**A picture containing text, screenshot, font, number

Description automatically generated**

**Integrantes:**

Carlos Adrián Araya Ramírez

Shakime Richards Sparks

A picture containing text, screenshot, font, number

Description automatically generated

Según [1], MPI es un estándar ampliamente utilizado en la programación paralela en sistemas distribuidos y de memoria compartida. Proporciona un conjunto de funciones, bibliotecas y reglas de comunicación que permiten a los desarrolladores escribir programas paralelos que se ejecutan en múltiples procesadores y computadoras. Se utiliza para facilitar la comunicación y la sincronización entre los procesos paralelos.

A diferencia de otros enfoques de programación paralela que dependen de la memoria compartida, MPI se basa en el intercambio de mensajes, lo que implica que no requiere el uso de memoria compartida entre los procesos. Razón por la cual se convierte en una herramienta especialmente valiosa en sistemas distribuidos, como los supercomputadores LUSITANIA y LUSITANIA II, donde los trabajos se ejecutan en múltiples nodos [2].

A picture containing text, screenshot, font, number

Description automatically generated

En este contexto de MPI, un *rank* se refiere al ID único asignado a cada proceso en un programa paralelo. Según [1], un rank permite distinguir los procesos unos de otros. Se utiliza para identificar y direccionar de manera única cada proceso dentro de un grupo de procesos MPI. Asimismo, puede ser utilizado para establecer comunicación punto a punto entre procesos, especificando el *sender* y el *receiver* de los mensajes.

A picture containing text, screenshot, font, number

Description automatically generated

Según [1], se hace uso de la interfaz de paso de mensajes, el nodo raíz y los nodos conectados a él se establecen a través del uso de las funciones proporcionadas por la biblioteca: MPI\_Init, MPI\_Comm\_rank, MPI\_Comm\_size, MPI\_Send, MPI\_Recv y MPI\_Finalize. El código que ejecuta el nodo raíz y los nodos conectados depende de la lógica específica de la aplicación paralela implementada.

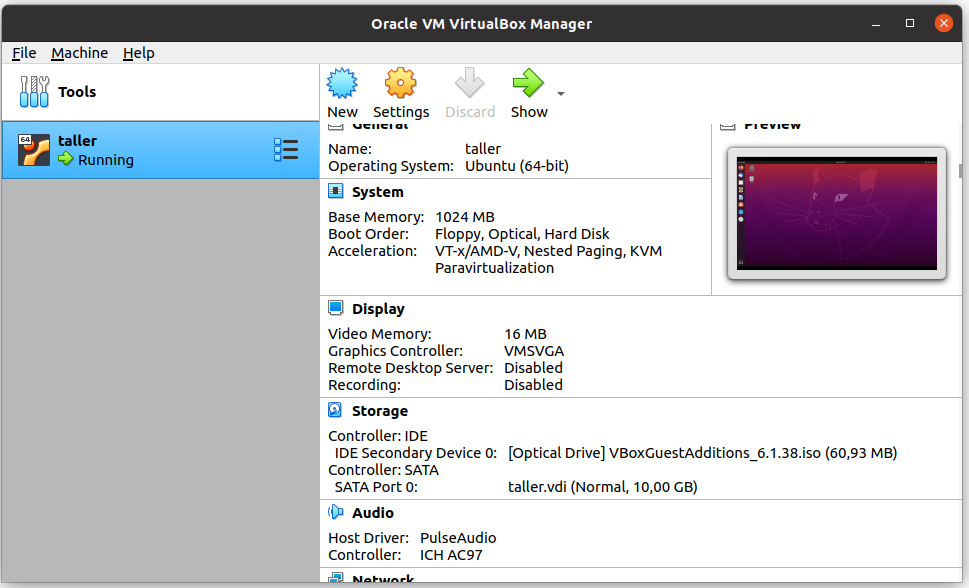
A picture containing text, screenshot, font, number

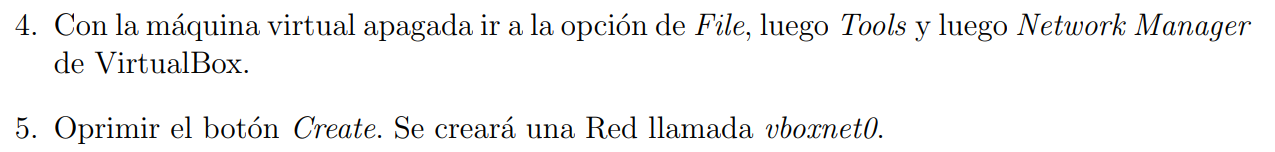
Description automatically generated

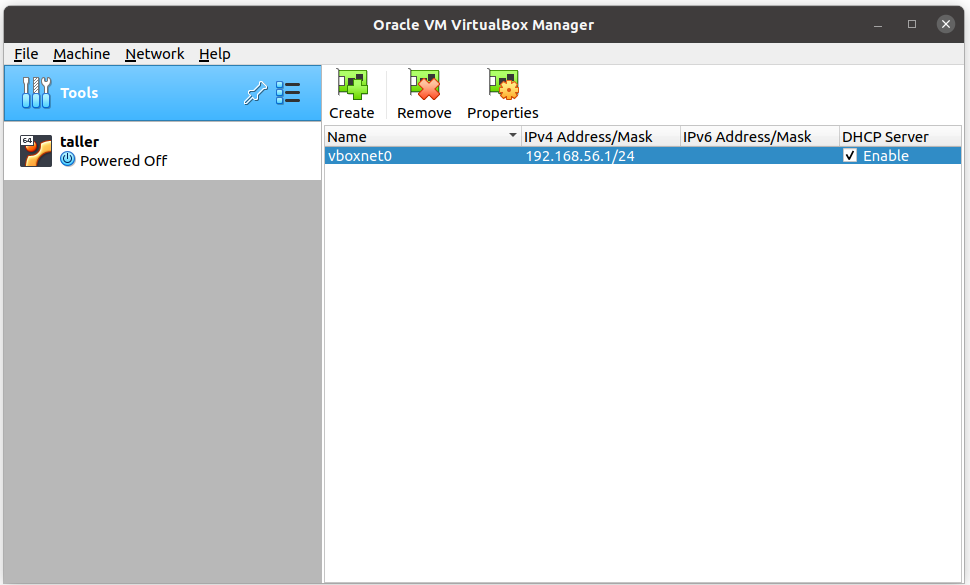
Según [3], MPICH es una implementación popular de la interfaz de paso de mensajes (MPI) que proporciona una biblioteca de funciones y herramientas para la programación paralela en sistemas distribuidos y de memoria compartida. MPICH sigue las especificaciones del estándar MPI y es utilizado por muchos desarrolladores para crear aplicaciones paralelas y distribuidas. Ofrece funcionalidades para la comunicación punto a punto, comunicación colectiva, manejo de grupos y otros aspectos de la programación paralela utilizando MPI. Además, MPICH es ampliamente utilizado en diferentes plataformas de computación y comunicación, incluyendo clústeres de bajo costo, redes de alta velocidad y sistemas de alto rendimiento. Su enfoque modular y su licencia de código abierto permiten su fácil extensibilidad y adaptabilidad a diversos entornos de programación paralela [4].

