name_servers=192.168.1.1 8.8.8.8
EOF
```

En el script anterior de configuración de DHCP se describe en la parte superior los parámetros generales por defecto, y los parámetros adicionales que se proporcionarán a los nodos workers.

- **Subnet**: Especifica el rango de direcciones IPs que serán cedidas a los nodos que lo soliciten.
- **Host**: Especificaciones concretas de equipos. En la parte principal se puede configurar los siguientes parámetros, que más tarde se podrá reescribir en las distintas secciones:
- **max-lease-time**: se utiliza para indicar el máximo tiempo de concesión de una IP.
- **default-lease-time**: Se utiliza para indicar el tiempo de concesión de una IP cuando el cliente en su solicitud no indica ningún tiempo.
- **option domain-name-server**: Se emplea para enviar al cliente los servidores DNS que utilizará.
- **option domain-name**: Nombre del dominio que se manda al cliente.
- **option subnetmask**: Se utiliza para especificar la máscara de subred que se enviará al cliente, si se omitiera se enviaría la máscara que va asociada con la declaración subnet a la que pertenezca la IP asignada.
- **option routers**: Se usa para enviarle al cliente su puerta de enlace. Puede especificarse una IP numérica o un nombre.
- **option broadcast-address**: Dirección de difusión de la red. Al indicar una sección subnet se debe indicar la dirección de la red, la máscara de red, y entre llaves se puede poner los siguientes parámetros:
- **range**: Se utiliza para especificar el rango de IP que una subred tiene disponible para servir a sus clientes. En una declaración subnet pueden ponerse una o más declaraciones range y todas las direcciones IP especificadas deben pertenecer a la subnet.

  Dentro de la sección host, se pueden generar reservas, lo que permite que a un host siempre se le suministre la misma dirección de IP. Además de especificar el nombre que identifica al host, pueden definirse los siguientes parámetros:

  - **hardware ethernet**: Se utiliza para especificar dentro de la declaración host la dirección física que se utilizará para asociar esta declaración a un cliente. En el caso de las redes ethernet la direcciones físicas se denominan direcciones MAC.
  - **fixed-address**: la dirección IP que se va a asignar.
  - **option host-name**: el nombre del nodo que se va a asignar.

Antes de reiniciar el servidor y que comience a dar servicio, hay que especificar por cual de las interfaces de red va a trabajar. Se debe editar el archivo /etc/default/isc-dhcp-server, en el cual se modifica el parámetro INTERFACESv4 asignándole el nombre de interfaz (por ejemplo, INTERFACESv4=”eth0"). A continuación, se debe reiniciar el servicio ejecutando la instrucción **systemctl restart isc-dhcp-server.service** para que el mismo tome los cambios realizados.

```
pi@master:~ $ ssh-copy-id nodo01
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/pi/.s
rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to f
out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are
ed now it is to install the new keys
pi@nodo01's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'nodo01'"
and check to make sure that only the key(s) you wanted were added.

pi@master:~ $
```

Figura 3.6: Instrucción para copiar las llaves a los nodos.

Para copiar la clave pública del nodo maestro a todos los nodos trabajadores se debe ejecutar la siguiente instrucción **ssh-copy-id nodo01**, de esa manera se copiará la llave pública a todos los nodos configurados en el host. Es decir, ya no será necesario copiar la llave a cada uno de los nodos, incluso se podrán conectar entre ellos con la misma llave, como se visualiza en la Figura 3.6:

3.5. PRUEBAS EN EL CLUSTER

Una vez encendidos todos los nodos, se procede a verificar el correcto desempeño del cluster. Para esto se realizó el siguiente programa llamado “hola mundo", que implementa la biblioteca OpenMPI.

```
nano mpi_hola_mundo.c
#include <mpi.h>
#include <stdio.h>
int main(int argc, char** argv){
//Inicializar el entorno MPI
MPI_Init (NULL, NULL);
//Obtener la cantidad de procesadores
int world_size;
MPI_Comm_size (MPI_COMM_WORLD, &world_size);
// Obtener el rank del proceso
int world_rank;
MPI_Comm_rank (MPI_COMM_WORLD, &world_rank);
//Obtener el nombre del procesador
//processor_name [MPI_MAX_PROCESSOR_NAME];
int name_len;
MPI_Get_processor_name (processor_name, &name_len);
//Imprimir el mensaje de Hola mundo
printf("Hola mundo desde el procesador %s, rank %d de %d procesadores \n",
processor_name, world_rank, world_ size);
//Finalizar el entorno MPI.
MPI_Finalize();
}

#include <mpi.h>
#include <stdio.h>
int main(int argc, char** argv){
//Inicializar el entorno MPI
MPI_Init (NULL, NULL);
//Obtener la cantidad de procesadores
int world_size;
MPI_Comm_size (MPI_COMM_WORLD, &world_size);
// Obtener el rank del proceso
int world_rank;
MPI_Comm_rank (MPI_COMM_WORLD, &world_rank);
//Obtener el nombre del procesador
//MPI_MAX_PROCESSOR_NAME (MPI_COMM_WORLD, &processor_name);
//processor_name[MPI_MAX_PROCESSOR_NAME];
char processor_name[MPI_MAX_PROCESSOR_NAME];
int name_len;
MPI_Get_processor_name (processor_name, &name_len);
//Imprimir el mensaje de Hola mundo
printf("Hola mundo desde el procesador %s, rank %d de %d procesadores \n",
processor_name, world_rank, world_size);
//Finalizar el entorno MPI.
MPI_Finalize();
}
```

Se debe ejecutar el siguiente instrucción para compilar los archivos fuentes de la aplicación de prueba:

```
mpicc mpi_hola_mundo.c -o mpi_hola_mundo
```

La descripción de la instrucción anterior sería:

```
mpicc <codigo_fuente.c> -o <ejecutable>
```

Donde la instrucción mpicc es un pequeño script de compilación para C.

Posteriormente debemos crear el archivo 'machinefile' (puede tener cualquier nombre, no necesariamente debe llamarse machinefile), nos servirá para hacer referencia a los núcleos de cada uno de los nodos registrados en el host, por eso es necesario guardar los nombres correctos de cada nodo y el número de procesadores que tendrán disponibles para trabajar, en ese caso máximo son 4 por cada nodo, (si no tiene el hostname exacto de algún nodo marcará error por no reconocerlo).

```
nano machinefile
nodo01 slots=4
nodo02 slots=4
nodo03 slots=4
nodo04 slots=4
```

El siguiente paso es ejecutar la aplicación, para eso se utiliza la instrucción mpirun, de la siguiente manera:

```
mpirun -np 16 --hostfile machinefile mpi_hola_mundo
```

La descripción de la instrucción anterior es la siguiente:

mpirun: instrucción de ejecución

-np 16: número de procesadores que deseamos usar para el trabajo en este caso máximo son 16 porque son 4 por cada nodo trabajadores que tengamos disponible.

–hostfile machinefile: nombre del archivo que se usará de referencia para identificar los nodos y sus procesadores.

mpi_hola_mundo: nombre del ejecutable compilado.

3.6. PRUEBAS DE EJECUCIÓN, CÁLCULO DE PI

Primero se debe de crear el algoritmo que calcule el número pi, para eso se usará el lenguaje C++ y se hará uso de la biblioteca MPI. El algoritmo se visualiza a continuación y se describe desde sus anotaciones a nivel de código [Cal].

```
nano 3_Calculo_de_PI.cpp
```

```
#include <math.h>
#include <mpi.h>   // Biblioteca de MPI
#include <cstdlib> // Incluido para el uso de atoi
#include <iostream>
using namespace std;

int main(int argc, char *argv[])
{
    int n, // Numero de iteraciones
```

```
10          rank, // Identificador de proceso
11          size; // Numero de procesos
12      double PI25DT = 3.14159265358979323846264643;
13      double mypi, // Valor local de PI
14          pi,   // Valor globarl de PI
15          h,   // Aproximación del área para el calculo de PI
16          sum;  // Acumulador para la suma del área de PI
17          // Comprueba si hay valores por parametros
18      bool valor_por_parametros = true;
19
20
21      MPI_Init(&argc, &argv); // Inicializamos los procesos
22      MPI_Comm_size(MPI_COMM_WORLD, &size); // Obtenemos el numero total de procesos
23      MPI_Comm_rank(MPI_COMM_WORLD, &rank); // Obtenemos el valor de nuestro identificador
24
25      // Solo el proceso 0 va a conocer el numero de iteraciones que vamos a
26      // ejecutar para la aproximación de PI
27          if (rank == 0) {
28                  cout<<"introduce la precisión del calculo (n > 0): ";
29                  cin>>n;
30          }
31
32          // El proceso 0 reparte al resto de procesos el numero de iteraciones
33          // que calcularemos para la aproximación de PI
34          MPI_Bcast(&n, // Puntero al dato que vamos a enviar
35                                                  1,  // Numero de datos a los que apunta el puntero
36                                                  MPI_INT, // Tipo del dato a enviar
37                                                  0, // Identificación del proceso que envia el dato
38                                                  MPI_COMM_WORLD);
39          if (n <= 0){
40                  MPI_Finalize();
41                  exit(0);
42          }else {
43                  // Calculo de PI
44                  h = 1.0 / (double) n;
45                  sum = 0.0;
46                  for (int i = rank + 1; i <= n; i += size) {
47                          double x = h * ((double)i - 0.5);
48                          sum += (4.0 / (1.0 + x*x));
49                  }
50                  mypi = h * sum;
51
52                  // Todos los procesos ahora comparten su valor local de PI,
53                  // lo hacen reduciendo su valor local a un proceso
54                  // seleccionada a través de una operación aritmética.
55                  MPI_Reduce(&mypi, // Valor local de PI
56                                          &pi,  // Dato sobre el que vamos a reducir el resto
57                                          1,          // Numero de datos que vamos a reducir
58                                          MPI_DOUBLE,  // Tipo de dato que vamos a reducir
59                                          MPI_SUM,  // Operación que aplicaremos
60                                          0, // proceso que va a recibir el dato reducido
61                                          MPI_COMM_WORLD);
62
63                  // Solo el proceso 0 imprime el mensaje, ya que es la única que
64
65                  // conoce el valor de PI aproximado.\\
66                  if (rank == 0)
67
68          cout << "El valor aproximado de PI es:
```

