



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

**Diseño e implementación de un clúster
HPC académico como plataforma para
análisis genómico y epidemiológico en
estudios poblacionales**

TESINA

Que para obtener el título de
Ingeniero en Computación

P R E S E N T A

Angel Gael Pacheco Saavedra

DIRECTOR DE TESINA

Mtro. Gerardo Gabriel Carrasco Zúñiga



Ciudad Universitaria, Cd. Mx., 2026



**PROTESTA UNIVERSITARIA DE INTEGRIDAD Y
HONESTIDAD ACADÉMICA Y PROFESIONAL
(Titulación con trabajo escrito)**



De conformidad con lo dispuesto en los artículos 87, fracción V, del Estatuto General, 68, primer párrafo, del Reglamento General de Estudios Universitarios y 26, fracción I, y 35 del Reglamento General de Exámenes, me comprometo en todo tiempo a honrar a la institución y a cumplir con los principios establecidos en el Código de Ética de la Universidad Nacional Autónoma de México, especialmente con los de integridad y honestidad académica.

De acuerdo con lo anterior, manifiesto que el trabajo escrito titulado DISEÑO E IMPLEMENTACION DE UN CLUSTER HPC ACADEMICO COMO PLATAFORMA PARA ANALISIS GENOMICO Y EPIDEMIOLOGICO EN ESTUDIOS POBLACIONALES que presenté para obtener el título de INGENIERO EN COMPUTACIÓN es original, de mi autoría y lo realicé con el rigor metodológico exigido por mi Entidad Académica, citando las fuentes de ideas, textos, imágenes, gráficos u otro tipo de obras empleadas para su desarrollo.

En consecuencia, acepto que la falta de cumplimiento de las disposiciones reglamentarias y normativas de la Universidad, en particular las ya referidas en el Código de Ética, llevará a la nulidad de los actos de carácter académico administrativo del proceso de titulación.

ÁNGEL GAEL PACHECO SAAVEDRA
Número de cuenta: 421027688

Contenido

- 1. Introducción1
 - 1.1 Contexto de los estudios genómicos y epidemiológicos en entornos académicos1
 - 1.2 Planteamiento del problema.....2
 - 1.3 Objetivo general y objetivos específicos3
 - 1.4 Alcances y limitaciones del trabajo3
 - 1.5 Convenciones terminológicas.....4
- 2. Marco conceptual.....5
 - 2.1 Cómputo de alto desempeño (HPC)5
 - 2.2 Arquitecturas de clúster en entornos académicos.....7
 - 2.3 Gestión de recursos y trabajos en sistemas HPC.....9
 - 2.4 Tecnologías de contenedores para aplicaciones científicas..... 10
 - 2.5 Almacenamiento para datos científicos: enfoques centralizados vs distribuidos 11
 - 2.6 Conceptos base de almacenamiento distribuido (objetos, bloques y archivos)..... 14
- 3. Análisis de requerimientos 16
 - 3.1 Descripción del entorno computacional inicial 16
 - 3.2 Problemáticas técnicas identificadas 17
 - 3.3 Problemáticas operativas y organizacionales..... 18
 - 3.4 Requerimientos funcionales y no funcionales 19
 - 3.5 Requerimientos de almacenamiento y manejo de datos (capacidad, acceso multiusuario, disponibilidad) 22
- 4. Diseño del clúster HPC académico 24
 - 4.1 Arquitectura lógica del clúster 24
 - 4.2 Roles de los nodos y modelo de operación 26
 - 4.3 Infraestructura de red 28
 - 4.4 Modelo de acceso multiusuario 29
 - 4.5 Consideraciones de seguridad y gestión de usuarios..... 29
 - 4.6 Diseño del subsistema de almacenamiento distribuido (Ceph)..... 31
 - 4.6.1 Rol del almacenamiento en el flujo de trabajo científico..... 31
 - 4.6.2 Componentes y servicios principales considerados 31
 - 4.6.3 Esquema general de acceso y permisos sobre datos 34
- 5. Implementación del clúster 34
 - 5.1 Instalación y configuración del sistema operativo 34

5.2 Implementación del gestor de recursos y trabajos	37
5.3 Integración de tecnologías de contenedores	41
5.4 Configuración de red y servicios base	46
5.5 Implementación del almacenamiento distribuido Ceph	49
5.5.1 Despliegue inicial y configuración general	49
5.5.2 Integración con el clúster y montaje para usuarios/servicios	54
5.5.3 Consideraciones de operación: cuotas, permisos y respaldos.....	55
6. Integración de servicios y soporte a usuarios	57
6.1 Servicios de acceso a entornos de análisis	57
6.2 Servicios de almacenamiento y colaboración	60
6.3 Sistema de soporte y atención a incidencias.....	62
6.4 Servicios de comunicación y notificación	64
6.5 Uso de Ceph como base de almacenamiento para servicios y flujos de trabajo	66
7. Validación operativa del sistema.....	67
7.1 Metodología de validación	67
7.2 Validación del gestor de trabajos y ejecución en entorno multiusuario.....	68
7.3 Validación de cargas de trabajo genómicas y epidemiológicas representativas	69
7.4 Validación del almacenamiento distribuido (Ceph)	71
7.4.1 Pruebas funcionales de disponibilidad y acceso concurrente.....	71
7.4.2 Verificación de permisos y aislamiento entre usuarios.....	72
7.5 Resultados de la validación	72
8. Discusión.....	75
8.1 Análisis de los resultados obtenidos	75
8.2 Limitaciones del sistema implementado.....	77
8.3 Consideraciones operativas: mantenimiento, monitoreo y continuidad	78
9. Conclusiones y trabajo futuro	80
9.1 Conclusiones.....	80
9.2 Líneas de trabajo futuro	81
10. Referencias	83
Anexos	84
Anexo A.....	85
Anexo B.....	102

Anexo C.....	104
Anexo D	110

1. Introducción

1.1 Contexto de los estudios genómicos y epidemiológicos en entornos académicos

Cuando se habla de estudios genómicos se hace referencia al análisis del genoma, entendido como el conjunto completo de información genética de un organismo. En los últimos años, el desarrollo de tecnologías de secuenciación ha generado grandes volúmenes de datos genéticos que son utilizados para estudiar la variabilidad genética entre individuos y su relación con distintas condiciones de salud.

Un ejemplo de este tipo de investigaciones es el Estudio Prospectivo de la Ciudad de México (Mexico City Prospective Study, MCPS). Tapia-Conyer et al. (2006) describen el diseño de esta cohorte, que reclutó a más de 150,000 adultos entre 1998 y 2004 en las delegaciones de Coyoacán e Iztapalapa con el propósito de estudiar los efectos del tabaco y otros factores de riesgo sobre las enfermedades crónicas. Casi dos décadas después, Ziyatdinov et al. (2023) reportaron la caracterización genómica de esa misma cohorte, que incluye 138,511 participantes genotipados con microarreglo, 141,046 exomas secuenciados y 9,950 genomas completos. De este trabajo proviene la información genética con la que se realizan los análisis descritos en este documento.

A diferencia de la mayoría de los estudios genómicos realizados principalmente en poblaciones europeas, este estudio se centra en la diversidad genética presente en la población mexicana, donde la ancestría amerindia juega un papel importante. De acuerdo con Ziyatdinov et al. (2023), la composición de ancestría promedio de los participantes es de aproximadamente 67 % indígena americana, 30 % europea y 3 % africana. Con esto aparecen variantes genéticas que podrían no ser detectadas en estudios realizados en otras poblaciones como la europea.

Desde el punto de vista epidemiológico, estos estudios buscan comprender la relación entre el genotipo (información genética) y el fenotipo (características observables o condiciones de salud), con el objetivo de analizar el comportamiento de las enfermedades dentro de una población. Las investigaciones realizadas dentro del Estudio Prospectivo de la Ciudad de México han mostrado que la ancestría genética local a lo largo del genoma puede influir en el riesgo de desarrollar ciertas enfermedades (Ziyatdinov et al., 2023).

El análisis de este tipo de información requiere el procesamiento de grandes volúmenes de datos genómicos y epidemiológicos, lo que implica la ejecución de algoritmos con una

complejidad considerable, además del manejo de múltiples conjuntos de datos en paralelo. En entornos académicos, donde participan diversos investigadores y existen diferentes grupos de trabajo, surge la necesidad de contar con la infraestructura computacional capaz de gestionar estos recursos sin desperdiciarlos, de modo que los análisis corran reproducibles y accesibles para múltiples usuarios.

1.2 Planteamiento del problema

Durante el análisis inicial del entorno computacional utilizado para realizar estudios genómicos y epidemiológicos, se identificaron diversas problemáticas técnicas, operativas y organizacionales que limitaban la ejecución eficiente y continua de las cargas de trabajo científicas. Entre estas problemáticas se encontraba la ausencia de una gestión formal de los recursos computacionales, así como la dependencia de una única estación de trabajo, la cual presentaba fallos recurrentes y no operaba bajo condiciones apropiadas para el procesamiento de datos de alta demanda.