A picture containing text, screenshot, font

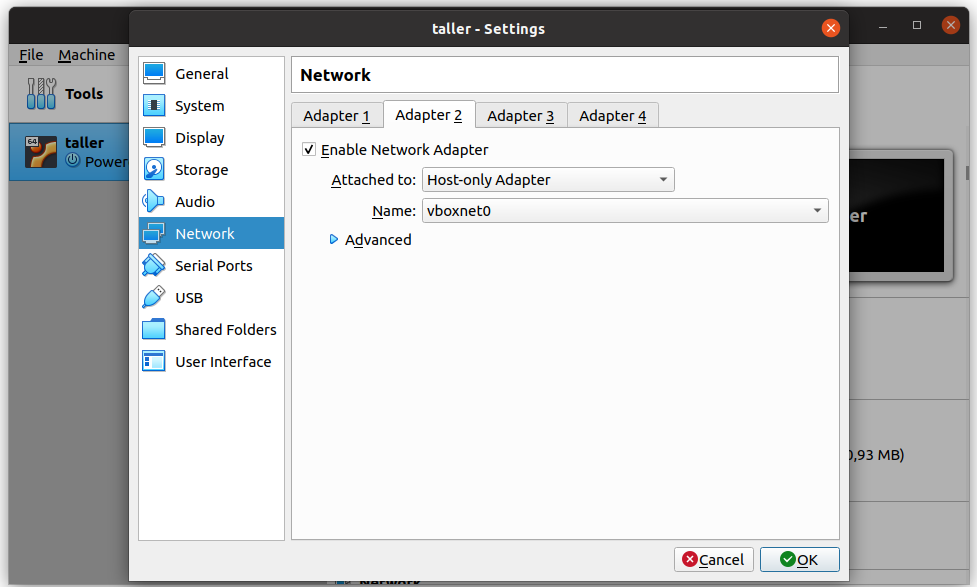
Description automatically generated



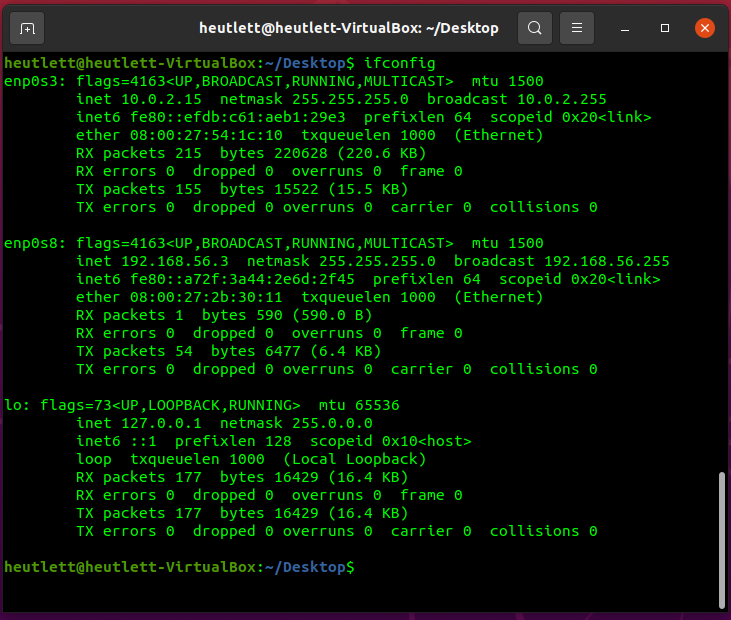


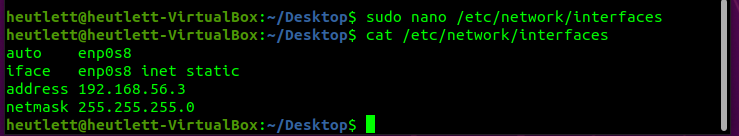


A picture containing text, font, screenshot, line

Description automatically generated

A screenshot of a computer

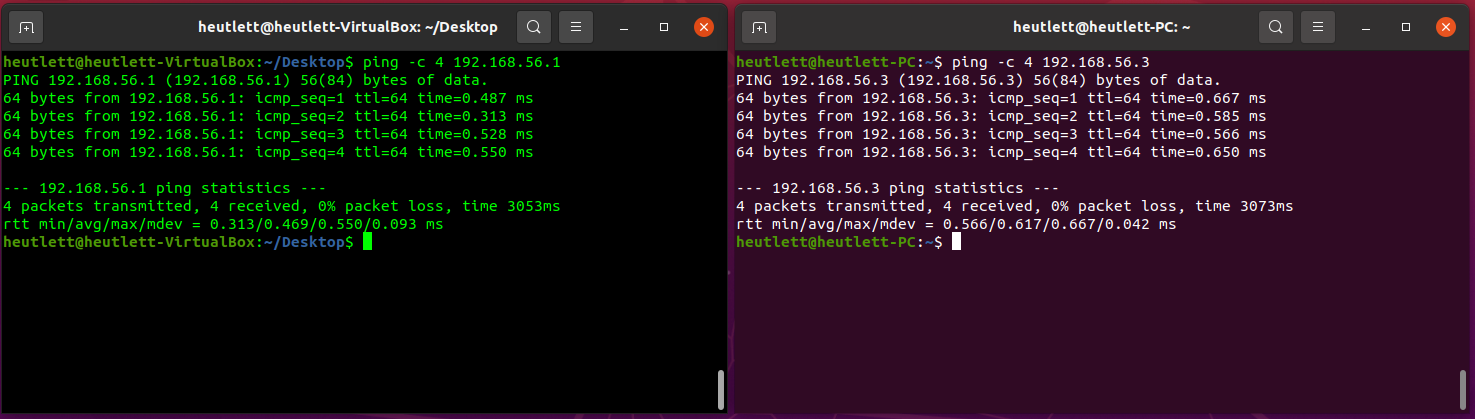
Description automatically generated with low confidence