```
69          << pi << ", con un error de "
70          << fabs(pi - PI25DT) << endl;
71          }
72
73          // Terminamos la ejecución de los procesos, después de esto solo existirá
74          // el proceso 0
75          // ¡Ojo! Esto no significa que los demás procesos no ejecuten el resto
76          // de código después de "Finalice", es conveniente asegurarnos con una
77          // condición si vamos a ejecutar mas código (Por ejemplo, con "if(rank==0)".
78      MPI_Finalize();
79      return 0;
80  }
81  }
```

Para compilar se usa la siguiente instrucción:

mpiCC 3_Calculo_de_PI.cpp -o PI

Para ejecutar el compilado se usa la instrucción: **mpirun –hostfile machinefile -np 16 PI -c** La descripción de las instrucciones de compilación y ejecución se describió en el ejemplo anterior del "hola mundo".

Como resultado de la ejecución del algoritmo para calcular el número PI, el programa muestra una sencilla interfaz en consola para ingresar el índice de operaciones para calcular el número PI. Entre más alto sea el índice de operaciones, con mayor exactitud se obtendrá el número 3.1416, y con una probabilidad de error más baja.

3.6.1. Pasos complementarios de configuración

Pasos para cambiar el idioma del teclado

1. Ingresar al clúster con:

   usuario: **pi**
   contraseña: **raspberry**

2. Escribir la instrucción **"sudo raspi-config"**, para obtener el "-" se obtiene con la tecla de "?" y se abrirá un menú como el de la Figura 3.7

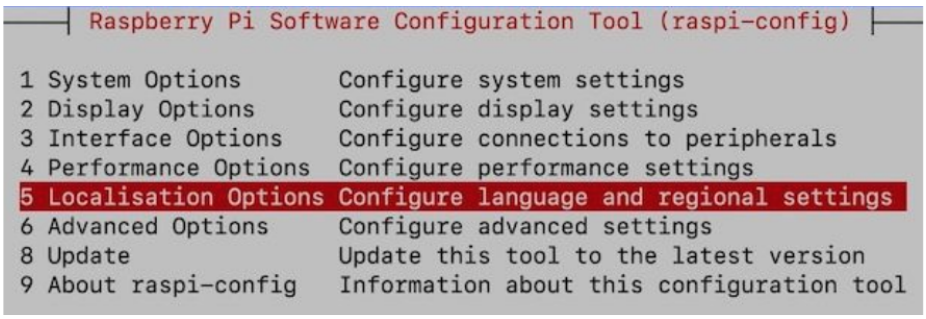


Figura 3.7: Menú para configurar lenguaje.

3. Seleccionar **Localisation Options** ->**L1 Locale** y el siguiente menú que se muestra seleccionar la opción de **"Change locale"** como en la Figura 3.8 y al seleccionar la opción se mostrará otro menú.

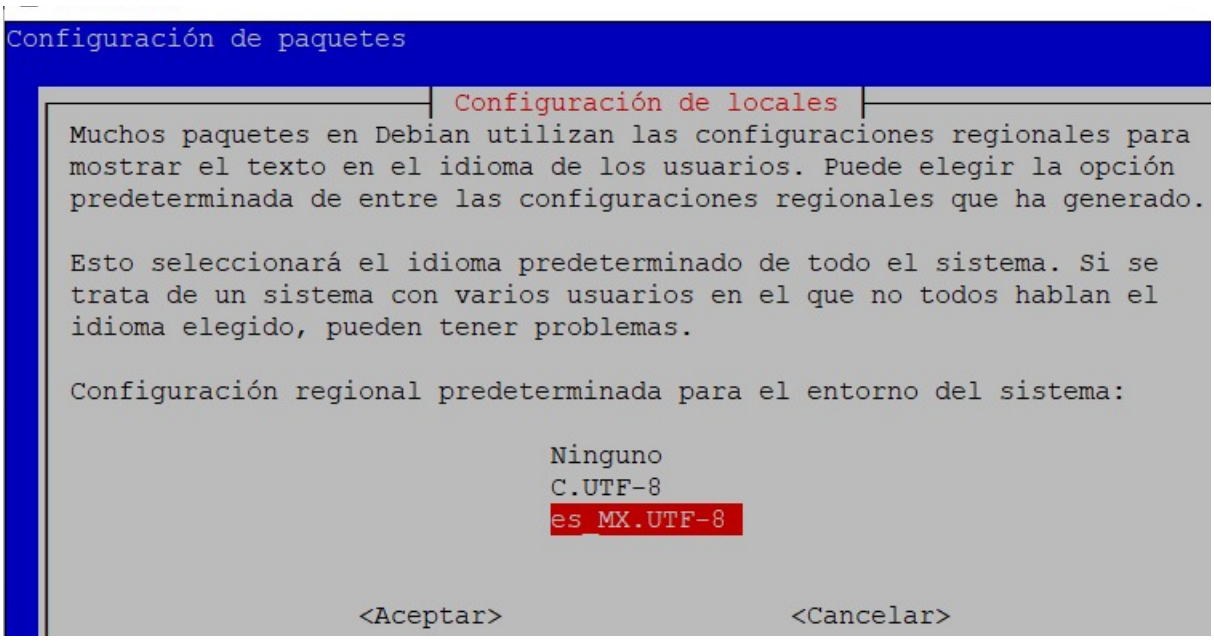


Figura 3.9: Menú para escoger el lenguaje por defecto a español UTF-8.

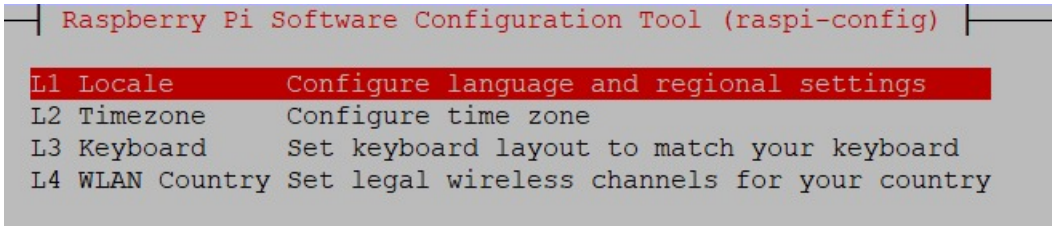


Figura 3.8: Menú Change Locale.

4. En el menú se visualizan las opciones de idiomas, debemos buscar la opción de es_Mx.UTF-8.
5. Posteriormente seleccionar como idioma por defecto a es_Mx.UTF-8 como en la Figura 3.9
6. Volvemos a ejecutar el instrucción sudo **“raspi-config"** ->**"Localisation Options”** para cambiar la zona horaria al país donde nos encontremos, en nuestro caso América ->México, como en la Figura 3.10.

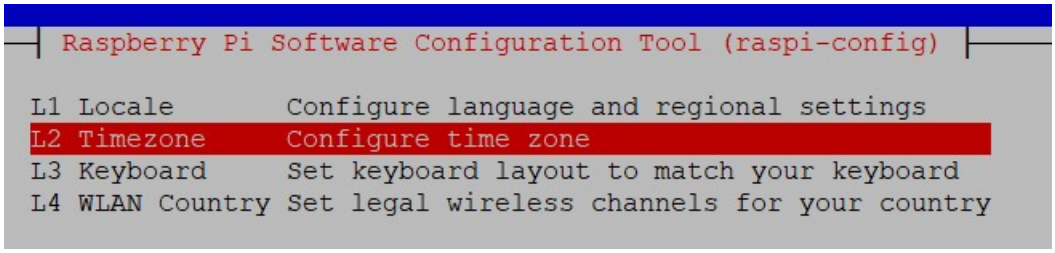


Figura 3.10: Menú para cambiar la zona horaria.

Capítulo 4

Automatización y Monitoreo

4.1. Automatización

La característica de que nuestro clúster sea escalable, nos permite agregar nuevos nodos para tener mayor poder de procesamiento.
Algunas de las problemáticas que encontramos al configurar el clúster es que en ocasiones era necesario ejecutar la misma instrucción en todos los nodos, ya sea para instalar alguna herramienta, compilar o ejecutar algún programa, o simplemente actualizar los paquetes de cada nodo; también el hecho de tener que conectar a un teclado y monitor las placas nuevas que agregáramos a nuestro clúster para poder configurarlas resultaba ser un poco tedioso. Pero encontramos algunas herramientas y configuraciones muy útiles que nos ayudaron a solventar estos inconvenientes.

4.1.1. Manejo de varias sesiones SSH de forma concurrente: ClusterSSH (cssh)

Lo que hace esta herramienta es que podemos abrir n terminales por cada uno de los servidores o nodos de nuestro clúster y todo lo que escribamos en la consola principal será replicado en todas ellas. O bien, podemos tener diferentes configuraciones para que los comandos de la consola principal se ejecuten solamente en ciertos nodos, incluso podríamos crear subgrupos de nodos para tener diferentes clústers y ejecutar las instrucciones deseadas en cada grupo.

Para utilizar esta herramienta al igual que con SSH, se necesitan un par de claves pública/privada sin passphrase de tal manera que no se nos solicite ninguna contraseña durante el acceso a los nodos de nuestro clúster.

Esta configuración y generación de claves sería la misma que revisamos en el capítulo 2.

El primer paso que hicimos fue instalar la herramienta ClusterSSH en nuestro nodo maestro con la siguiente instrucción:

```
$ sudo apt-get install clusterssh
```

Una vez instalado, si queremos abrir varias sesiones de forma concurrente, ejecutamos la siguiente instrucción:

```
$ cssh nodo01 nodo02 nodo03 ... nodoN
```

Posteriormente, programa nos mostrará una consola principal(CSSH) en la que ejecutaremos el mismo comando en los nodos que hayamos indicado y también se mostrarán N sesiones dependiendo de los nodos en los que queramos ejecutar las instrucciones como se muestra en la Figura 4.1

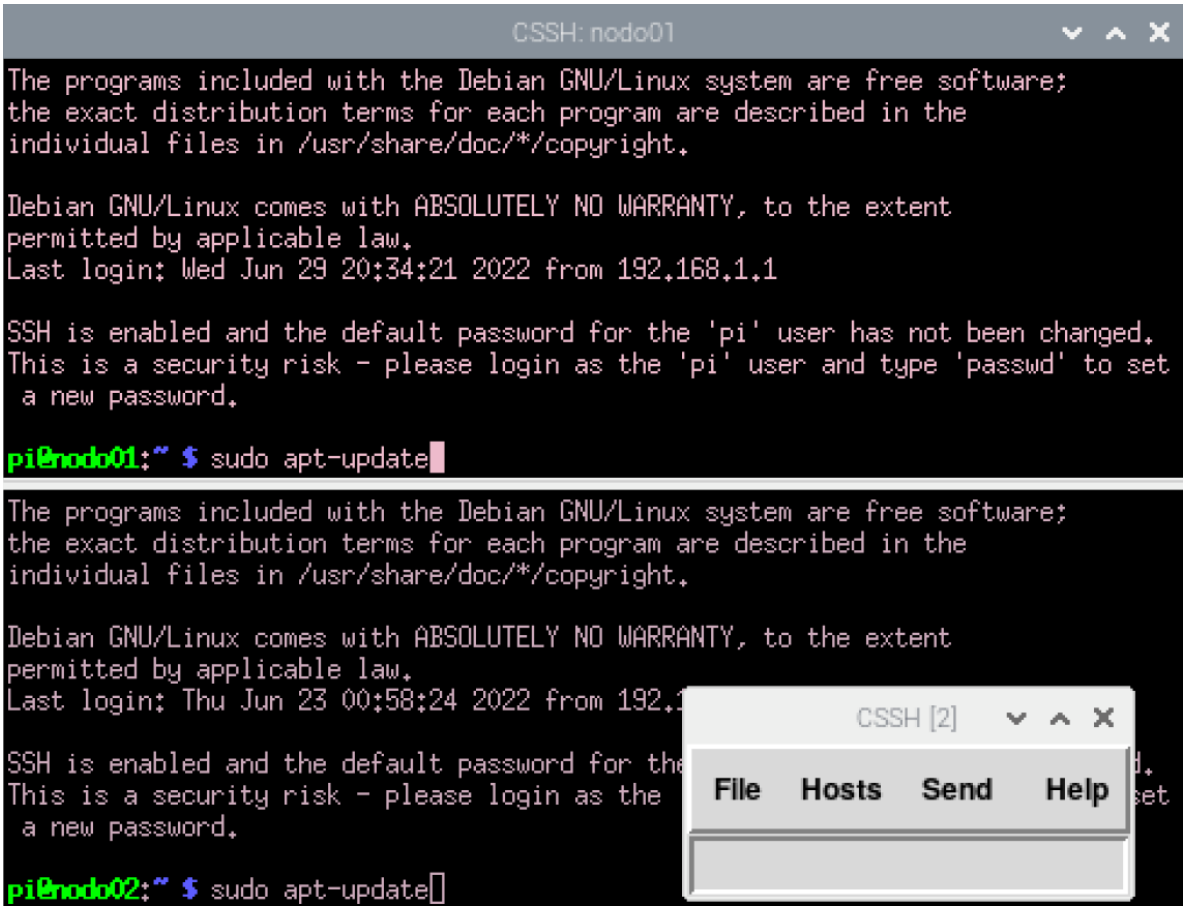


Figura 4.1: Uso de CSSH.

Para hacer más fácil el llamado de la instrucción cssh, definimos un grupo de IPs o hostnames de nuestro clúster para no tener que ingresarlos todos. Esto se logró configurar con la siguiente instrucción:

```
$ cluster1 nodo01 nodo02 nodo03 nodo04
```

Ahora sólo bastaría escribir:

```
cssh cluster1
```

y automáticamente se nos abrirá una terminal por cada una de las IPs o hostmanes definidos. Y por último si queremos escribir alguna instrucción simultáneamente en todas las sesiones, debemos escribir igual en el recuadro de la consola principal CSSH como se muestra en la Figura 4.2

4.1.2. Habilitar SSH sin necesidad de teclado o monitor

El primer paso es habilitar el uso de SSH para poder conectarnos al nuevo nodo desde nuestra computadora personal o PC.
Como medida de seguridad SSH viene desactivado por defecto en la instalación de Raspberry Pi OS. Sin embargo, podemos habilitarlo simplemente **creando un archivo llamado 'ssh', sin extensión, dentro de la tarjeta SD** que hemos preparado con la instalación del sistema operativo, como lo vimos en capítulos anteriores.
Una vez creado el archivo ya estará activada la conexión SSH para poder conectarnos al nuevo nodo.

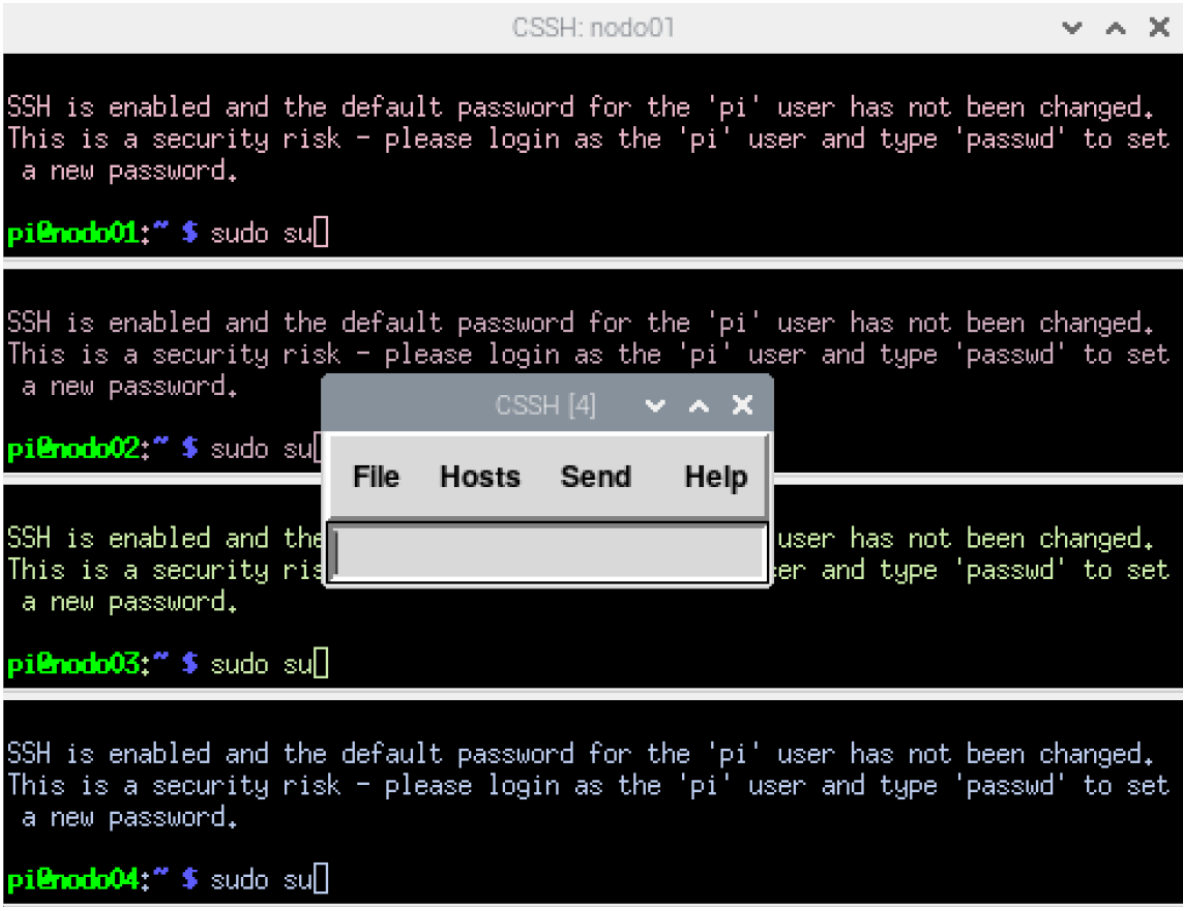


Figura 4.2: Uso de CSSH con grupos.

4.1.3. Configurar WIFI sin necesidad de teclado o monitor

Si vamos a conectar la Raspberry Pi por WiFi deberemos configurar el archivo **'wpa_supplicant.conf'** como vimos en la configuración básica de capítulos anteriores.

De forma similar al SSH, podemos añadir nuestra configuración WiFi simplemente creando el archivo 'wpa_supplicant.conf' en la tarjeta SD que hemos preparado con la instalación del Sistema Operativo. Al arrancar, el sistema operativo tomará el archivo de configuración del WiFi de la tarjeta SD y se conectará a la red con las credenciales especificadas en el mismo.

El formato del archivo 'wpa_supplicant.conf' tiene la siguiente forma:

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=MX

network={
        ssid="nombre-de-tu-wifi"
        psk="password-de-tu-wifi"
        key_mgmt=WPA-PSK
}
```