Adicionalmente, se observó una dependencia significativa a la plataforma externa de cómputo, DNA-Nexus, que, si bien permiten la ejecución de análisis especializados, implican costos asociados al tiempo de ejecución y al uso de recursos computacionales, lo que restringe su utilización sostenida en entornos académicos. Esta situación generaba frustración entre los usuarios y limitaba la flexibilidad operativa para la ejecución de análisis genómicos y epidemiológicos.

En el ámbito organizacional, se detectaron carencias en los mecanismos de comunicación interna, en la definición de normas y estructura operativa, así como la ausencia de documentación formal para el uso y manejo de datos genómicos y epidemiológicos de la organización. Estas deficiencias dificultaban el acceso concurrente a los recursos, la reproducibilidad de los análisis y la continuidad de los trabajos científicos.

El problema que se aborda en este trabajo consiste en la falta de una plataforma computacional integrada donde los recursos de cómputo se gestionen desde un solo punto, con acceso controlado a múltiples usuarios y soportar la ejecución de análisis genómicos y epidemiológicos, considerando las restricciones institucionales existentes. La solución propuesta se enfoca en el diseño e implementación de un clúster HPC académico que integre mecanismos de gestión de trabajos, servicios de soporte y validación funcional, con el objetivo de habilitar un entorno computacional operativo y sostenible para la investigación académica.

1.3 Objetivo general y objetivos específicos

Objetivo general

Diseñar e implementar un clúster HPC académico que funcione como plataforma computacional multiusuario para la ejecución de análisis genómicos y epidemiológicos en estudios poblacionales, integrando servicios de soporte y validando su operación a través de la ejecución de cargas de trabajo representativas.

Objetivos específicos

1. Definir la arquitectura lógica del clúster HPC académico, considerando los roles de los nodos, la infraestructura de red, el modelo de administración y el acceso multiusuario bajo restricciones institucionales de hardware.
2. Implementar el clúster HPC con la configuración del sistema operativo, el gestor de recursos y trabajos, así como la integración de tecnologías de contenedores y servicios web para el acceso a entornos de análisis.
3. Integrar servicios de soporte y gestión de usuarios que permitan la operación continua del clúster, incluyendo servicios de almacenamiento, acceso remoto y atención a incidencias.
4. Validar el funcionamiento del clúster con la ejecución de cargas de trabajo genómicas y epidemiológicas representativas, verificando su correcta operación en un entorno multiusuario.

1.4 Alcances y limitaciones del trabajo

Alcances

El presente trabajo abarca el diseño, documentación e implementación de una plataforma de cómputo de alto desempeño (HPC) orientada a soportar análisis genómicos y epidemiológicos en un entorno académico. La plataforma busca proporcionar una infraestructura estable y accesible donde los investigadores corran sus cargas de trabajo sin pagar por cómputo externo.

Dentro de los alcances del proyecto se incluye la integración de herramientas de soporte y colaboración que faciliten la interacción entre los usuarios del sistema. Para ello se incorporan soluciones de software libre que permiten el acceso a entornos de análisis, almacenamiento colaborativo y sistemas de soporte técnico. Entre estas herramientas se encuentran plataformas

como Nextcloud, Zammad, Jupyter y RStudio, que se usan para el trabajo diario y la comunicación interna.

También, el proyecto contempla la gestión centralizada de los recursos computacionales con el uso del gestor de cargas de trabajo Slurm. Con este gestor se administra la ejecución de tareas científicas usando políticas de Calidad de Servicio (QoS), con el objetivo de controlar el uso de los recursos del clúster, evitar la saturación del sistema y promover un uso equilibrado de la infraestructura por parte de los usuarios.

El diseño de la infraestructura incluye el análisis de la arquitectura de red interna y externa del clúster, la integración de tecnologías de contenedores usando Apptainer para la ejecución aislada de aplicaciones científicas, así como la implementación de un sistema de almacenamiento distribuido basado en Ceph para la gestión de datos científicos. Finalmente, el funcionamiento del sistema es validado con la ejecución de cargas de trabajo gestionadas por Slurm, verificando el acceso remoto de los usuarios a través de SSH y de las interfaces web disponibles.

Limitaciones

El presente trabajo no contempla la evaluación comparativa de rendimiento (benchmarking) frente a otras infraestructuras de cómputo de alto desempeño. Esto se debe tanto a restricciones presupuestales como al hecho de que la adquisición del hardware utilizado no formó parte del proceso de diseño del sistema.

Además, no es posible garantizar un nivel estricto de alta disponibilidad (High Availability) debido a que la infraestructura se encuentra construida a partir de componentes de escritorio convencionales, y no sobre servidores especializados diseñados específicamente para entornos de supercómputo. Por lo tanto, el sistema debe entenderse como una infraestructura académica funcional orientada a facilitar el procesamiento de datos científicos dentro de las limitaciones del hardware disponible.

1.5 Convenciones terminológicas

A lo largo de este documento se emplean términos técnicos que carecen de una traducción establecida en la literatura especializada en español. El criterio adoptado es el siguiente: cuando el término aparece por primera vez se presenta en español seguido del original en inglés entre

paréntesis, y en las menciones posteriores se conserva la forma en inglés. Los nombres de comandos, herramientas y componentes de software se mantienen siempre en su forma original.

2. Marco conceptual

2.1 Cómputo de alto desempeño (HPC)

El cómputo de alto desempeño (*High Performance Computing*, HPC) se refiere al uso de sistemas computacionales con capacidades significativamente superiores a las de una computadora convencional, con el propósito de resolver problemas que demandan grandes cantidades de procesamiento, memoria o almacenamiento (Eijkhout, 2010). A diferencia de una estación de trabajo individual, un sistema HPC corre tareas que de otro modo tomarían tiempos impracticables o que simplemente exceden los recursos disponibles en un solo equipo.

Los HPC se han convertido en un componente fundamental de la investigación científica contemporánea. Disciplinas como la genómica, la simulación molecular, la climatología, la física de partículas y, más recientemente, el entrenamiento de modelos de aprendizaje automático, generan volúmenes de datos que solo se procesan con infraestructuras de cómputo de alto desempeño. En el caso particular de la bioinformática, tareas como el alineamiento de secuencias genómicas, la imputación de genotipos y los estudios de asociación del genoma completo requieren capacidades de procesamiento paralelo y acceso concurrente a grandes volúmenes de datos (Sterling et al., 2018).

Existen distintos modelos para proveer capacidades de HPC. Las supercomputadoras son sistemas monolíticos diseñados para ofrecer el máximo rendimiento posible en una sola instalación. El cómputo en la nube ofrece recursos bajo demanda a través de proveedores externos, con costos asociados al tiempo de uso. El cómputo en malla (*grid computing*) conecta recursos distribuidos geográficamente para resolver problemas de gran escala. La Figura 1 muestra un modelo de sistema de grid donde múltiples organizaciones virtuales comparten recursos heterogéneos; clústeres, supercomputadoras, almacenamiento y satélites, interconectados a través de redes de área amplia (tomado de Hussain et al., 2013, p. 714).

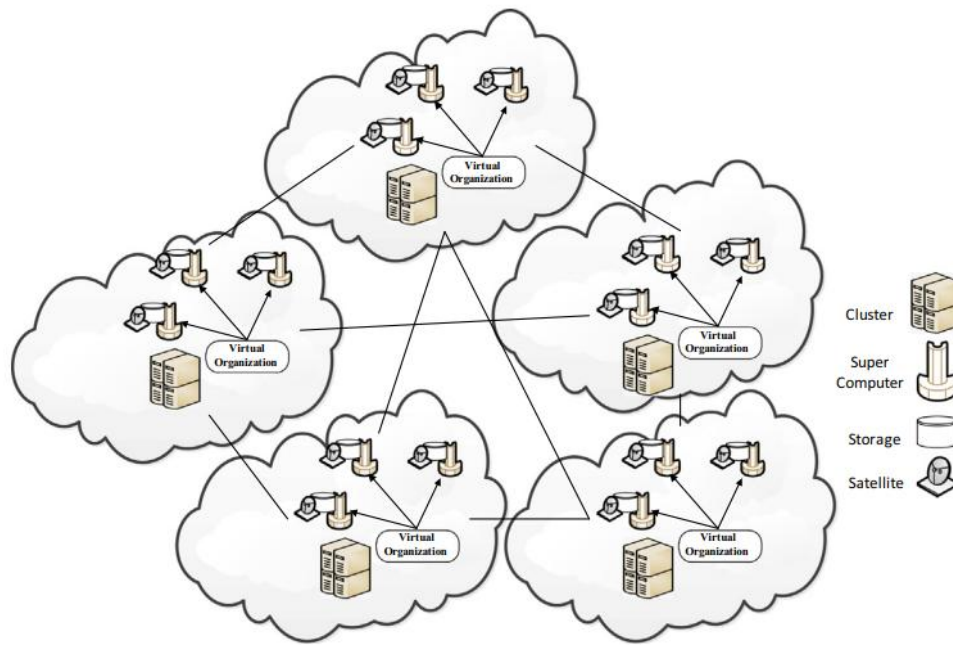


Figura 1. Modelo de un sistema de cómputo grid. De "A survey on resource allocation in high performance distributed computing systems", por H. Hussain et al., 2013, *Parallel Computing*, 39(12), p. 714 (<https://doi.org/10.1016/j.parco.2013.09.009>).