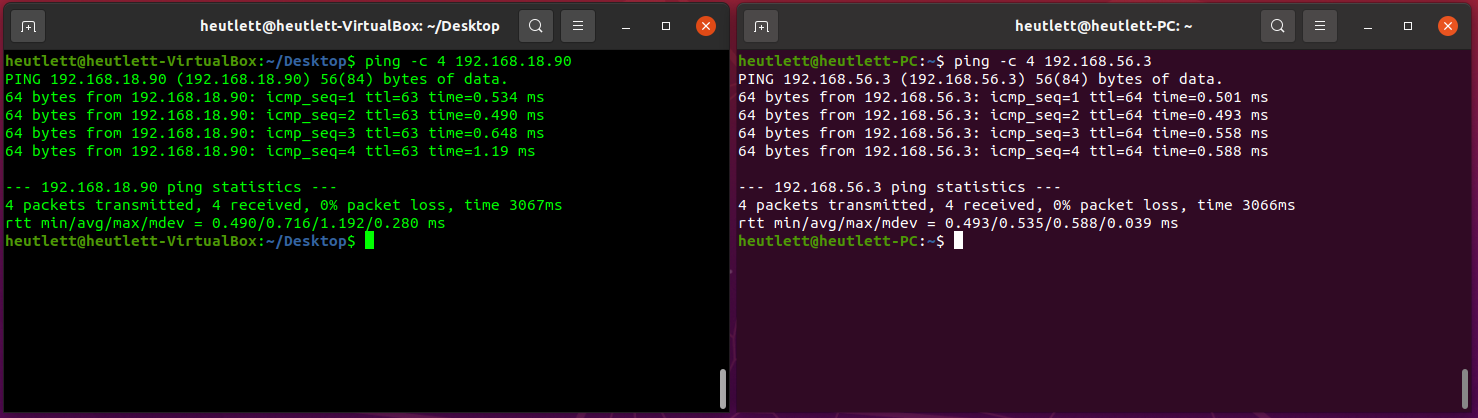


A picture containing text, font, screenshot, algebra

Description automatically generated

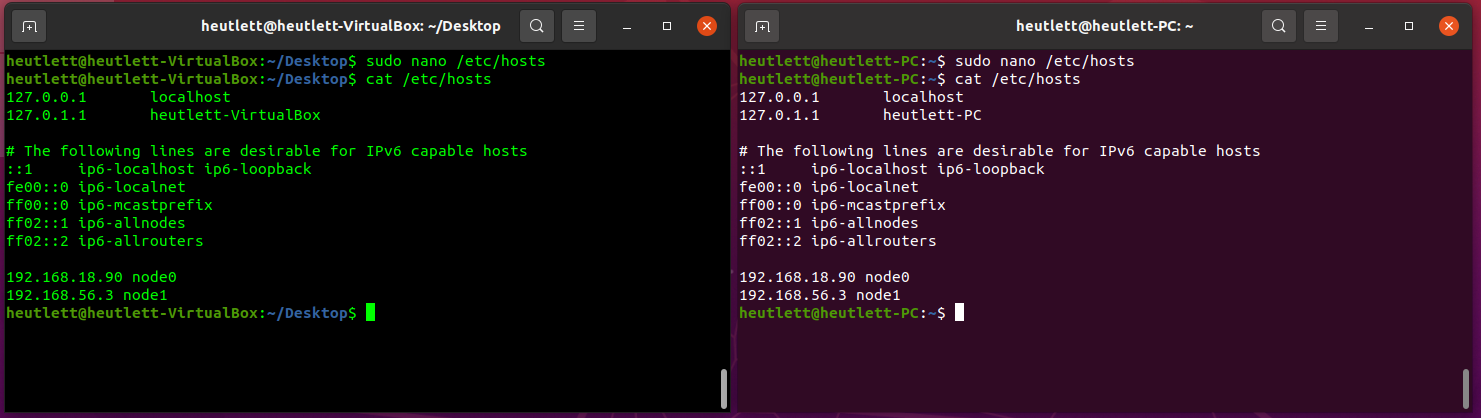




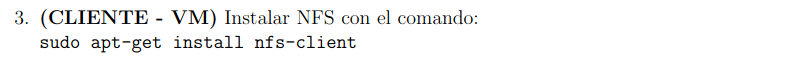


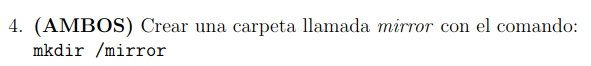