En el archivo de configuración anterior se indican los parámetros se describen a continuación:

- **ssid** : Es para especificar el nombre de la red de wifi.
- **psk** : Es para registrar el password de la red de wifi.

4.2. Monitoreo

Como ya se ha explicado anteriormente, un clúster puede tener diferentes usos o aplicaciones que en su mayoría requieren una alta demanda de procesamiento y recursos. Después de realizar algunas pruebas con el clúster compilando y ejecutando varios algoritmos, tuvimos la curiosidad por saber cómo funcionan los nodos. Cómo funcionan bajo carga, si tienen poca potencia o si la temperatura de la CPU supera los niveles deseados. ¿Cuánta memoria te queda en cada uno de los nodos mientras realizan tareas? Cuantos más nodos tengamos en nuestro rack, más información tendremos que revisar. Afortunadamente, existen varias herramientas, de las cuales elegimos un software llamado Glances, que nos ayudó a visualizar toda esa información.

4.2.1. Glances como herramienta de monitoreo

El software que utilizamos para integrar y monitorear los nodos de nuestro clúster se conoce como middleware, específicamente utilizamos uno llamado glances y a continuación se detallará paso a paso cómo realizamos su instalación, configuración y utilización para visualizar la información en una página web local [Gla].

4.2.2. Instalación de Glances

Este software libre está basado en python. Hay varias formas de instalarlo. La mayoría de las distribuciones de Linux ya lo tienen en sus repositorios, por lo que en nuestro S.O. Raspberry Pi OS. sólo ejecutamos la siguiente instrucción para instalar glances:

```
$ sudo apt install glances
```

Con la instrucción anterior, todas las dependencias del paquete se resolverían, pero también podríamos hacerlo a la manera de Python, usando pip como se muestra en la siguiente instrucción:

```
$ sudo apt install pip
$ pip install glances
```

Diferencia entre apt install y pip install

PyPI es el índice de Paquetes python — repositorio de módulos de python.
pip se utiliza para descargar e instalar los paquetes directamente desde PyPI. PyPI es alojado por la Python software Foundation. Es un gestor de paquetes especializado que se ocupa únicamente de los paquetes de python.

apt se utiliza para descargar e instalar paquetes desde los repositorios de ubuntu.

Ambos apt y pip son maduros, los gestores de paquetes que instala automáticamente cualquier otra dependencia de los paquetes durante la instalación. Podemos usar cualquiera de las dos instrucciones. Sin embargo, si necesitamos instalar una versión particular de python, o instalar el paquete en un ambiente virtual, o instalar un paquete que sólo es alojado en PyPI, lo recomendable sería realizar la instalación con pip. De lo contrario, si no son relevantes estas cuestiones para nuestro proyecto, podemos utilizar apt para la instalación, debido a que trae la última versión y las dependencias de los paquetes que deseemos instalar.

Glances tiene varios módulos como información RAID (Redundant Array of Independent Disks, por sus siglas en inglés), información de red WiFi, interfaz web o integración Docker. Para asegurarse de obtenerlos todos, si usamos pip, instalaremos glances con la siguiente instrucción:

```
$ pip install 'glances[action,browser,cloud,cpuinfo,docker,export,folders,
gpu,graph,ip,raid,snmp,web,wifi]'
```

Finalmente, instalamos glances en todos los nodos del clúster.

4.2.3. Uso de glances

Hay varias formas de utilizar éste software. Si deseamos ver información sobre un nodo del clúster en particular y nos conectamos a él a través de SSH, simplemente ejecutamos la siguiente instrucción:

```
$ glances
```

Si deseamos iniciar glances en modo servidor en un nodo del clúster, debemos agregar el atributo -s

```
$ glances -s
```

Si deseamos iniciar glances y monitorear los nodos del clúster desde otra máquina que esté en la misma red, podemos iniciar glances como una aplicación cliente simple y conectarnos al servidor glances con el que acabamos de iniciar con la siguiente instrucción:

```
$ glances -c 192.168.x.x
```

dónde 192.168.x.x es la IP del servidor glances. Otra forma de conectarse es poniendo el hostname que ya configuramos previamente en capítulos anteriores.

```
$ glances -c nodo01
```

También definimos nuestros nodos del clúster en el archivo de configuración glances.conf ubicándonos en la sección Client/server

```
[servidores]
# Define the static servers list
server_1_name = 192.168.1.1
server_1_alias = master
server_1_port = 61209
server_2_name = 192.168.1.2
server_2_alias = nodo01
server_2_port = 61209
server_3_name = 192.168.1.3
server_3_alias = nodo02
server_3_port = 61209
server_4_name = 192.168.1.4
server_4_alias = nodo03
servidor_4_puerto=61209
```

Como podemos ver, tenemos un número, un nombre, un alias y un puerto. El puerto debe permanecer el mismo número establecido para todos los nodos, mientras que el alias debe corresponder al nombre de host de cada IP de los nodos del clúster. Al usar múltiples nodos, para evitar ingresar la contraseña de inicio de sesión cada vez que nos conectabamos a un servidor glances, definimos esa contraseña en el archivo de configuración, lo editamos en la ruta: **/etc/glances/glances.conf** y agregamos las contraseñas de nuestros nodos en la sección **[passwords]**

```
[passwords]
# Define the passwords list
nodo01=password
nodo02=password
nodo03=password
nodo04=password
```

Ahora podemos llamar glances desde una pc o computadora portátil conectada a nuestra red local, con la siguiente instrucción:

```
$ glances --browser
```

y vemos todos los nodos de su clúster en una lista navegable. Podemos ingresar a cada uno de ellos, ver qué está haciendo cada uno con sus recursos, procesos y luego pasar al siguiente.

4.2.4. Creación de una página web para monitorear todos los nodos

Para tener una mejor visión y poder monitorear al mismo tiempo todos los nodos de nuestro clúster, nos dimos a la tarea de crear una página web que funciona como estación de monitoreo. Como primer paso se instaló Apache en el nodo maestro, utilizando la siguiente instrucción:

```
$ sudo apt install apache2
```

La raíz del servidor web de apache se encuentra en el siguiente directorio: /var/www/html. En éste directorio, creamos un archivo llamado index.html con la siguiente instrucción:

```
# touch /var/www/html/index.html
```

Con el comando **touch** de linux, creamos un archivo vacío con extensión .html, el cual procedimos a editar utilizando el comando **nano**:

```
# nano /var/www/html/index.html
```

Por último agregamos el siguiente código HTML con la información de nuestros nodos [Vér09]:

```
<!DOCTYPE html>
<html>
<meta charset="UTF-8" />
<head>
<title>Cluster UACM</title>
</head>
<body>
<center>
<h1>Informacion de cluster</h1>
</center>
<br>
<center>
<iframe src="http://192.168.1.1:61208" height="465" width="280"></iframe>
<iframe src="http://192.168.1.2:61208" height="465" width="280"></iframe>
<iframe src="http://192.168.1.3:61208" height="465" width="280"></iframe>
<iframe src="http://192.168.1.4:61208" height="465" width="280"></iframe>
<iframe src="http://192.168.1.5:61208" height="465" width="280"></iframe>
</center>
</br>
</body>
</html>
```

Guardamos nuestro archivo y antes de abrirlo ejecutamos la instrucción: **glances -w** en todos los nodos para iniciar los servidores web de monitoreo en cada nodo del clúster como se muestra en la figura 4.3. (para ejecutar esta instrucción más fácilmente en todos los nodos, utilizamos la herramienta cssh que vimos anteriormente).



Figura 4.3: glances-w con cssh.

Una vez hecho esto, podemos abrir nuestro archivo html o escribir localhost en nuestro navegador web (estando en nuestra red local) y tendremos nuestra estación de monitoreo, donde veremos todos los nodos, incluyendo el nodo maestro como se muestra en la Figura 4.4.