Por su parte, los clústeres consisten en un conjunto de computadoras independientes, denominadas nodos, interconectadas con una red que los hace funcionar como un solo sistema coordinado (Sterling et al., 2018).

Un clúster HPC se compone regularmente de un nodo maestro, que coordina la distribución de trabajos y la administración del sistema, y uno o más nodos de cómputo, que ejecutan las tareas asignadas. La comunicación entre nodos se realiza a través de una red interna de interconexión, y el acceso a los datos se gestiona a través de un sistema de archivos compartido. Un gestor de recursos y trabajos, como Slurm, se encarga de recibir las solicitudes de los usuarios, asignar los recursos disponibles y programar la ejecución de las tareas en los nodos correspondientes (Yoo et al., 2003).

En la práctica, los clústeres HPC desplegados en entornos académicos frecuentemente se construyen con hardware comercial de escritorio o de gama prosumidor, como procesadores Intel Core i9 o AMD Threadripper y tarjetas gráficas de consumo, en contraste con los centros

de supercómputo que emplean procesadores y aceleradores de grado empresarial como Intel Xeon, AMD EPYC o NVIDIA Tesla, así como redes de interconexión de alta velocidad como InfiniBand. Así nace el llamado commodity cluster o clúster tipo Beowulf (Sterling et al., 2018): hardware común que, combinado, da a los grupos de investigación capacidades de cómputo paralelo a una fracción del costo, aun cuando el rendimiento individual de cada nodo sea inferior al de sistemas especializados. De acuerdo con Sterling et al. (2018), lo que define a un clúster como sistema HPC no es el tipo de hardware empleado en cada nodo, sino la integración y coordinación de múltiples computadoras independientes para ejecutar cargas de trabajo que exceden las posibilidades de un solo equipo. La decisión entre los distintos modelos de HPC depende del contexto de uso. Para entornos académicos con necesidades continuas de procesamiento, restricciones presupuestales y requerimientos de privacidad de datos, la implementación de un clúster propio ofrece ventajas en términos de control, disponibilidad y costo a largo plazo frente a alternativas como el cómputo en la nube. También, un clúster escala horizontalmente: se agregan nodos sin cambiar el diseño base. En contraste con el escalamiento vertical, que consiste en incrementar los recursos de una sola máquina y que presenta limitaciones físicas y económicas inherentes, un clúster crece por piezas. En nuestro caso, cada nodo del clúster corre un Threadripper 3960X o un Intel Core i9 sobre Ubuntu 24.04, no Xeons ni EPYCs de grado empresarial. Es exactamente el patrón Beowulf descrito arriba.

2.2 Arquitecturas de clúster en entornos académicos

La arquitectura de un clúster HPC describe la forma en la que se organizan e interconectan sus componentes para funcionar como un sistema integrado. El modelo arquitectónico más común es el denominado maestro-trabajador (*master-worker*), en el cual un nodo maestro actúa como punto central de administración y acceso al sistema, mientras que uno o más nodos de cómputo ejecutan las tareas asignadas. El nodo maestro es responsable de recibir las solicitudes de los usuarios, gestionar la cola de trabajos usando un planificador de recursos, y coordinar la comunicación con los nodos de cómputo. Los usuarios interactúan exclusivamente con el nodo maestro, ya sea con conexiones de terminal remoto o a través de interfaces web, sin acceder directamente a los nodos de cómputo (Sterling et al., 2018).

Además del nodo maestro y los nodos de cómputo, la arquitectura de un clúster requiere dos componentes fundamentales: una red de interconexión y un sistema de almacenamiento compartido. La red de interconexión permite la comunicación entre los nodos y la transferencia

de datos entre ellos. En clústeres de gran escala orientados a simulaciones que requieren comunicación intensiva entre procesos, se emplean redes de alta velocidad y baja latencia como InfiniBand. Sin embargo, para cargas de trabajo que son predominantemente independientes entre sí, como muchos análisis bioinformáticos donde cada muestra o cromosoma se procesa por separado, una red Ethernet estándar de 1 o 10 Gbps resulta suficiente (Sterling et al., 2018). El sistema de almacenamiento compartido, por su parte, proporciona un espacio de datos al que todos los nodos llegan por igual, de modo que los trabajos ejecutados en distintos nodos puedan leer los archivos de entrada y escribir sus resultados sin depender del almacenamiento local de cada máquina. Este almacenamiento puede implementarse por medio de distintas tecnologías, desde sistemas de archivos en red como NFS hasta soluciones de almacenamiento distribuido como CephFS, cuyas características se abordan en las secciones 2.5 y 2.6 de este documento.

En entornos académicos, la arquitectura del clúster generalmente está condicionada por factores que difieren de los centros de supercómputo o de la industria. Entre estos factores se encuentran las restricciones presupuestales, que limitan la adquisición de hardware especializado; la heterogeneidad del equipo disponible, que puede provenir de distintas adquisiciones realizadas en momentos diferentes; las restricciones institucionales de red, que pueden impedir la asignación de direcciones IP públicas a todos los nodos; y la necesidad de dar servicio a múltiples usuarios con perfiles y necesidades diferentes, desde investigadores que ejecutan análisis genómicos hasta estudiantes que realizan prácticas de programación. Estas condiciones obligan a diseñar arquitecturas flexibles que puedan operar con los recursos disponibles y sean administrables por equipos técnicos reducidos.

Un patrón común en clústeres académicos es que el nodo maestro funcione simultáneamente como puerta de enlace (*gateway*) hacia la red externa, proporcionando acceso a internet a los nodos de cómputo que residen en una red interna. Centralizar la seguridad en un solo punto simplifica la administración. Del mismo modo, es común que el nodo maestro aloje servicios adicionales como interfaces web para el envío de trabajos, plataformas de análisis interactivo y sistemas de monitoreo, aprovechando su conectividad externa para ofrecer estos servicios a los usuarios sin exponer directamente los nodos de cómputo (Sterling et al., 2018).

2.3 Gestión de recursos y trabajos en sistemas HPC

En un clúster HPC utilizado por múltiples usuarios. Los recursos computacionales son varios: CPU, memoria RAM, almacenamiento, GPUs. Todos estos recursos se comparten entre los usuarios. Sin reglas, esto se vuelve un caos. Deben compartirse con reglas claras. Sin un mecanismo de control, distintos usuarios podrían intentar utilizar los mismos recursos simultáneamente, lo que provocaría contención, degradación del rendimiento e incluso fallos en la ejecución de las tareas. Para resolver este problema, los sistemas HPC emplean software de gestión de recursos, también denominado gestor de trabajos o planificador de tareas.

De acuerdo con Sterling et al. (2018), un gestor de recursos cumple tres funciones:

- Decide qué nodos, cuántos núcleos, cuánta memoria y, cuando aplica, qué GPU corresponden a cada solicitud.
- Programa el orden de ejecución según la disponibilidad de recursos y las políticas configuradas.
- Lanza los trabajos y registra el uso de recursos.

Los usuarios interactúan con el gestor de recursos usando la definición de trabajos (*jobs*), que son unidades autocontenidas que especifican el programa a ejecutar, los recursos requeridos y las condiciones de ejecución. Estos trabajos se envían a una cola de espera donde el planificador los ordena según las políticas configuradas QoS, como el orden de llegada, la prioridad del usuario o el algoritmo de relleno de huecos (*backfill*), que adelanta trabajos pequeños cuando existen recursos momentáneamente disponibles que no pueden ser aprovechados por trabajos de mayor tamaño en espera. Una vez que los recursos solicitados están disponibles, el gestor lanza los trabajos en los nodos correspondientes y lo retira de la cola.

Los gestores de recursos también organizan los nodos del clúster en particiones, que son agrupaciones lógicas de nodos destinadas a distintos tipos de cargas de trabajo o grupos de usuarios. Por ejemplo, un clúster puede tener una partición de propósito general y otra reservada para trabajos que requieren aceleración por GPU. Cada partición puede tener sus propias políticas de tiempo máximo de ejecución y número máximo de trabajos por usuario.