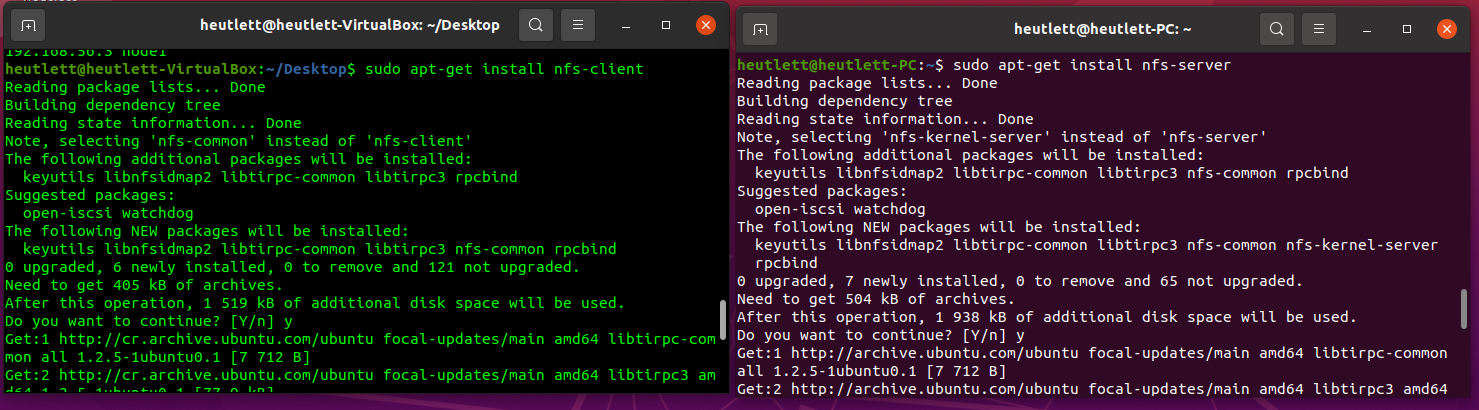
A picture containing text, screenshot, font

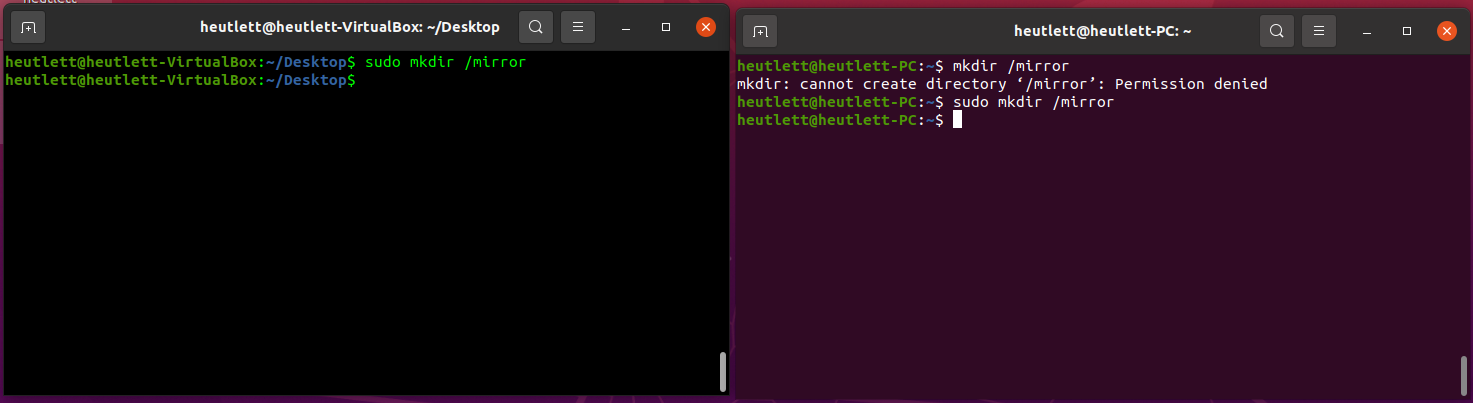
Description automatically generated

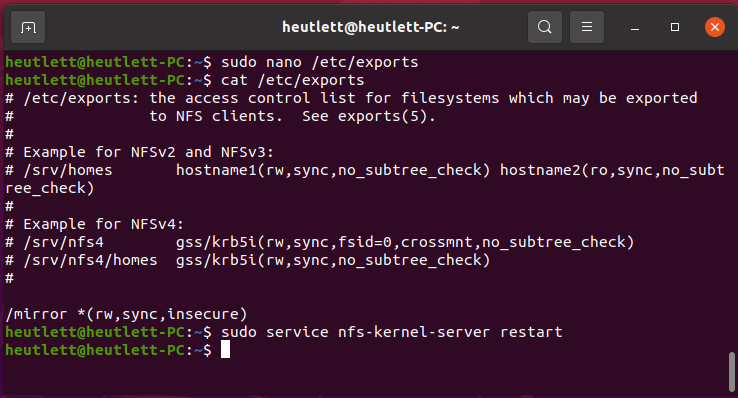
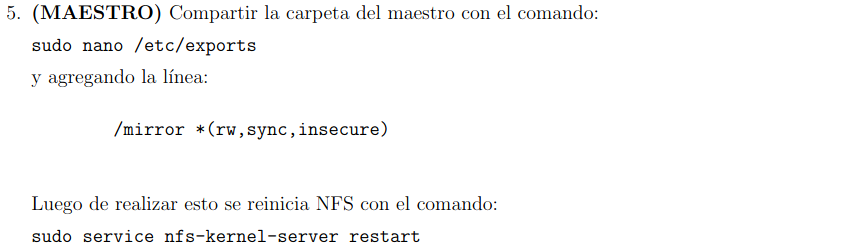