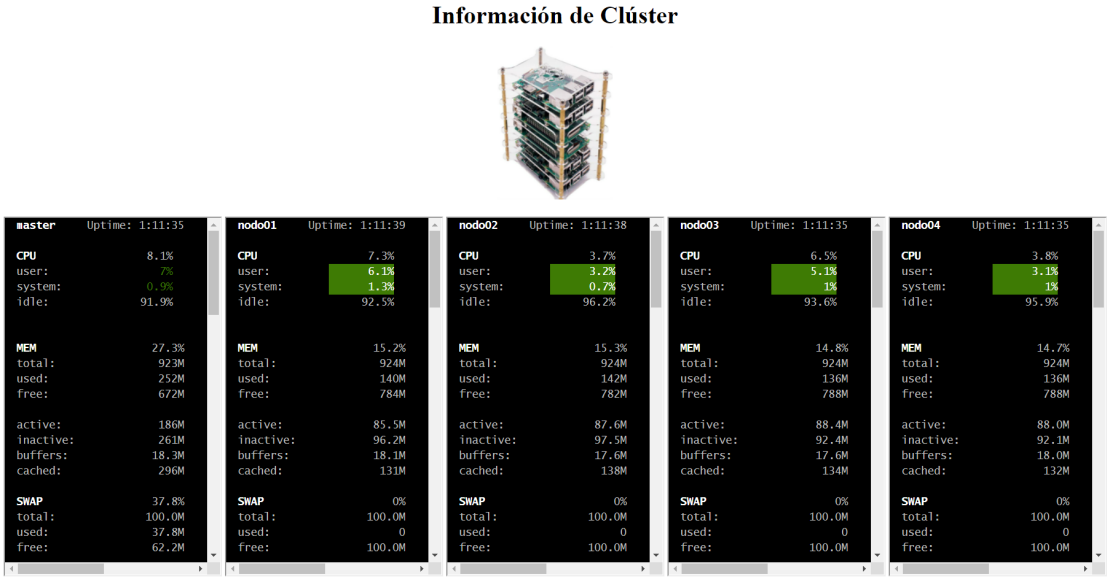


Figura 4.4: Dashboard de Monitoreo para todos los nodos del clúster.

A continuación mostramos un listado, describiendo las principales métricas que se visualizan en nuestro dashboard de monitoreo y el de la figura 4.5

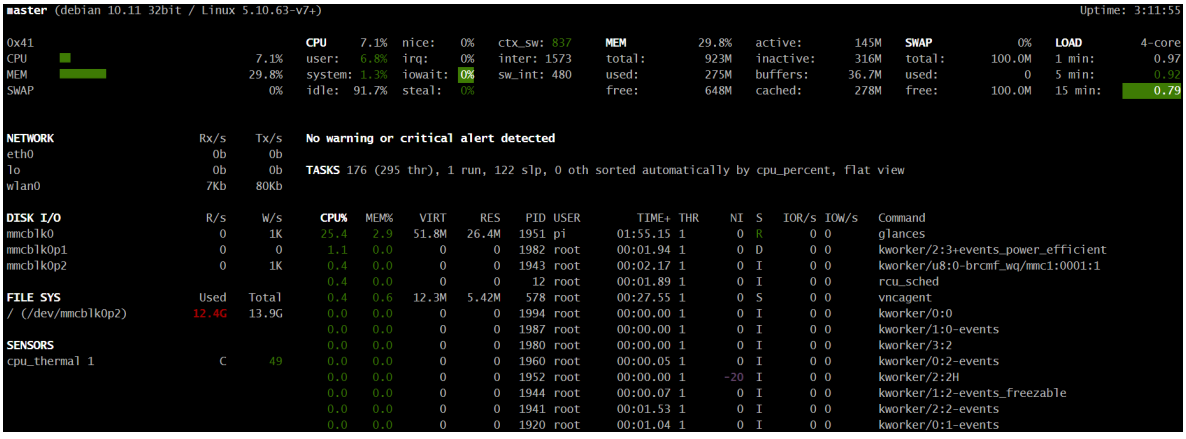


Figura 4.5: Métricas de monitoreo.

- **HEADER**: El encabezado muestra el nombre de host, el nombre del sistema operativo y el tiempo de actividad del sistema. Además, en GNU/Linux, también muestra la versión del kernel.
- **CPU**: Las estadísticas de la CPU se muestran como un porcentaje en tiempo real
  - **user**: Porcentaje de tiempo dedicado en el espacio del usuario. Es el tiempo dedicado en el procesador para correr los programas del usuario (o código de librerías).
  - **system**: Porcentaje de tiempo dedicado en el espacio del kernel. Es el tiempo dedicado corriendo código en el kernel del sistema operativo.

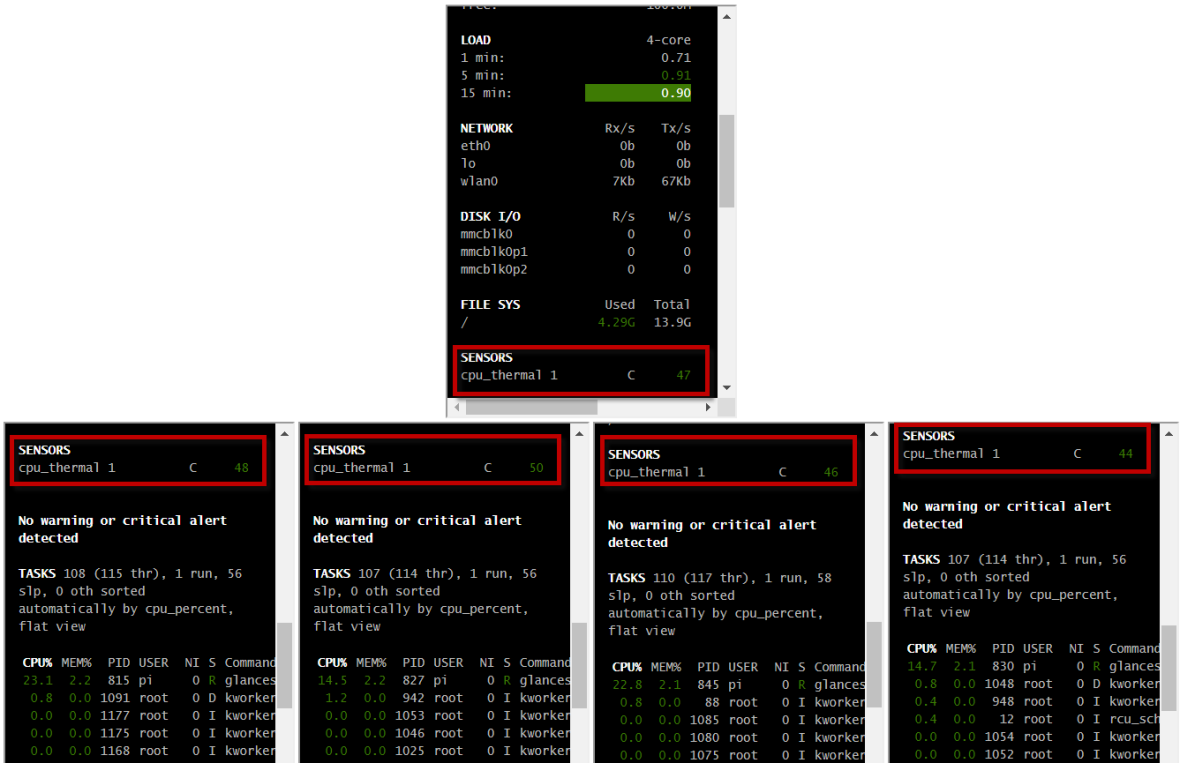
- **idle**: Porcentaje de CPU usado por cualquier programa. Cada programa o tarea que se ejecuta en un sistema informático ocupa una cierta cantidad de tiempo de procesamiento en la CPU. Si la CPU ha completado todas las tareas, está inactiva.

- **MEM**: Glances usa dos columnas: Una para la memoria RAM y otra para la memoria SWAP.
  - **percent**: Es el porcentaje usado, calculado como: (total-disponible) / total*100.
  - **total**: Memoria física total disponible
  - **used**: Memoria utilizada.
  - **free**: Memoria que no se usa en absoluto (puesta a cero).
  - **active**: Memoria actualmente en uso o muy recientemente usada en la memoria RAM.
  - **inactive**: Memoria que se marca como no utilizada.
  - **buffers**: Caché para cosas como metadatos del sistema de archivos.
  - **cached**:Memoria caché para varias cosas.
- **SWAP**: El Espacio swap en Linux es usado cuando la cantidad de memoria física (RAM) está llena. Si el sistema necesita más recursos de memoria y la memoria física está llena, las páginas inactivas de la memoria se mueven al espacio swap (o memoria de intercambio).
  - **total**: Memoria SWAP total (la unidad es bytes).
  - **used**: Memoria SWAP usada
  - **free**: Memoria SWAP libre
- **LOAD**:Promedio de carga en el sistema operativo.
  Es la suma promedio de la cantidad de procesos que esperan en la cola de ejecución más la cantidad que se está ejecutando actualmente durante períodos de tiempo de 1, 5 y 15 minutos
- **NETWORK**: Glances muestra la tasa de bits de la interfaz de red. La unidad se adapta dinámicamente (bit/s, kbit/s, Mbit/s, etc).
- **DISK I/O**: Glances muestra el rendimiento de Entradas /Salidas del disco. La unidad se adapta dinámicamente. Se pueden mostrar bytes por segundo (comportamiento predeterminado / Bytes/s, KBytes/s, MBytes/s, etc.) ó solicitudes por segundo.
- **FILE SYS**: Glances muestra el espacio en disco del sistema de archivos total y utilizado. La unidad se adapta dinámicamente.
- **SENSORS**: Glances puede mostrar la información de los sensores usando:
  - **psutil**: Temperaturas de la placa base y de la CPU
  - **hddtemp**: Temperatura del disco duro
  - **batinfo**: Capacidad de la batería
- **PROCESSES LIST**: Lista de procesos, en la cuál se muestran las siguientes características:
  - Número total de tareas/procesos
  - Número de hilos
  - Número de tareas/procesos en ejecución
  - Número de tareas/procesos durmientes
  - Otro número de tareas/procesos (no en estado de ejecución o de suspensión)
  - Clave de clasificación para la lista de procesos