Entre los gestores de recursos más utilizados en entornos HPC se encuentran Slurm (*Simple Linux Utility for Resource Management*) y PBS (*Portable Batch System*). Slurm es un sistema

de código abierto y arquitectura modular, desarrollado originalmente en el Lawrence Livermore National Laboratory (Yoo et al., 2003), que se ha convertido en el estándar de facto en la comunidad HPC. Su arquitectura se basa en un demonio central (*slurmctld*) que se ejecuta en el nodo maestro el cual se encarga de la planificación y asignación de recursos, y un demonio en cada nodo de cómputo (*slurmd*) que gestiona la ejecución local de los trabajos asignados. En el clúster que se documenta en este trabajo, Slurm 23.11.4 se instaló directamente desde los repositorios de Ubuntu 24.04, sin compilar desde fuente. La contabilidad de uso (*accounting*) se apoya en *slurmdbd* con MariaDB, lo que habilita reportes de uso por usuario y por grupo.

2.4 Tecnologías de contenedores para aplicaciones científicas

Las herramientas utilizadas en bioinformática requieren distintas bibliotecas y, con frecuencia, versiones incompatibles entre sí. Un pipeline de análisis genómico puede depender simultáneamente de una versión específica de R, una versión particular de Python, bibliotecas estadísticas como *bigsnpr* y binarios compilados como GCTB, cada uno con sus propias dependencias de bibliotecas compartidas. Cuando el administrador del sistema actualiza una biblioteca matemática básica como BLAS o LAPACK para satisfacer los requerimientos de otro grupo de trabajo, esa actualización puede romper la compatibilidad de herramientas que funcionaban correctamente el día anterior. En la ingeniería de software, este problema se conoce coloquialmente como el síndrome de "en mi máquina sí funciona". En bioinformática, donde el ecosistema de herramientas de código abierto es particularmente fragmentado, el problema escala a el manejo de las dependencias de software frecuentemente consume más tiempo y esfuerzo que el propio análisis biológico (Buffalo, 2015).

Los contenedores resuelven esta problemática. Un contenedor es una unidad de virtualización a nivel de sistema operativo que empaqueta una aplicación junto con todas sus dependencias. Dentro de una imagen portátil y autocontenido van bibliotecas compartidas, binarios, archivos de configuración, intérpretes y variables de entorno. Esta imagen se ejecuta aislada del sistema anfitrión por medio de dos primitivas del kernel de Linux: los espacios de nombres (*namespaces*) y los grupos de control (*cgroups*). Los *namespaces* proporcionan aislamiento lógico, de modo que el proceso dentro del contenedor percibe su propio árbol de procesos, su propio sistema de archivos raíz y su propia interfaz de red. Los *cgroups*, por su parte, limitan la utilización de recursos físicos como la memoria RAM y los ciclos de CPU asignados al contenedor (Merkel, 2014).

A diferencia de las máquinas virtuales, que incluyen un sistema operativo completo con su propio kernel y requieren un hipervisor para su ejecución, los contenedores resultan mucho más ligeros que las VMs en cuanto a consumo de recursos y más rápidos de iniciar, lo que resulta práctico en entornos HPC donde se lanzan cientos o miles de tareas concurrentes a través del gestor de trabajos.

Una de las tecnologías de contenedores más extendida en la industria, Docker, presenta limitaciones importantes para su uso en clústeres HPC multiusuario. Docker opera con un demonio (dockerd) que se ejecuta con privilegios de superusuario, y los usuarios que interactúan con él obtienen, en la práctica, capacidades equivalentes a las de root en el sistema anfitrión. En un entorno compartido por múltiples investigadores, esto representa un riesgo de seguridad inaceptable (Kurtzer et al., 2017).

Para resolver esta limitación, se desarrolló Singularity, actualmente conocido como Apptainer tras su incorporación a la Linux Foundation en 2021. Apptainer ejecuta los contenedores con los mismos permisos del usuario que los invoca, sin requerir un demonio privilegiado. Las imágenes se distribuyen como archivos .sif (Singularity Image Format) de solo lectura, lo que garantiza su integridad y portabilidad. Apptainer se integra nativamente con gestores de trabajos como Slurm y con sistemas de archivos compartidos, permitiendo que un trabajo programado en el clúster ejecute un contenedor de la misma manera en que ejecutaría cualquier otro programa (Kurtzer et al., 2017).

Desde la perspectiva de la investigación científica, los contenedores contribuyen directamente a la reproducibilidad de los análisis. Una imagen de contenedor congela el entorno de software exacto; las versiones de cada herramienta, las bibliotecas de sistema, los archivos de configuración, tal como estaban cuando corrió el análisis. Con la imagen congelada, otro investigador puede reproducir los mismos resultados meses o años después, sin enfrentar problemas de compatibilidad. En el contexto de un clúster académico, este enfoque aísla los entornos de distintos grupos de trabajo sin contaminar el sistema operativo base del clúster, lo que simplifica la administración y reduce el riesgo de conflictos entre dependencias.

2.5 Almacenamiento para datos científicos: enfoques centralizados vs distribuidos

En un clúster HPC los datos no pueden residir aislados en los discos locales de cada nodo. Cuando un gestor de trabajos como Slurm asigna una tarea a un nodo de cómputo, ese nodo

necesita acceso inmediato a los archivos del proyecto sin importar dónde se hayan generado originalmente. Si cada nodo solo puede leer sus propios discos, el clúster deja de funcionar como un sistema integrado y se convierte en un conjunto de máquinas independientes. Por esta razón, todo clúster HPC requiere algún mecanismo de almacenamiento compartido que permita a todos los nodos acceder a los mismos datos sin notar la diferencia.

La magnitud de este problema depende directamente del tipo de datos que se procesan. En bioinformática, los volúmenes son particularmente grandes. Un solo genoma humano secuenciado a una cobertura de 30× genera entre 100 y 200 GB de datos crudos en formato FASTQ. A lo largo del pipeline de análisis; alineamiento, ordenamiento, marcado de duplicados, llamado de variantes, una sola muestra puede acumular entre 300 y 500 GB en archivos intermedios y finales (Buffalo, 2015). Cuando el estudio involucra miles de muestras, como en el caso del MCPS, la demanda de almacenamiento escala rápidamente a decenas o cientos de terabytes. Estos datos no pueden eliminarse al terminar el análisis: deben persistir durante meses o años para permitir la reproducibilidad de los resultados y el reanálisis con herramientas actualizadas.

El enfoque más sencillo para proveer almacenamiento compartido es el modelo centralizado, es el protocolo NFS (Network File System). En esta configuración existe un único servidor exporta un directorio a través de la red y todos los nodos del clúster lo montan como si fuera almacenamiento local. NFS es fácil de configurar, está integrado nativamente en el kernel de Linux, no requiere software adicional y un administrador puede ponerlo en funcionamiento en cuestión de minutos. Para clústeres pequeños con cargas de trabajo secuenciales y pocos usuarios concurrentes NFS funciona bien.

Las limitaciones del modelo centralizado se manifiestan cuando la escala crece. Todo el tráfico de datos de todos los nodos debe pasar por las interfaces de red y el controlador de disco de un solo servidor. Cuando decenas de nodos intentan leer archivos genómicos de varios gigabytes simultáneamente, el servidor se satura: la latencia aumenta, el ancho de banda por nodo se reduce y los procesadores quedan inactivos esperando datos. El problema no se limita a la transferencia de archivos grandes; las aplicaciones bioinformáticas frecuentemente generan miles de archivos pequeños y realizan operaciones intensivas sobre metadatos (listar directorios, verificar permisos, gestionar bloqueos), lo que sobrecarga al servidor incluso cuando

el volumen total de datos transferidos es bajo. A nivel de disponibilidad, el servidor NFS constituye un punto único de falla: si se apaga o falla, todo el clúster pierde acceso a los datos y los trabajos en ejecución se corrompen o fallan. La escalabilidad es vertical: cuando se agota la capacidad, la única opción es reemplazar el servidor por uno más grande. Aunado a lo anterior, también hay que migrar todos los datos, resultando en un proceso caro

Los sistemas de almacenamiento distribuido abordan estas limitaciones con un enfoque opuesto. En lugar de concentrar los datos en un solo servidor, los distribuyen entre múltiples nodos de almacenamiento. Cuando un archivo ingresa al sistema, se fragmenta en bloques que se reparten entre los nodos disponibles. Desde la perspectiva del usuario, el sistema presenta un único namespace unificado, un árbol de directorios que se ve y se comporta como un disco local pero, internamente los datos están dispersos en la infraestructura.

Esta arquitectura ofrece tres ventajas directas frente al modelo centralizado:

- **Escalabilidad horizontal.** El rendimiento y la capacidad del sistema crecen al agregar más nodos de almacenamiento, sin necesidad de reemplazar el hardware existente.
- **Tolerancia a fallos.** Los datos se replican o se protegen con códigos de borrado (erasure coding), de modo que la falla de un disco o incluso de un nodo completo no causa pérdida de datos ni interrupción del servicio.
- **Flexibilidad de hardware.** Un sistema distribuido combina dentro de la misma infraestructura lógica distintos tipos de dispositivos: discos mecánicos HDD para capacidad, unidades de estado sólido SSD para metadatos y dispositivos NVMe para rendimiento, asignando los datos según su frecuencia de acceso.