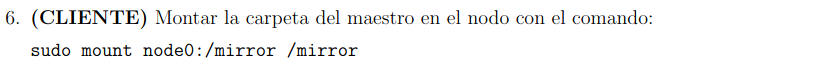


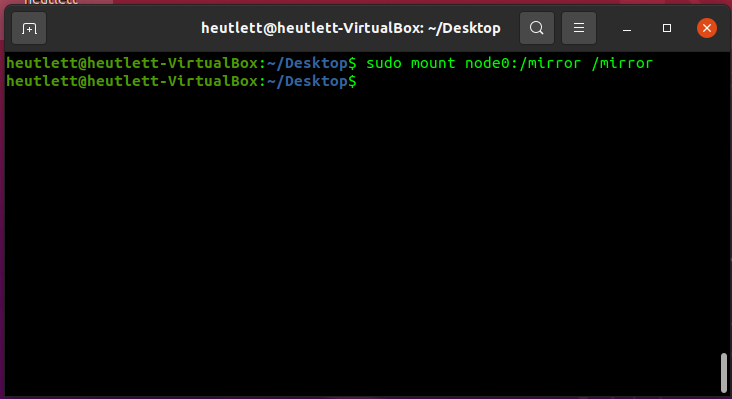


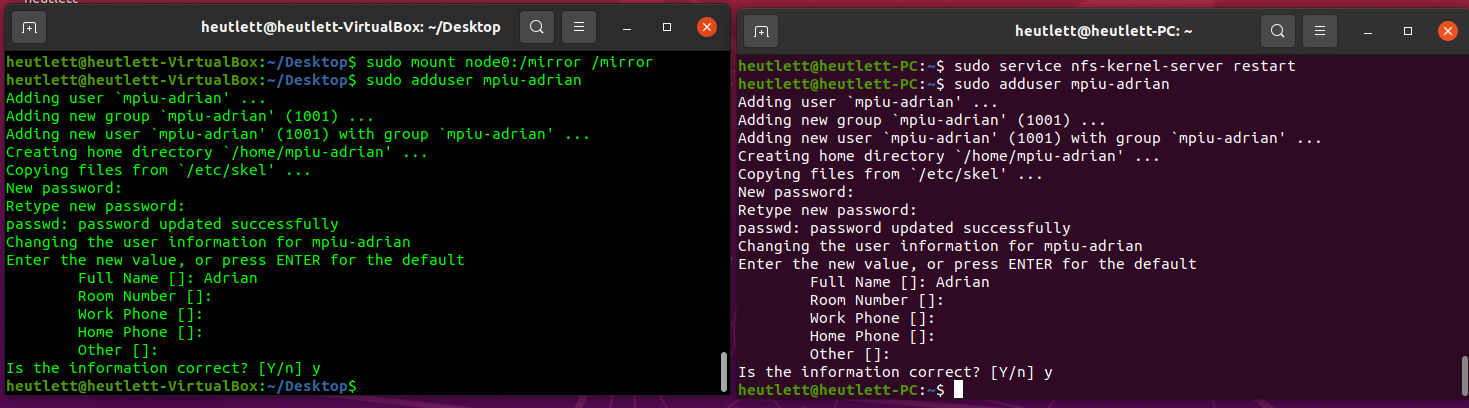


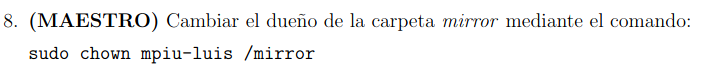


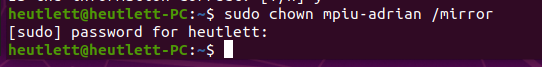


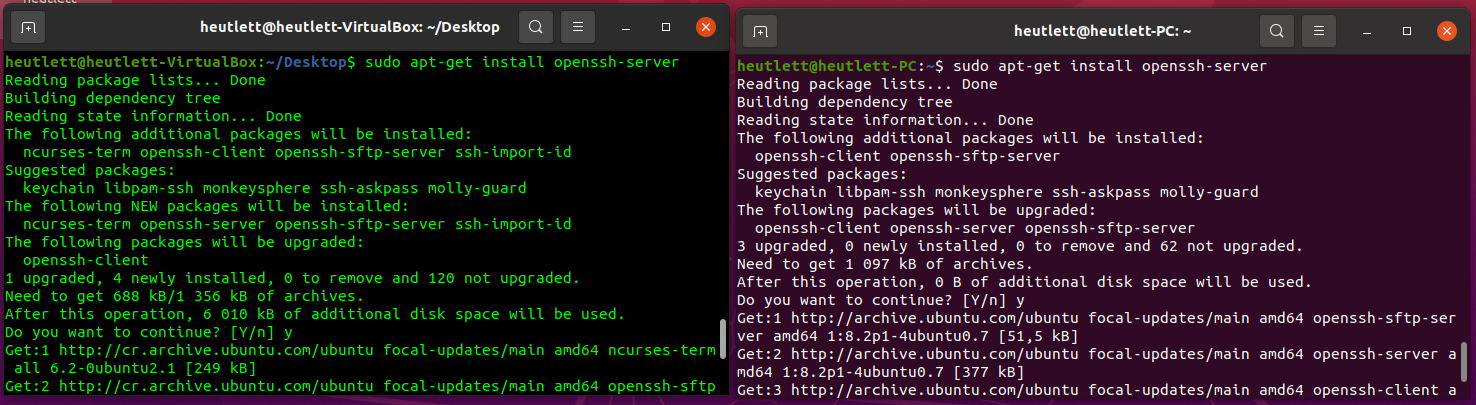
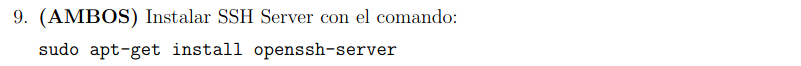