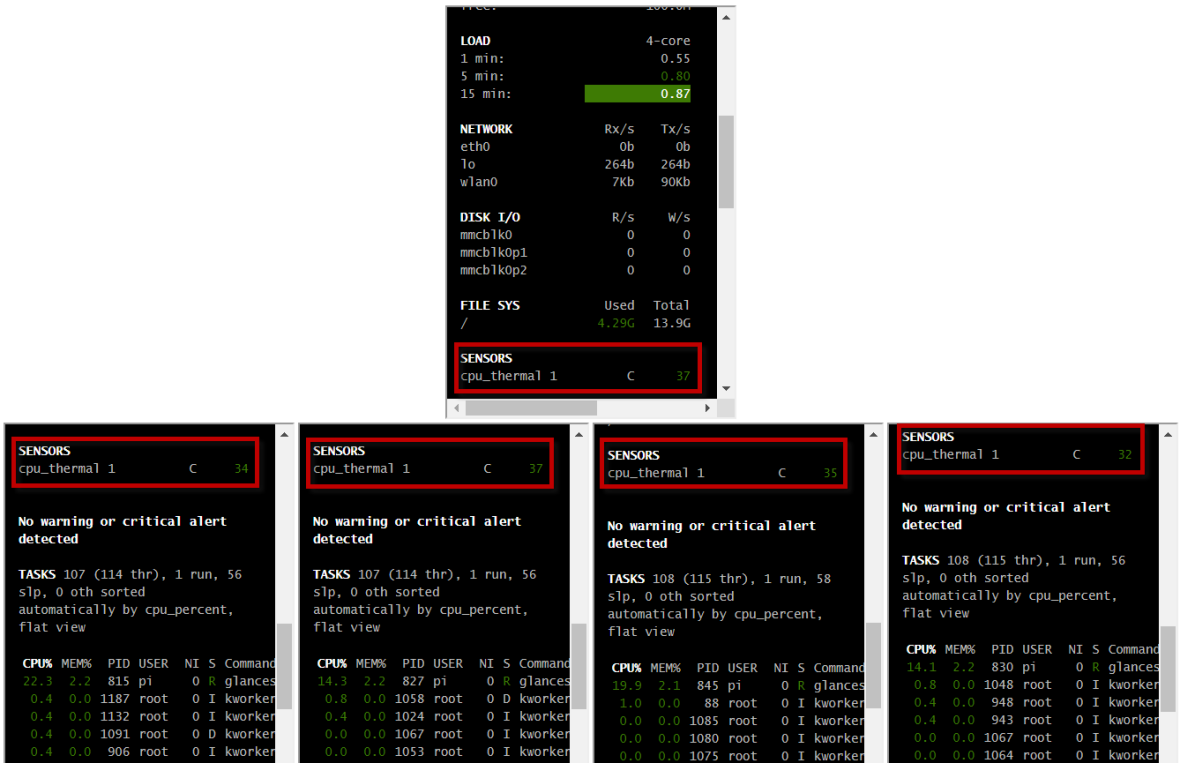
4.3. Pruebas para monitorear la temperatura de los nodos

Una buena forma de comprobar que nuestro dashboard de monitoreo está funcionando de forma correcta, fue realizar una prueba poniendo un pequeño ventialdor cerca de nuestro clúster para medir la temperatura de las placas y notar la variación de la temperatura.
Normalmente, nuestro clúster presenta una temperatura promedio de 48^{o}C pero al colocar el ventilador cerca de las placas Raspberry Pi para enfriarlas, la temperatura de cada placa puede disminuir en promedio de (5^{o}C - 10^{o}C), aunque existen otros sistemas de enfriamiento más eficientes.

Cabe mencionar que uno de los factores para que la temperatura pueda cambiar, es la cantidad de procesamiento que esté realizando cada nodo, es decir, entre más procesamiento la temperatura puede aumentar y entre menos procesamiento la temperatura disminuirá o se mantendrá [Ric11].



(a) Sin ventilador



(b) Con ventilador

Figura 4.6: Comparativa de temperatura del clúster

Capítulo 5

Dificultades y Conclusiones

5.1. Dificultades.

A lo largo del proceso de este proyecto, nos encontramos con varias dificultades tanto físicas para el armado del clúster, como de configuración.

5.1.1. Alimentación de energía

Inicialmente alimentábamos las tarjetas Raspberry Pi 3 por medio de cables v8-USB, pero todos estaban conectados a un switch que disminuye el voltaje y no permitía alimentar a 5 tarjetas al mismo tiempo, solo soportaba 3 tarjetas (el máster y 2 esclavos); por ello, tuvimos que alimentar a dos tarjetas de una fuente de alimentación independiente, y 3 directamente del switch, pero al final para facilitarnos las cosas cambiamos el switch por uno de mejor calidad y más potente.

5.1.2. Sobrecalentamiento

Al principio, pensamos que por el tipo de carcasa y topología física del clúster sería suficiente para evitar problemas de sobrecalentamiento, pero durante la marcha y después de que el clúster estuvo encendido varias horas continuas, el sistema operativo mostró un aviso de sobrecalentamiento, por lo que vimos la necesidad de adaptar un ventilador para enfriar todas las placas y poder continuar trabajando sin problemas de sobrecalentamiento.

5.1.3. Directorios de DHCP

Un problema que nos surgió fue la configuración de las direcciones de IP fijas en el archivo DHCP, debido a que había dos directorios parecidos donde se ubica este archivo: dhcpd.conf, hasta que detectamos la ruta y el archivo correcto era: /etc/dhcpcd.conf Esa dificultad surge porque consultamos diferentes fuentes de información donde en algunos utilizan el archivo /dhcpd.conf mientras que nosotros necesitamos el archivo /dhcpcd.conf, posiblemente por los diferentes sistemas operativos y versiones.

5.1.4. Llaves SSH

Inicialmente generamos las llaves SSH desde la configuración inicial para poder tener acceso a los nodos y por proceso ortodoxo copiamos las llaves públicas a los nodos con el comando **“scp archivo.txt usuario@dominio.com:/home/usuario/.ssh"**, pero al momento de montar el sistema de archivos NFS en los nodos, estas llaves no funcionaron para poder acceder a los nodos sin password, por lo que tuvimos que utilizar una instrucción especial **“ssh-copy-id"** nodo01 que resolvió el problema del password e incluso le permite a los nodos acceder entre ellos debido a que comparten el mismo directorio /home/pi/.ssh.

5.1.5. Programación paralela

Existen diversas dificultades que se pueden encontrar a la hora de escribir un programa paralelo. Por ejemplo, no siempre es posible paralelizar un programa; la paralelización requiere de tareas que no están presentes en la programación secuencial, (descomposición del problema, comunicación y sincronización, mapeo, entre otras); mayor propensión a cometer errores por aumento de la complejidad; mayor dificultad a la hora de probar o depurar un programa; y por último, la fuerte dependencia entre el programa paralelo y la arquitectura de soporte para obtener alto rendimiento.