En el caso de este clúster, los dispositivos NVMe se emplean para almacenar los metadatos y los índices de los discos HDD, aunque también cuentan con particiones destinadas al almacenamiento directo de datos y a su uso como espacio temporal (scratch).

El compromiso entre ambos enfoques es una decisión de ingeniería que depende del contexto. La infraestructura distribuida impone una complejidad operativa significativa: configurar la red, ajustar parámetros de distribución de datos, monitorear la salud de decenas de dispositivos y planificar la recuperación ante fallos requiere conocimientos especializados y tiempo de

administración continuo. En entornos donde las cargas de trabajo son ligeras y el volumen de datos es manejable, NFS sigue siendo la solución pragmática.

Sin embargo, cuando las necesidades del clúster incluyen el procesamiento concurrente de datos genómicos a gran escala, la persistencia de decenas o cientos de terabytes y el acceso simultáneo de múltiples usuarios e investigadores, las limitaciones del modelo centralizado se convierten en un obstáculo para la investigación. En este punto, la complejidad del almacenamiento distribuido se justifica como una inversión necesaria para sostener las demandas computacionales del proyecto.

2.6 Conceptos base de almacenamiento distribuido (objetos, bloques y archivos)

Un sistema de almacenamiento distribuido puede presentar los datos a sus usuarios usando distintas interfaces o abstracciones. No se trata de tecnologías de almacenamiento físico separadas, sino de tres formas distintas de acceder a los mismos datos subyacentes. Hay tres modelos principales: almacenamiento de objetos (la abstracción más plana), almacenamiento de bloques y almacenamiento de archivos (Johnson, 2025).

El almacenamiento de objetos organiza los datos como unidades planas, cada una identificada por una clave única y acompañada de metadatos descriptivos, sin jerarquía de directorios. El acceso se realiza a través de una interfaz programática, típicamente una API REST compatible con protocolos como Amazon S3 u OpenStack Swift. Los objetos encajan con datos que se escriben una vez y se leen muchas veces. En la arquitectura de Ceph, esta capacidad es provista por RADOS Gateway (RGW), que ofrece compatibilidad nativa con ambos protocolos. Su principal limitación es que las aplicaciones deben estar diseñadas para interactuar con la API específica; un programa convencional que espera abrir archivos usando rutas de directorio no puede usar almacenamiento de objetos sin modificaciones en su código.

El almacenamiento de bloques presenta volúmenes de disco virtual que el sistema operativo reconoce y trata como si fueran dispositivos de almacenamiento físicos. Sobre estos volúmenes se puede instalar cualquier sistema de archivos convencional (ext4, XFS) y utilizarlos como almacenamiento local dedicado. Los bloques sirven para máquinas virtuales, orquestadores de contenedores y bases de datos, que requieren acceso de baja latencia a nivel de bloques individuales. En Ceph, esto se logra con RBD (RADOS Block Device), que soporta operaciones avanzadas como instantáneas (snapshots) y clones con semántica de copia en escritura (copy-

on-write), que hace posible crear copias eficientes de los datos sin duplicar el espacio de almacenamiento (Johnson, 2025).

El almacenamiento de archivos proporciona un sistema de archivos distribuido completo con jerarquía de directorios, permisos, marcas de tiempo y compatibilidad con el estándar POSIX. Este estándar define el comportamiento esperado de un sistema de archivos convencional: las operaciones de abrir, leer, escribir y cerrar archivos, la gestión de bloqueos concurrentes y la evaluación de permisos de acceso. Un sistema de archivos que cumple con POSIX se monta en los nodos del clúster de la misma forma que un disco local, de modo que cualquier aplicación existente puede interactuar con él sin necesidad de adaptaciones en su código fuente. Las operaciones de metadatos, como listar directorios, verificar permisos o renombrar archivos son gestionadas por servidores de metadatos dedicados, mientras que la transferencia de datos se realiza directamente entre el cliente y los nodos de almacenamiento (Johnson, 2025). En Ceph, esta interfaz es provista por CephFS.

La plataforma Ceph integra estos tres modelos sobre una misma capa de almacenamiento subyacente denominada RADOS (Reliable Autonomic Distributed Object Store). RADOS organiza todos los datos como objetos distribuidos entre los nodos del clúster por medio de el algoritmo CRUSH (Controlled Replication Under Scalable Hashing), una función determinista que calcula la ubicación de cada objeto sin necesidad de tablas de asignación centralizadas (Johnson, 2025). Sobre esta capa base, Ceph ofrece simultáneamente las tres interfaces descritas: RGW para almacenamiento de objetos, RBD para almacenamiento de bloques y CephFS para almacenamiento de archivos. Tener las tres interfaces sobre la misma base hace que un clúster académico use el mismo hardware para distintos propósitos sin duplicar sistemas.

En el contexto de un clúster HPC orientado a bioinformática, la interfaz de archivos (CephFS) es la opción natural. Las herramientas de análisis genómico como PLINK, bcftools, GATK o los entornos de R y Python están diseñadas para leer datos desde rutas de directorio convencionales y escribir resultados en ubicaciones del sistema de archivos. CephFS hace que estas herramientas operen sin modificación alguna, mientras que, por debajo, RADOS se encarga de distribuir los datos, replicarlos y protegerlos contra fallas.

3. Análisis de requerimientos

3.1 Descripción del entorno computacional inicial

Al inicio de este proyecto, la infraestructura computacional disponible para la realización de trabajos consistía en una única estación de trabajo compartida por múltiples usuarios. Este equipo estaba basado en un procesador AMD Ryzen Threadripper 3960X de 24 núcleos y 48 hilos a 3.8 GHz, 256 GB RAM, una tarjeta gráfica NVIDIA GeForce RTX 3060 con 12 GB de VRAM, y un arreglo de almacenamiento compuesto por dos discos duros mecánicos de 20 TB cada uno, un disco de 4 TB destinado al sistema operativo y un disco adicional de 4 TB. Los volúmenes de almacenamiento estaban configurados con una capa de abstracción de tipo volume group (LVM), sin separación lógica entre los datos de los distintos usuarios ni entre datos de trabajo y datos de preservación a largo plazo.

No existía un gestor de recursos ni un planificador de tareas. Los investigadores debían coordinarse entre sí informalmente por medio de mensajes o turnos acordados verbalmente para evitar lanzar tareas simultáneamente y saturar los recursos de la máquina. Esta situación provocaba tiempos de espera impredecibles y conflictos frecuentes cuando dos o más usuarios necesitaban ejecutar análisis que demandaban una proporción significativa de los recursos disponibles.

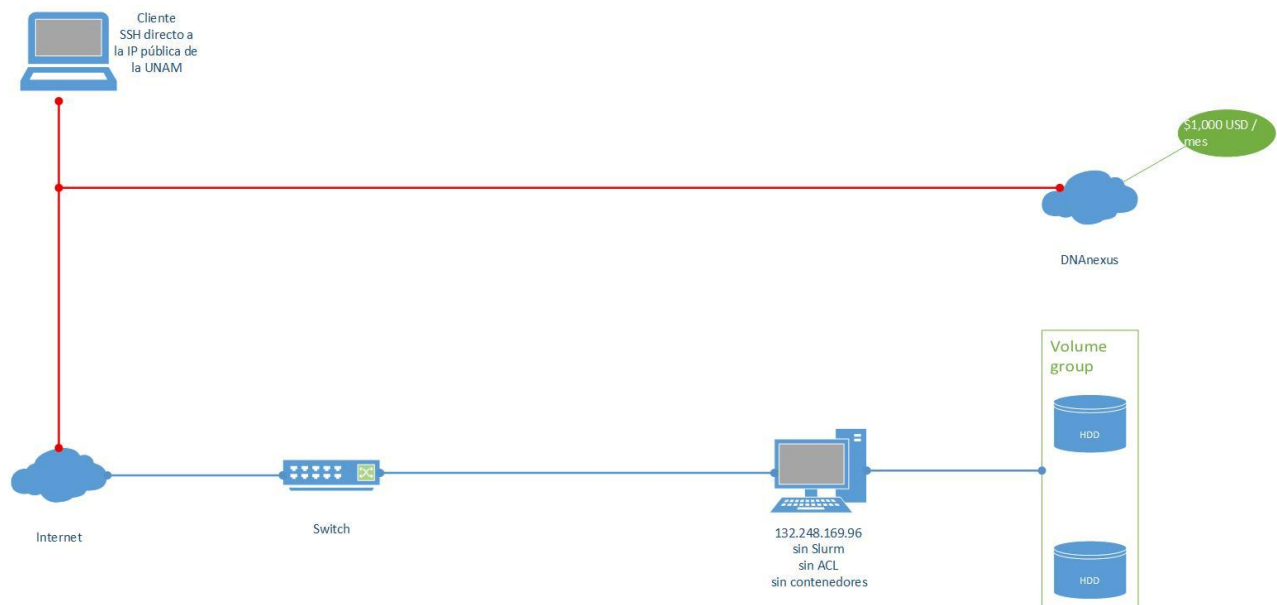


Figura 2. Arquitectura del entorno computacional inicial.