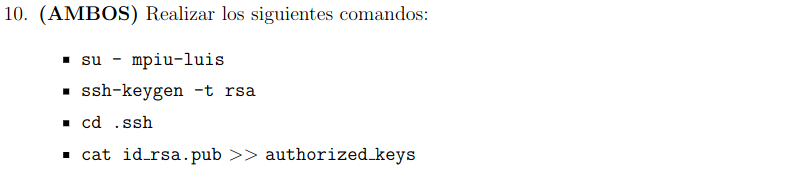






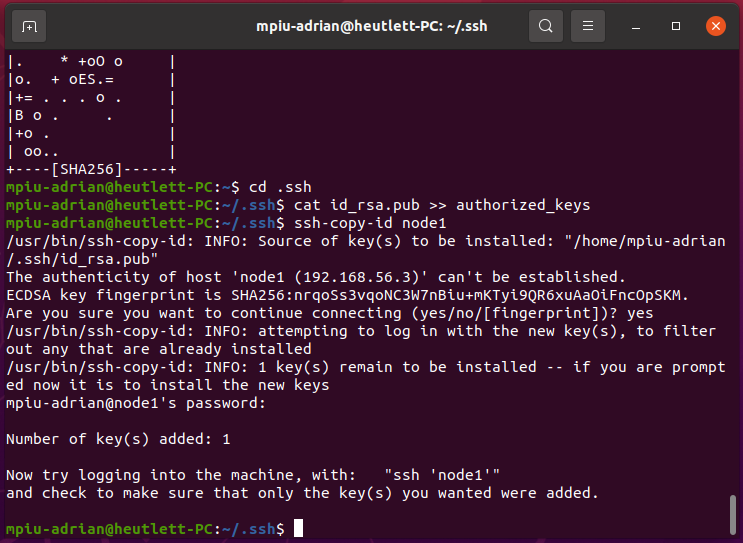


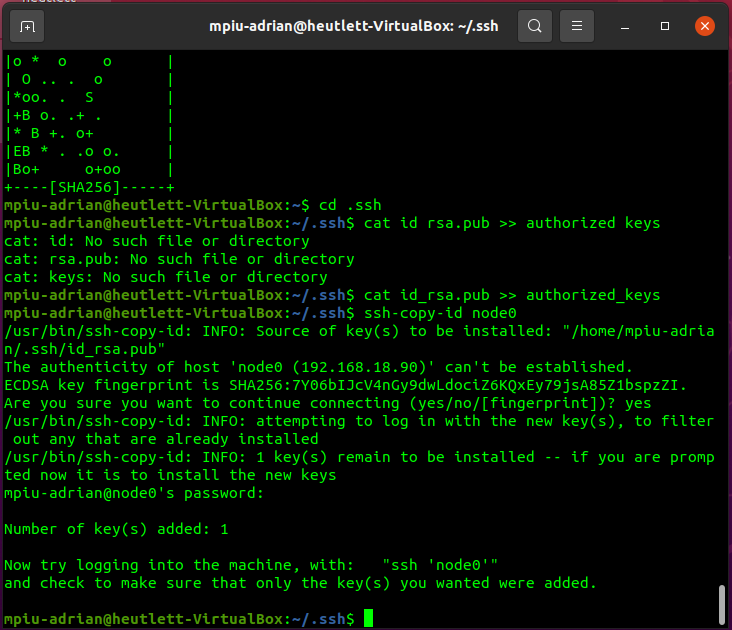


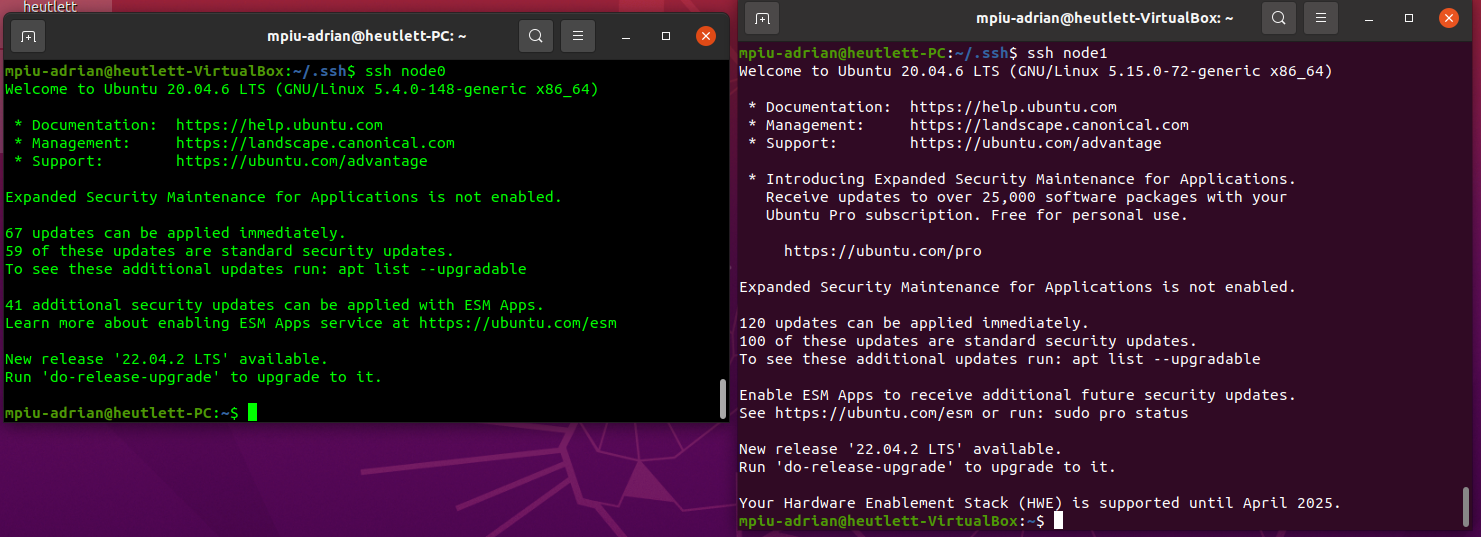
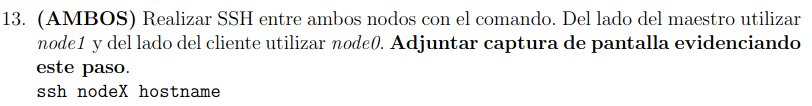