5.2. Conclusiones.

- Construir un clúster con placas únicas de cómputo o SBC (Single Board Computing) en este caso con Raspberry Pi resultó ser un reto para nosotros sobre todo para aquellos que es su primera vez, (la primera vez siempre resulta ser un poco más complicado por ser un camino desconocido), además, nos acoplamos al trabajo remoto sobre un clúster físico donde gracias a los dispositivos inteligentes como la domótica que se puede conectar a la red de Internet fue más sencillo controlar el funcionamiento del clúster sin la necesidad de encenderlo presencial y manualmente.
- Es útil tener un clúster con placas únicas que se puede transportar y es de gran utilidad para la experimentación y aprendizaje académico debido a sus dimensiones, a su bajo costo económico y rápido crecimiento.
- El clúster con Raspberry Pi resultó ser compatible con muchos sistemas operativos Linux que se pueden adaptar a cada tipo de solución, además, con sus 4 núcleos relativamente potentes, presentan un bajo consumo de energía (5,1 Voltios y un suministro de 2 amperios suele ser suficiente; sin embargo, se recomienda un suministro de 2,5 amperios) lo que lo convierte en una solución atractiva desde la perspectiva de consumo energético que se puede modificar si el número de placas crece significativamente para tener un cluster de alto rendimiento.
- El monitoreo de la temperatura es importante que se atienda porque un sobrecalentamiento puede provocar daños en los circuitos y estropear algún componente del clúster o incluso la pérdida total, por esta razón es que se instaló un ventilador de tamaño considerable para el clúster de 5 placas, que nos ayude a mantener una temperatura baja, este sistema de enfriamiento es un pequeño recurso y de bajo costo, pero puede ayudar a mejorar el rendimiento del clúster. En caso de aumentar el número de placas existen dos opciones como tener carcasas con ventilador y apilar las placas con sistemas de ventiladores externos secundarios.
- Este tipo de clúster será muy útil para el área académica, es decir, aprender a armar, configurar y administrar clústers de mayor tamaño e incluso clústers en la nube, pero siempre en indispensable tener una herramienta con la cual practicar y aprender las bases. También se puede adaptar a muchas áreas de estudio que requieran: procesar grandes cantidades de datos, consultas en paralelo o distribuidas o para realizar cálculo numérico en paralelo. Además, escalar este clúster será relativamente sencillo gracias a los scripts de automatización y configuración de este trabajo.
- La integración del clúster se logra por amplio desarrollo del software libre iniciando desde el sistema operativo como Raspberry OS, el administrador de los sistemas de archivos como NFS, los sistemas de monitoreo y distribución de carga de trabajo como Glances, las bibliotecas de procesamiento paralelo como OpenMPI y los sistemas de scripts que se ejecutan en paralelo como ClusterSSH. Es importante mencionar que este software que es conocido como una la capa intermedia (middleware) permite ver este clúster y otros de mayor tamaño como una computadora paralela funcional.
- Los materiales que se usaron para el clúster resultaron de buena calidad ya que nos permitieron configurar de mejor manera el clúster. Consideramos que contar con los materiales es importante para la administración del tiempo del proyecto, ya que conseguir los materiales actualmente puede ser un poco complicado porque el hardware va cambiando rápidamente y lo que hoy puede estar en existencia con el paso del tiempo puede llegar a ya no estar disponible en un futuro. Además, no es fácil conseguir proveedores de SBC en México con suficientes unidades en existencia y a buen precio, sin embargo a nivel global sí es posible adquirir placas de diversos fabricantes.

Apéndice A

Configuración de clúster con placas Rpi4

Para configurar un clúster con Rpi4 de manera automatizada se usaron los scripts que se mostraron en el Capítulo 2, y fue necesario instalar el mismo sistema operativo. Los pasos a seguir se explican a continuación:

1. Instalar el sistema operativo en una tarjeta micro SD para el nodo maestro, se recomienda la versión Desktop (con interfaz gráfica).

2. Configurar el nodo maestro como se explica en el Capítulo 2.

3. Instalar el sistema operativo en un nodo secundario, se recomienda la versión lite (sin interfaz gráfica).

4. Configurar el nodo secundario como se explica en el Capítulo 2.

5. Clonar la imagen del sistema operativo del nodo secundario con la configuración básica. El clonado de la imagen se puede realizar de diferentes formas, pero en esta ocasión se usó el programa "Win32 Disk Imager", cuya descarga se puede realizar en el siguiente enlace: https://sourceforge.net/projects/win32di

En cuanto se tenga instalado el programa "Win32 Disk Imager", basta con insertar la micro SD en una computadora para poder copiarla dando clic en ”Read", inmediatamente iniciará la creación de la copia de respaldo con toda la información que se encuentra en la unidad D:\ como se visualiza en la imagen A.1

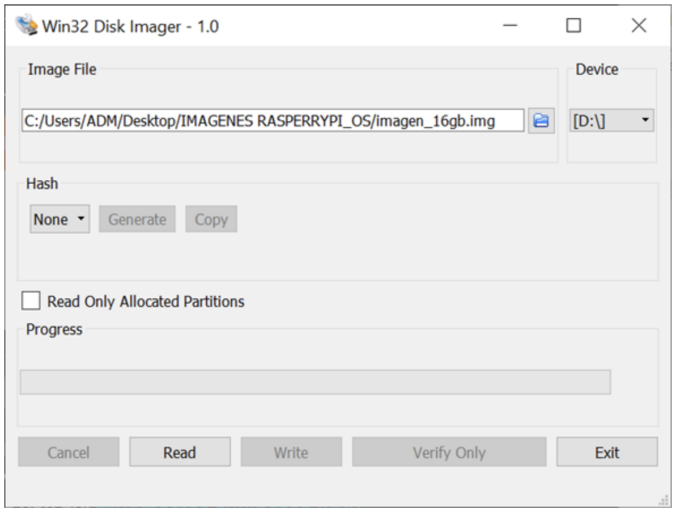


Figura A.1: Win32 Disk Imager Read.

Una vez que se tenga la imagen con la copia de respaldo, se debe insertar otra micro SD sin información y dar clic en el botón "Write"para escribir la imagen en la nueva micro SD, Figura A.2.

Si se desea explorar diferentes formas de clonar imágenes se puede consultar el siguiente enlace: https://beebom.com/how-clone-raspberry-pi-sd-card-windows-linux-macos/

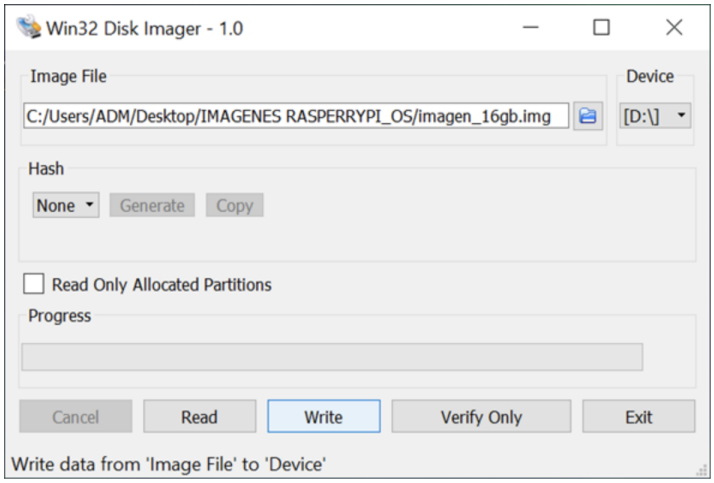


Figura A.2: Win32 Disk Imager Write.

A.1. Scripts de automatización adicionales.

A continuación se muestran algunos scripts que pueden ayudar a desplegar la configuración básica de los nodos de manera más automatizada.

A.1.1. Scripts para el nodo maestro

En el nodo maestro se pueden ejecutar los siguientes scripts:

1. **Apagar los nodos.**

   Para apagar n nodos simplemente se necesita ejecutar el siguiente script. De lo contrario se tendría que apagar nodo por nodo.

```
#!/bin/sh
ssh nodo01 sudo shutdown -h now
ssh nodo02 sudo shutdown -h now
ssh nodo03 sudo shutdown -h now
ssh nodo04 sudo shutdown -h now
ssh nodo05 sudo shutdown -h now
ssh nodo06 sudo shutdown -h now
ssh nodo07 sudo shutdown -h now
ssh nodo08 sudo shutdown -h now
sudo shutdown -h now
```

2. **Reiniciar Nodos.**

   En ocasiones puede ser necesario reiniciar todos los nodos, para ello se puede ejecutar el siguiente script:

```
#!/bin/sh
ssh nodo01 sudo reboot -h now
ssh nodo02 sudo reboot -h now
ssh nodo03 sudo reboot -h now
ssh nodo04 sudo reboot -h now
sudo reboot -h now
```

3. **Copiar las llaves SSH en los diferentes nodos secundarios.**

   El siguiente script ayudará a copiar las llaves SSH del nodo maestro a los nodos secundarios para poder ejecutar los procesamientos distribuidos o paralelos a través de las conexiones SSH.

```
#!/bin/bash
ssh-copy-id nodo01
ssh-copy-id nodo02
ssh-copy-id nodo03
ssh-copy-id nodo04
```

4. **Configuración de DHCP.**

   Para ejecutar el script de configuración de DHCP, se debe ejecutar en cada nodo pasándole el parámetro del último segmento de la IP, por ejemplo, si se desea asignar la IP 192.168.1.1 se debe ejecutar de la siguiente manera: <./DHCPCD.sh 1>

```
#!/bin/bash
rm /etc/dhcpcd.conf

cat <<EOF>> /etc/dhcpcd.conf

interface eth0
static ip_address=192.168.1.$1
static routers=192.168.0.1
static domain_name_servers=192.168.1.1 8.8.8.8

interface wlan0
static ip_address=192.168.1.$1
static routers=192.168.0.1
```

```
static domain_name_servers=192.168.1.1 8.8.8.8
EOF
```

5. **Configuración del host.**

   Para identificar a cada nodo por su nombre de host se puede ejecutar el siguiente script que ayudará a editar el archivo /etc/hosts

```
#!/bin/bash

#Se elimina el archivo hosts actual
rm /etc/hosts

#Se agregan los hostnames de los nodos de nuestro clúster
cat <<EOF>> /etc/hosts
192.168.1.1 master.cluster.rpi master
192.168.1.2 nodo01.cluster.rpi nodo01
192.168.1.3 nodo02.cluster.rpi nodo02
192.168.1.4 nodo03.cluster.rpi nodo03
192.168.1.5 nodo04.cluster.rpi nodo04
EOF
```