El manejo de software presentaba dificultades adicionales. No se utilizaban tecnologías de contenedores, por lo que las herramientas bioinformáticas se instalaban directamente en el sistema operativo. Cuando un investigador requería una versión específica de una biblioteca o un intérprete de Python o R para su pipeline, la instalación podía entrar en conflicto con las dependencias de otros usuarios. No existían listas de control de acceso (ACL) sobre los archivos, de modo que cualquier usuario podía leer, modificar o eliminar archivos de otros proyectos sin restricción alguna.

La administración del sistema recaía en el jefe de bioinformática, quien asumía esta responsabilidad además de sus funciones de investigación. Dado el volumen de tareas que esta persona atendía simultáneamente, la disponibilidad y el mantenimiento del equipo quedaban frecuentemente relegados a un segundo plano. No había documentación formal sobre los procedimientos de uso del sistema, ni un mecanismo de atención a incidencias al que los usuarios pudieran recurrir cuando el equipo presentaba fallos o problemas de configuración.

Los usuarios principales de esta infraestructura eran investigadores del Estudio Prospectivo de la Ciudad de México (MCPS), en su mayoría estudiantes de doctorado. Sus cargas de trabajo incluían análisis de asociación del genoma completo (GWAS), imputación de genotipos, cálculo de puntajes de riesgo poligénico y otros análisis estadísticos sobre datos genómicos a escala poblacional. Estas tareas requerían procesamiento sostenido durante horas o días, grandes volúmenes de memoria RAM y acceso intensivo a los discos de almacenamiento.

3.2 Problemáticas técnicas identificadas

La estación de trabajo presentaba problemas técnicos derivados tanto de las limitaciones del hardware como de la ausencia de mecanismos de control sobre los recursos compartidos.

Uno de los problemas más frecuentes era la sobreasignación de recursos de cómputo. Varias herramientas bioinformáticas detectan automáticamente los componentes del equipo donde se ejecutan y asignan recursos en consecuencia, por ejemplo, utilizando todos los hilos de procesamiento disponibles. Cuando dos o más usuarios lanzaban trabajos simultáneamente, cada programa intentaba apropiarse de la totalidad de los 48 hilos del procesador, lo que provocaba contención severa de CPU. A esto se sumaba que la mayoría de los investigadores no tenían formación en administración de sistemas ni experiencia en la estimación de recursos computacionales, por lo que ejecutaban sus tareas sin especificar límites de memoria o hilos.

La acumulación de procesos concurrentes sin control de recursos provocaba que la estación se apagara sin aviso en múltiples ocasiones, interrumpiendo trabajos que llevaban horas o días de ejecución.

El almacenamiento presentaba dos problemas críticos. El primero era operativo: al no existir separación de espacios ni cuotas por usuario, los archivos de uso general eran eliminados o sobrescritos cuando un investigador necesitaba liberar espacio para sus propios datos. Esta dinámica generaba conflictos constantes, ya que al borrar archivos para dar cabida a un proyecto se eliminaban datos que otro usuario aún necesitaba. El segundo problema era estructural: los discos estaban configurados bajo una capa de abstracción de tipo volume group (LVM) sin redundancia. Si un disco físico fallaba, se perdía toda la información contenida en el volumen, incluyendo datos de todos los usuarios y proyectos. No existían respaldos formales ni mecanismos de recuperación ante fallos de hardware.

DNAnexus, la plataforma en la nube utilizada para ejecutar análisis especializados, opera bajo un modelo de cobro por tiempo de ejecución y recursos solicitados. El presupuesto asignado al grupo de investigación era de 1,000 dólares mensuales. En al menos una ocasión, este límite se agotó en una sola semana debido a la ejecución simultánea de múltiples trabajos de alto consumo. Al alcanzarse el tope presupuestal, la plataforma canceló automáticamente todos los trabajos en curso de todos los investigadores, lo que resultó en la pérdida del avance computacional acumulado y la imposibilidad de continuar los análisis durante el resto del mes.

3.3 Problemáticas operativas y organizacionales

El proceso de alta de usuarios era informal y no seguía un procedimiento documentado. Los investigadores se acercaban directamente con el jefe de bioinformática para solicitar una cuenta, y la creación podía demorar una semana o más dependiendo de la disponibilidad de esta persona. No existía un formulario de solicitud, un registro de usuarios activos ni un mecanismo para dar de baja cuentas de investigadores que ya no colaboraban con el proyecto.

La comunicación entre los usuarios y el responsable del sistema se realizaba exclusivamente a través de canales informales, principalmente mensajes de WhatsApp o conversaciones verbales. Las solicitudes de soporte quedaban a criterio y memoria del jefe de bioinformática. Con frecuencia las dudas se olvidaban, las respuestas se retrasaban o simplemente no

llegaban. No existía un sistema de seguimiento que permitiera registrar las incidencias reportadas, asignarles prioridad o verificar que fueran atendidas.

No existía documentación de uso del sistema. Los nuevos usuarios aprendían a trabajar con la estación por prueba y error, lo que implicaba una curva de aprendizaje prolongada y un riesgo constante de errores como ejecutar un proceso que consumía todos los recursos del equipo o eliminar archivos de otros proyectos por desconocimiento de la estructura de directorios. Tampoco existían políticas de uso de recursos: no había límites de CPU, memoria o disco por usuario, ni horarios establecidos para la ejecución de trabajos pesados, ni días de mantenimiento planificado. El acceso a los recursos operaba bajo una lógica de "primero en llegar, primero en usar", sin priorización ni equidad.

La gestión de los datos genómicos y epidemiológicos carecía de estructura formal. No existían convenciones sobre dónde almacenar los datos de cada proyecto, qué datos debían preservarse y cuáles podían eliminarse. Tratándose de información derivada de un estudio con participantes humanos como el MCPS, la ausencia de controles de acceso y políticas de manejo de datos representaba una vulnerabilidad operativa que debía resolverse.

3.4 Requerimientos funcionales y no funcionales

Requerimientos funcionales	Descripción
Gestión de trabajos usando Slurm	Los usuarios deben poder enviar trabajos al clúster con scripts de Slurm (sbatch), especificando los recursos requeridos (núcleos, memoria, tiempo). El sistema debe gestionar la ejecución de estos trabajos con una cola de asignación automática de nodos.
Cuentas de usuario individuales con cuotas	Cada investigador debe contar con una cuenta propia en el sistema, con un directorio personal y límites de almacenamiento definidos para evitar que un solo usuario agote la capacidad del clúster.

Almacenamiento compartido visible desde todos los nodos	Todos los nodos de cómputo deben tener acceso al mismo sistema de archivos, de modo que un trabajo pueda ejecutarse en cualquier nodo sin depender de datos almacenados localmente.
Entornos de análisis interactivos vía web	El sistema debe ofrecer interfaces web para entornos de análisis interactivo, incluyendo JupyterHub para Python y RStudio Server para R, accesibles desde el navegador del usuario.
Plataforma de almacenamiento colaborativo	Se debe disponer de un servicio de almacenamiento y colaboración de archivos, que permita a los investigadores compartir datos y documentos con control de accesos.
Sistema de atención a incidencias	Se requiere un sistema de mesa de ayuda accesible usando correo electrónico, administrado con una plataforma de tickets Zammad, que permita registrar, dar seguimiento y resolver las solicitudes de soporte de los usuarios.
Documentación y guía de uso	Se debe contar con una página web que contenga guías de uso del sistema, instrucciones para el envío de trabajos con Slurm, ejemplos de scripts y documentación de las herramientas disponibles.
Ejecución de aplicaciones por medio de contenedores	Las herramientas bioinformáticas deben ejecutarse dentro de contenedores Apptainer para garantizar el aislamiento de dependencias y evitar conflictos entre las

	necesidades de software de distintos usuarios.
Soporte para cargas de trabajo genómicas y epidemiológicas	El clúster debe ser capaz de ejecutar los tipos de análisis documentados en la plataforma DNAnexus utilizada previamente por el grupo de investigación.