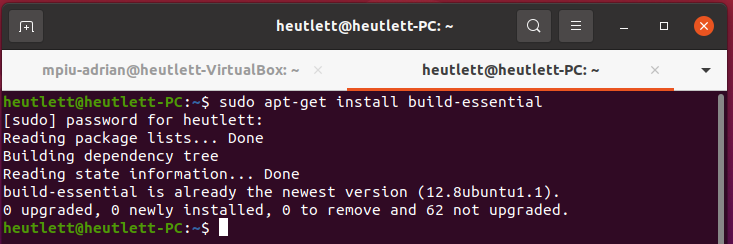




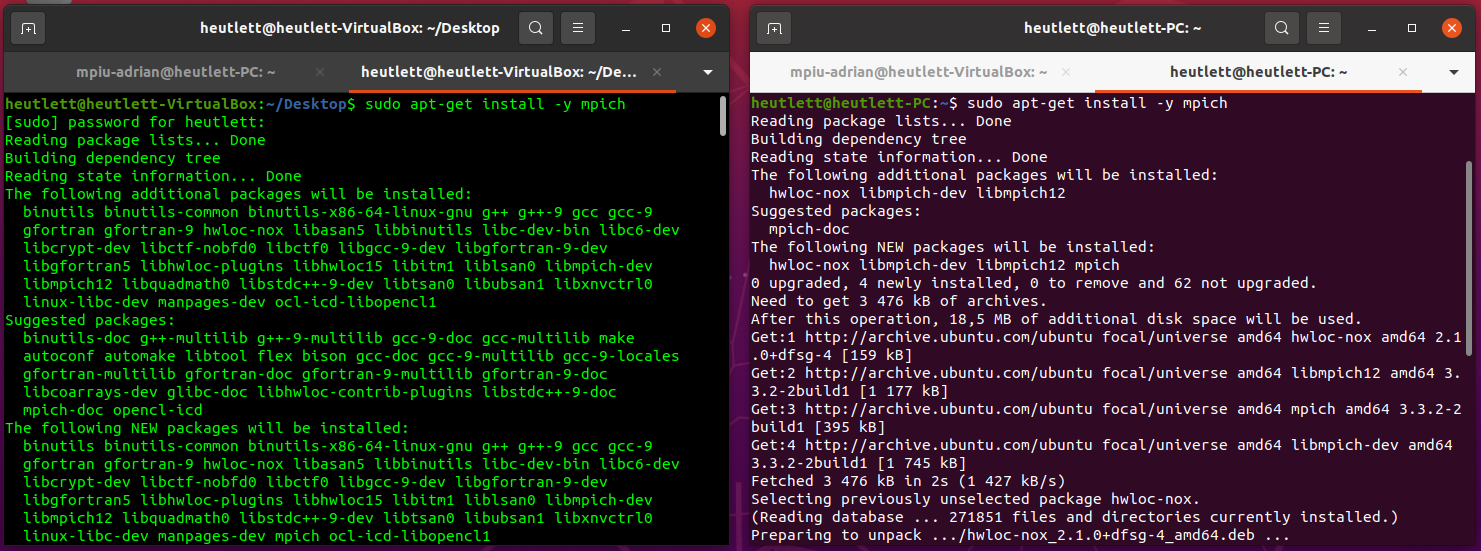


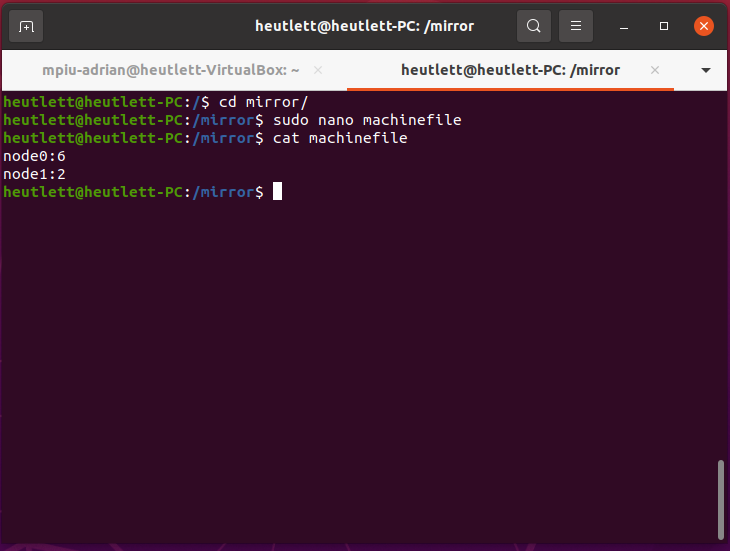
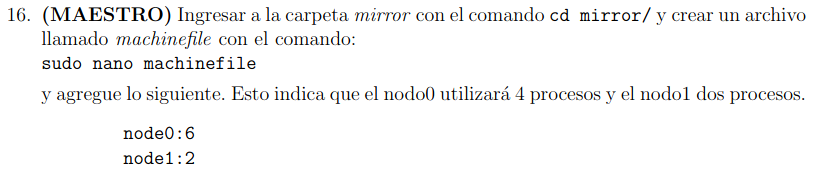