6. **Configuración NFS**

```
#!/bin/bash

# Instalamos los paquetes
apt install nfs-kernel-server nfs-common -y
# Creamos el directorio donde se alojan las librerías de compilación
mkdir /SOFT
# Cambiamos sus permisos
chown pi:pi /SOFT
# En el archivo /etc/export configuramos los directorios que se compartirán en la red
cat « EOF » /etc/exports
/SOFT 192.168.1.0/255.255.255.0(rw,sync,no_subtree_check,no_root_squash)
/home 192.168.1.0/255.255.255.0(rw,sync,no_subtree_check,no_root_squash)
EOF
# Crea en el servidor NFS la tabla de directorios compartidos.
exportfs -a
# Reiniciamos el servicio
systemctl restart nfs-kernel-server
```

7. **Configuración OpenMPI**

```
#!/ bin / bash

#Nos posicionamos en el directorio /tmp con el instrucción:
cd /tmp
#Descargamos los archivos fuentes de OpenMPI con el instrucción wget:
wget https://download.open-mpi.org/release/open-mpi/v3.0/openmpi-3.0.0.tar.bz2
#Descomprimimos el archivo descargado con el instrucción tar:
tar -xf openmpi-3.0.0.tar.bz2
#Accedemos al directorio que se descomprimió.
cd openmpi-3.0.0
```

```
#Creamos el directorio "build" donde se irán compilando los proyectos
mkdir -p build
#Creamos el directorio donde se instalará OpenMPI.
mkdir /SOFT/openmpi
#Le damos permisos al usuario Pi sobre el directorio
chown pi:pi /SOFT/openmpi
#Ejecutamos la configuración y preparamos las directivas para Make, eso hará que se adapte
#a nuestra arquitectura (este paso puede tardar varios minutos)
../configure CFLAGS=-march=armv7-a CCASFLAGS=-march=armv7-a
--prefix=/SOFT/openmpi/ --enable-mpi-fortran --disable-mpi-cxx
--disable-mpi-cxx-seek --disable-java --disable-ipv6
#Compilamos e instalamos (también tardará varios minutos).
make -j 3 && make install
```

A.1.2. Scripts para nodos secundarios

En los nodos secundarios se pueden ejecutar los siguientes scripts:

1. **NFS Cliente** En los nodos secundarios se deben montar los directorios NFS, por lo que será necesario ejecutar el siguiente script.

```
#!/ bin / bash
#Instalamos el cliente NFS con el siguiente instrucción:
sudo apt-get install nfs-common -y
#Creamos el directorio /SOFT donde se montará el directorio remoto,
#(el directorio /home ya lo tiene creado por default)
mkdir /SOFT
# Modificamos los permisos del directorio al usuario pi
#para que pueda hacer uso de ellos.
chown pi:pi /SOFT
# Editamos el archivo /etc/fstab para montar los directorios
#compartidos de manera automática desde el arranque de los nodos.
cat << EOF >> /etc/fstab
maestro:/SOFT /SOFT nfs defaults 0 0
maestro:/home /home nfs defaults 0 0
EOF
```

2. (Configuración de DHCP expuesto en el Apéndice A, punto 4 de los scripts del nodo maestro, pero adaptando la respectiva IP).

Apéndice B

Infraestructura física del clúster Rpi4

Los materiales que se usaron para construir el clúster con Rpi4 son muy similares a los que se usaron en la versión de Rpi3, pero más modernos.



Figura B.1: Accesorios de Rpi4.

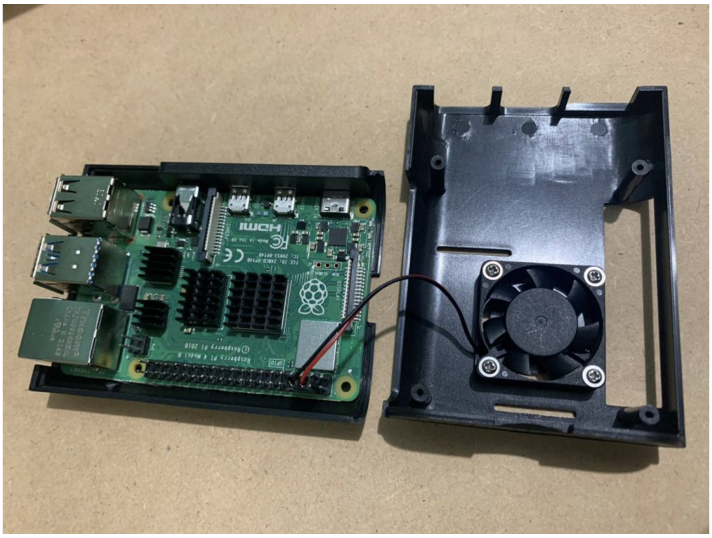


Figura B.2: Tarjeta Rpi4 con ventilador incluido.



Figura B.3: Memoria micro SD 32GB.



Figura B.4: Cable de corriente con interruptor.



Figura B.5: Cable HDMI.



Figura B.6: Adaptador mini USB a HDMI.

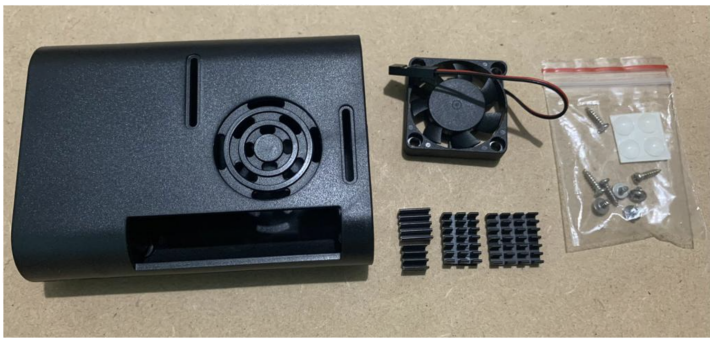


Figura B.7: Carcasas Rpi4 con ventilador y disipadores de calor.



Figura B.8: Rpi4.



Figura B.9: Clúster rpi3 + rpi4, topografía 1.



Figura B.10: Clúster rpi3 + rpi4, topografía 3.

Bibliografía

[AGSS19] Sandra Liliana Allende, Fabian Alejandro Gibellini, Cecilia Beatriz Sánchez, and Monica Mariel Serna. Sistema operativo linux: teoría y práctica 2º edición. *edUTecNe*, 2019.

[AW15] Seemal Asif and Philip Webb. Software system integration-middleware-an overview. In /. Foundation of Computer Science, 2015.

[BBSS01] Daniel J Barrett, Daniel J Barrett, Richard E Silverman, and Richard Silverman. *SSH, the Secure Shell: the definitive guide*. .ºReilly Media, Inc.", 2001.

[Cal] Calculo de pi con mpi. https://lsi2.ugr.es/jmantas/ppr/tutoriales/codigos/mpi/. Accessed: 2022-07-05.

[Cal99] Brent Callaghan. *NFS Illustrated: Network File Sys Illus_ c1*. Addison-Wesley, 1999.

[doc] documentacion mpi. https://www.open-mpi.org/doc/. Accessed: 2022-07-05.

[Esp] Especificaciones rpi 3. https://www.electronicsdatasheets.com/datasheet/56f13a20e34e248d34bd2ca6.pdf. Accessed: 2022-07-05.

[etc] Balena etcher. https://www.balena.io/etcher/. Accessed: 2022-07-06.

[Gla] Glances. https://glances.readthedocs.io/en/latest/index.html. Accessed: 2022-07-05.

[his] Historia del internet. https://www.timetoast.com/timelines/historia-de-el-internet-d81637b7-69d3-44b2-ba7e-837c29c57b5d. Accessed: 2022-07-05.

[Mon16] Simon Monk. *Raspberry Pi cookbook: Software and hardware problems and solutions*. .ºReilly Media, Inc.", 2016.

[OC14] Cristobal Ortega Carrasco. Diseno y evaluación de un clúster hpc: Software de sistema. B.S. thesis, Universitat Politècnica de Catalunya, 2014.

[Pon16] Nicolas Pons. *Linux: principios básicos de uso del sistema*. Ediciones ENI, 2016.

[put] putty, programa para conexión remota. https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html. Accessed: 2022-07-05.

[Ras] Raspberry pi os. https://www.raspberrypi.com/software/. Accessed: 2022-07-05.

[RE18] Pablo Sebastián Rodríguez Eguren. *Análisis del uso de un cluster de Raspberry Pi para cómputo de altas prestaciones*. PhD thesis, Universidad Nacional de La Plata, 2018.

[Ric11] Javier Richarte. *Técnico hardware*. USERSHOP, 2011.

[Sch13] Richard Schmidt. *Software engineering: architecture-driven software development*. Newnes, 2013.

[Smi02] Roderick W Smith. *Advanced Linux Networking*. Addison-Wesley Longman Publishing Co., Inc., 2002.

[spe] speedtest. https://www.speedtest.net/. Accessed: 2022-07-05.

[Ste02] Thomas Lawrence Sterling. *Beowulf cluster computing with Linux*. MIT press, 2002.

[top] Lista top500. https://www.top500.org/. Accessed: 2022-07-05.

[Vér09] Equipo Vértice. *Diseño básico de páginas web en HTML*. Editorial Vértice, 2009.

[Wil06] Philip Wilkinson. *Parallel Programming: Techniques and Applications Using Networked Workstations and Parallel Computers, 2/E*. Pearson Education India, 2006.