Requerimientos no funcionales	Descripción
Operación continua	El clúster debe operar en continuo, con ventanas de mantenimiento programado comunicadas previamente a los usuarios. La caída de un nodo de cómputo no debe afectar la operación del resto del sistema.
Aislamiento entre usuarios	Los datos de cada usuario deben estar protegidos con permisos del sistema operativo y listas de control de acceso (ACL). Ningún usuario debe poder leer, modificar o eliminar archivos de otro usuario sin autorización explícita.
Acceso externo restringido	El acceso al clúster desde redes externas debe realizarse exclusivamente a través del nodo maestro, usando conexiones SSH autenticadas por llave pública. Los nodos de cómputo no deben ser accesibles directamente desde internet.
Capacidad de procesamiento concurrente	El sistema debe permitir la ejecución simultánea de múltiples trabajos de distintos usuarios, con la asignación controlada de recursos por parte del gestor de trabajos.

Crecimiento horizontal	La arquitectura del clúster debe permitir la incorporación de nuevos nodos de cómputo y almacenamiento sin requerir la reinstalación del sistema ni la migración de datos existentes.
Inmutabilidad de entornos	Los análisis ejecutados con contenedores deben ser reproducibles en el tiempo. Las imágenes de contenedor deben preservarse en su versión original para permitir la replicación exacta de los resultados.
Protección contra pérdida de datos	El sistema de almacenamiento debe tolerar la falla de al menos un disco sin pérdida de información, con mecanismos de replicación o redundancia.
Administración con personal reducido	El sistema debe poder ser administrado por una sola persona con conocimientos de administración de sistemas Linux.

3.5 Requerimientos de almacenamiento y manejo de datos (capacidad, acceso multiusuario, disponibilidad)

Los datos del MCPS incluyen genotipos de microarreglo en formato PLINK (archivos .bed, .bim, .fam) correspondientes al Freeze 150 con aproximadamente 150,000 participantes, datos genómicos imputados con el panel TOPMed en formatos BGEN, PLINK y VCF segmentados por cromosoma, y datos de secuenciación de genoma completo (WGS) almacenados como archivos pVCF comprimidos. A estos se suman bases de datos de referencia que los pipelines bioinformáticos consultan de forma recurrente: el genoma de referencia humano GRCh38 con sus índices de alineamiento, anotaciones funcionales de GENCODE v49, frecuencias poblacionales de dbSNP, predicciones funcionales de dbNSFP5 y datos de expresión cuantitativa de tejidos del proyecto GTEx v10. En conjunto, estas bases de datos de referencia comprenden miles de archivos. Solo los datos de GTEx superan los 6,700 archivos en formato Parquet que los procesos de análisis leen intensivamente y en paralelo.

La capacidad total de almacenamiento requerida debía contemplar no solo los datos existentes sino también el crecimiento proyectado. Los datos del MCPS y las bases de referencia ocupaban varias decenas de terabytes al momento del diseño, y las actividades de análisis generan continuamente archivos intermedios (matrices de desequilibrio de ligamiento, resultados de faseado, archivos temporales de imputación) que pueden duplicar o triplicar el volumen de los datos originales durante las fases activas de procesamiento. Se estimó que el sistema debía proveer una capacidad bruta de al menos 200 TB para operar sin restricciones de espacio en el mediano plazo.

Los requerimientos de acceso concurrente se derivan del modelo multiusuario del clúster. Múltiples trabajos de Slurm ejecutándose en distintos nodos necesitan leer simultáneamente los mismos archivos de referencia genómica por ejemplo, el archivo del genoma de referencia hg38.fa o los genotipos del Freeze 150 sin que el acceso de un trabajo degrade el rendimiento de los demás. Del mismo modo, los resultados de cada trabajo deben escribirse en ubicaciones accesibles desde cualquier nodo, sin depender del almacenamiento local de la máquina donde se ejecutó el proceso.

El sistema de almacenamiento debía soportar perfiles de acceso heterogéneos. Los datos del MCPS en formato PLINK y BGEN se leen secuencialmente en bloques grandes durante los análisis de asociación, lo que demanda ancho de banda sostenido. Los datos de GTEx en formato Parquet se consultan a través de lecturas selectivas por columna y cromosoma, lo que genera un patrón de acceso más aleatorio con alta demanda de operaciones por segundo (IOPS). Los archivos temporales de los pipelines de imputación y faseado requieren escrituras rápidas de archivos grandes seguidas de lecturas inmediatas en el mismo nodo o en otro. Estas demandas requerían un subsistema de almacenamiento capaz de ofrecer tanto capacidad como rendimiento al mismo tiempo.

En cuanto a la disponibilidad y durabilidad, los datos del MCPS son irremplazables. Proviene de un proceso de secuenciación y genotipado que no puede repetirse. La pérdida de estos datos por falla de un disco representaría un daño irreversible para el proyecto. El sistema de almacenamiento debía tolerar la falla de al menos un dispositivo sin pérdida de información y sin interrupción del servicio, usando algún mecanismo de replicación o redundancia.

Respecto al manejo de permisos, se requería una estructura de directorios que separara los datos compartidos de uso general (genomas de referencia, bases de anotación, imágenes de contenedores) de los datos privados de cada usuario (resultados de análisis, scripts, archivos temporales). Los datos compartidos debían ser legibles por todos los usuarios, pero modificables solo por el administrador. Los datos de cada usuario debían estar protegidos contra lectura y escritura por parte de otros usuarios, con la posibilidad de establecer espacios de colaboración para proyectos específicos. Se requería también la implementación de cuotas de disco por usuario para prevenir que un solo investigador agotara la capacidad del sistema.

4. Diseño del clúster HPC académico

4.1 Arquitectura lógica del clúster

El clúster está compuesto por cinco nodos interconectados por una red privada. Un nodo maestro (mcps-hpc-01) centraliza la administración del sistema, los servicios web y la conectividad con el exterior. Los cuatro nodos restantes (MCPS-HPC02 a MCPS-HPC05) funcionan exclusivamente como nodos de cómputo, ejecutando los trabajos asignados por el gestor de recursos Slurm.

El almacenamiento distribuido CephFS está desplegado sobre los cinco nodos, con cinco monitores (MON), dos gestores (MGR) uno activo y uno en espera, y tres servidores de metadatos (MDS) dos activos y uno en espera. Esta distribución da a todos los nodos acceso al mismo sistema de archivos sin necesidad de replicar los datos localmente en cada máquina, optimizando la capacidad total de almacenamiento disponible.

El flujo de trabajo de un usuario sigue una ruta definida. El investigador entra al nodo maestro por VPN y una conexión SSH autenticada por llave pública. Una vez dentro, redacta un script de trabajo y lo envía a Slurm con sbatch. Slurm verifica la calidad de servicio (QoS) asignada al usuario para determinar si los recursos solicitados están dentro de sus límites permitidos. Si existe un nodo con los recursos disponibles, Slurm lanza el trabajo en ese nodo; en caso contrario, el trabajo entra a una cola de espera. Durante la ejecución, el nodo de cómputo lee los datos de entrada directamente desde CephFS y escribe los resultados en el mismo sistema de archivos. Al finalizar, los resultados quedan accesibles desde cualquier nodo del clúster sin necesidad de transferencias manuales.

Los servicios web del clúster; JupyterHub, RStudio Server, Nextcloud y Zammad, están alojados en el nodo maestro como contenedores Podman. El acceso externo a estos servicios pasa por una cadena de proxy inverso: las solicitudes llegan a un servidor EC2 de AWS que actúa como punto de entrada público con certificados TLS, y este las redirige a través de un túnel WireGuard hacia el nodo maestro, donde un segundo servidor Caddy las distribuye al contenedor correspondiente. Este mismo servidor EC2 aloja el servicio de correo electrónico institucional bajo el dominio mcps-epcm.org, gestionado a través de Route 53 de AWS. En la figura 3 se muestra panorama general de la arquitectura lógica del clúster.