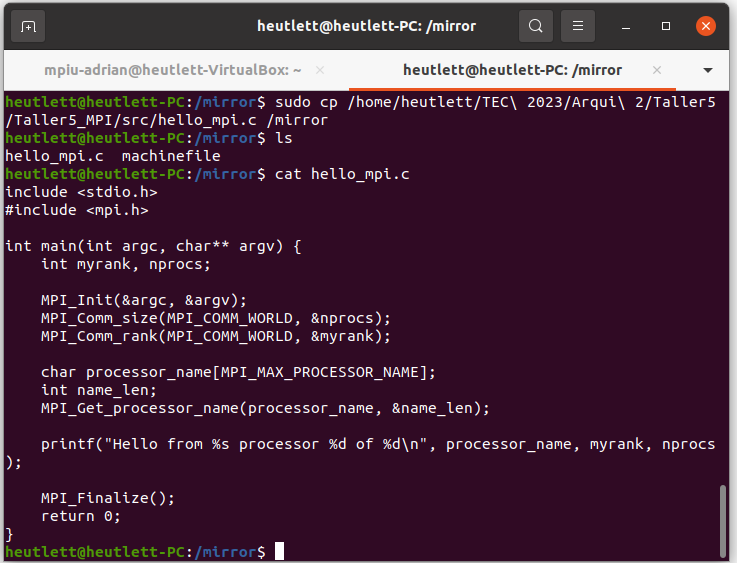


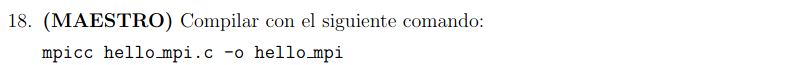


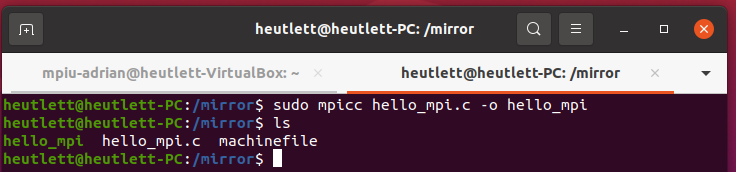


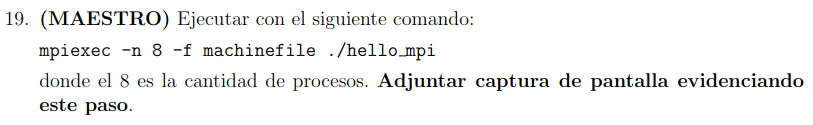


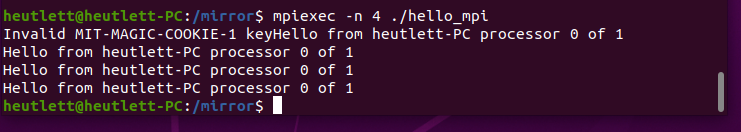


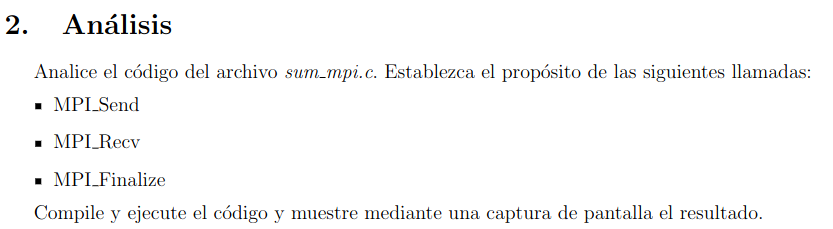




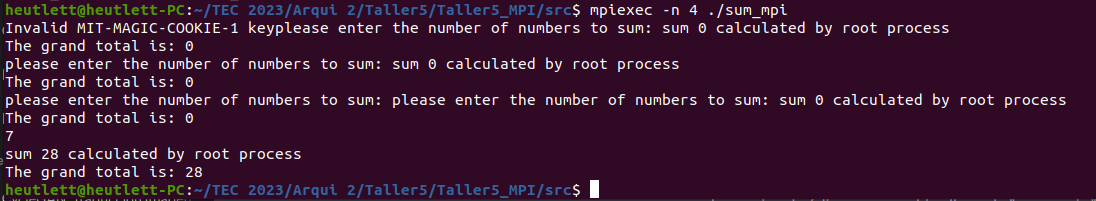


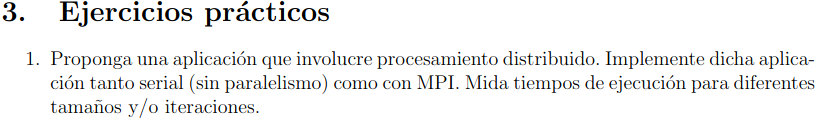






1. MPI\_Send: Esta llamada se utiliza para enviar datos desde un proceso a otro. En el código, se utiliza para que el proceso raíz (root\_process) envíe una porción del array a cada proceso hijo. Envía el número de filas a enviar y las filas correspondientes del array a un proceso hijo específico.
2. MPI\_Recv: Esta llamada se utiliza para recibir datos en un proceso desde otro proceso. En el código, los procesos hijos utilizan esta llamada para recibir su segmento del array desde el proceso raíz. Reciben el número de filas a recibir y las filas correspondientes del array.
3. MPI\_Finalize: Esta llamada se utiliza para finalizar el entorno MPI. Se llama al final del programa para liberar los recursos utilizados por MPI.

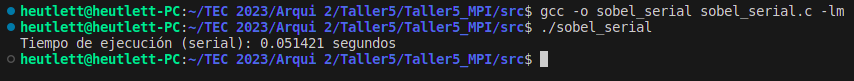




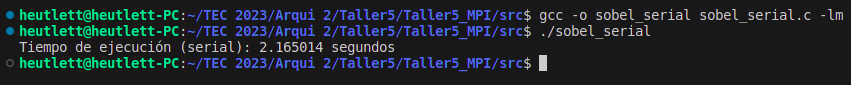
* Serial
  + 100 iteraciones, tamaño de imagen 100



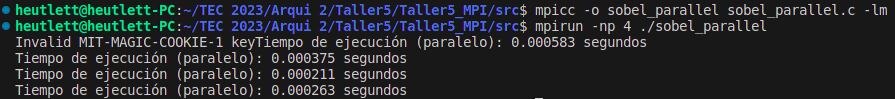
* + 1000 iteraciones, tamaño de imagen 1000



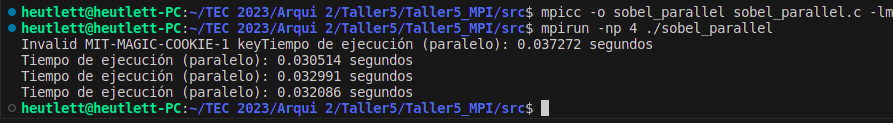
* + 10000 iteraciones, tamaño de imagen 10000



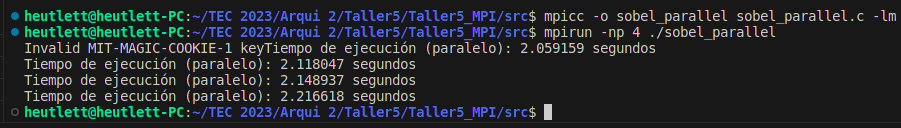
* Paralelismo
  + 100 iteraciones y numRowsPerProc = HEIGHT = 100 / numProcs = 4;



* + 1000 iteraciones y numRowsPerProc = HEIGHT = 1000 / numProcs = 4;



* + 10000 iteraciones y numRowsPerProc = HEIGHT = 10000 / numProcs = 4;



**References**

[1] W. Gropp, E. Lusk, and R. Thakur, "Using MPI: Portable Parallel Programming with the Message-Passing Interface," 2nd ed. Cambridge, MA: MIT Press, 1999. <https://doc.lagout.org/science/0_Computer%20Science/5_Parallel%20and%20Distributed/Using%20MPI-2%20-%20Advanced%20Features.pdf>

[2] ¿Qué es MPI?, CénitS - COMPUTAEX, May 21, 2010. <https://www.cenits.es/faq/preguntas-generales/que-es-mpi>.

[3] M. Snir, S. Otto, S. Huss-Lederman, D. Walker, and J. Dongarra, "MPI: The Complete Reference," Vol. 1: The MPI Core. Cambridge, MA: MIT Press, 1996. <http://www.cslab.ntua.gr/courses/common/mpi-book.pdf>

[4] “MPICH Overview | MPICH.” <https://www.mpich.org/about/overview/>