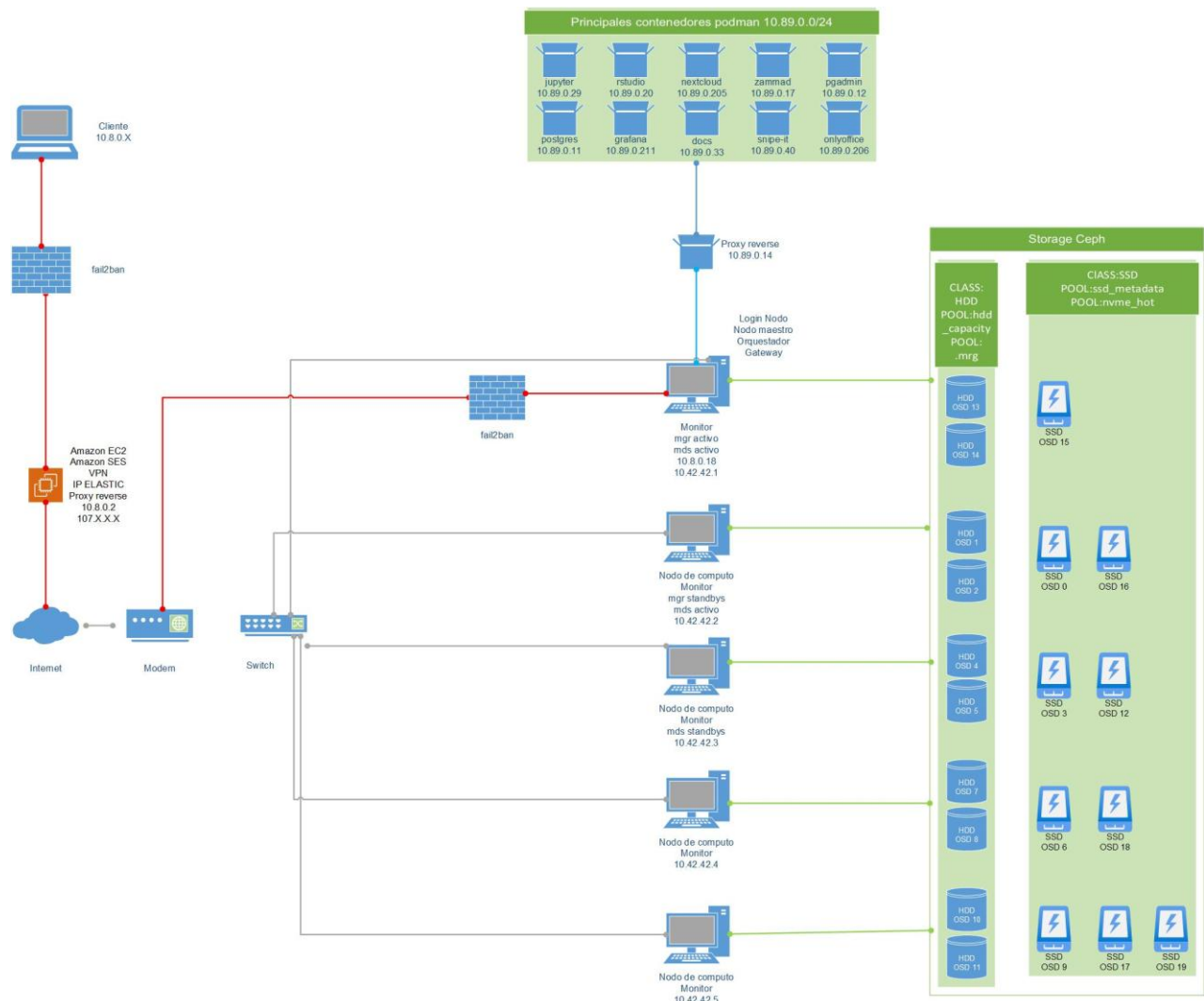


Figura 3. Arquitectura del clúster MCPS-HPC

Cuando se requiere añadir un nuevo servicio web, el procedimiento consiste en crear un subdominio (por ejemplo, `jupyter.mcps-epcm.org`) en Route 53. Si el servicio es interno, el

subdominio se resuelve a la dirección del nodo maestro en la VPN (10.8.0.18). Si el servicio debe ser público, se resuelve a la IP elástica de EC2, y el proxy inverso de Caddy se encarga de enrutar la solicitud hasta el contenedor correspondiente en el nodo maestro.

4.2 Roles de los nodos y modelo de operación

El nodo maestro (mcps-hpc-01) cumple tres funciones simultáneas: actúa como puerta de enlace (gateway) para proveer acceso a internet a los nodos de cómputo, aloja todos los servicios web contenerizados, y ejecuta los demonios de control de Slurm y Ceph. Este nodo no está configurado como nodo de cómputo de Slurm. En una fase inicial del proyecto sí lo estaba, pero cuando Slurm asignaba trabajos pesados al nodo maestro, el rendimiento de los servicios web se degradaba notoriamente. Dado que los usuarios pueden ser descuidados al estimar los recursos que requieren sus trabajos y podrían provocar que el nodo dejara de responder, se tomó la decisión de excluirlo de las particiones de cómputo. Los servicios interactivos como JupyterHub y RStudio Server sí consumen recursos del nodo maestro, pero al no pasar por Slurm, sus recursos se asignan de forma dinámica y su impacto es controlable. Se instruye a los usuarios para que utilicen estas herramientas como entornos de desarrollo y depuración, y que reserven los trabajos pesados para enviarlos con sbatch a los nodos de cómputo.

Los nodos de cómputo (MCPS-HPC02 a MCPS-HPC05) están dedicados exclusivamente a la ejecución de trabajos de Slurm. Las especificaciones de cada nodo se detallan en la siguiente tabla:

Nodo	CPU	GPU	RAM	Dispositivos de almacenamiento
1	AMD Ryzen Threadripper 3960X 24-Core Processor Unknown CPU @ 3.8GHz Hilos 48	NVIDIA GeForce RTX 3060 12GB VRAM	256GB	2 x 20TB 1 x 4TB 1 x 4TB SO

2	Intel(R) Core(TM) Ultra 9 285K To Be Filled By O.E.M. CPU @ 5.4GHz Hilos 24	NVIDIA GeForce RTX 3060 12GB VRAM	196GB	2 x 20TB 2 x 4 TB 1 x 4TB SO
3	Intel(R) Core(TM) i9- 14900KF To Be Filled By O.E.M. CPU @ 5.7GHz Hilos 32	NVIDIA GeForce RTX 3060 12GB VRAM	196GB	2 x 20TB 2 x 4 TB 1 x 4TB SO
4	Intel(R) Core(TM) Ultra 9 285 To Be Filled By O.E.M. CPU @ 4.7GHz Hilos 24	NVIDIA GeForce RTX 3060 12GB VRAM	196GB	2 x 20TB 2 x 4 TB 1 x 4TB SO
5	Intel(R) Core(TM) Ultra 9 285 To Be Filled By O.E.M. CPU @ 4.5GHz Hilos 24	NVIDIA GeForce RTX 3060 12GB VRAM	196GB	2 x 20TB 2 x 4 TB 1 x 1TB 1 x 4TB SO

Slurm organiza los nodos de cómputo en dos particiones: normal (para trabajos de CPU) y gpu (para trabajos que requieren aceleración por GPU). Para controlar el uso de recursos, se definieron seis niveles de calidad de servicio (QoS) que limitan la cantidad de CPUs, memoria, tiempo de ejecución y trabajos simultáneos que cada usuario puede solicitar:

Qos	CPUs Máx.	Memoria Máx.	Tiempo Máx.	Jobs Simultáneos
small	8	32	Sin límite	20
normal	24	150	2 días	10

interactive	16	64	12 horas	2 (+ 1 GPUs)
gpu	48	128	3 días	5 (+2 GPUs)
large	96	256	7 días	Sin límite
vip	96	512	Sin límite	Sin límite (+ 5 GPUs)

El nivel small y normal están diseñado para usuarios en proceso de aprendizaje, limitando los recursos para evitar que un error afecte al clúster. Los niveles superiores se asignan a investigadores con experiencia según las necesidades de sus proyectos. El nivel vip está reservado para cargas de trabajo críticas que requieren acceso prioritario al máximo de recursos disponibles.

4.3 Infraestructura de red

Los cinco nodos están interconectados mediante una red interna con el segmento 10.42.42.0/24. La red opera a 10 Gbps sobre conexiones de fibra óptica SFP+ con un switch administrado; una mejora respecto a la configuración inicial de 1 Gbps con cable de cobre, que se migró al detectar saturación del enlace durante la ejecución concurrente de pipelines genómicos sobre CephFS. El nodo maestro cuenta con dos interfaces de red físicas: una conectada al módem del proveedor de internet (enp68s0), y otra conectada a la red interna del clúster (eth10g), configurada con un MTU de 9000 bytes (jumbo frames) para aprovechar el ancho de banda de la fibra. El nodo maestro funciona como puerta de enlace NAT para los nodos de cómputo, que no tienen acceso directo a internet. Todo el tráfico saliente de los nodos de cómputo se enruta a través del nodo maestro.

Los contenedores Podman que alojan los servicios web operan en una red virtual independiente denominada mcps-net, con el segmento 10.89.0.0/24. Esta separación evita conflictos con las reglas de iptables y con el sistema de resolución de nombres de Netavark. Cada contenedor lleva un alias de red definido en su archivo de servicio systemd, lo que elimina la necesidad de un servidor DNS interno para la comunicación entre contenedores.

La resolución de nombres interna del clúster sale de un archivo de hosts en el nodo maestro que asocia las direcciones IP con los nombres de los nodos y con el dominio de correo electrónico (mail.mcps-epcm.org, apuntando a la IP elástica de EC2). La gestión remota de los

nodos de cómputo se hace con ClusterShell (clush), que ejecuta comandos en todos los nodos a la vez desde el nodo maestro.

4.4 Modelo de acceso multiusuario

El procedimiento de alta de un nuevo usuario sigue un flujo estandarizado en cuatro fases. Primero, se crea la cuenta en el nodo maestro con un UID y GID explícitos para garantizar la paridad de identidades en todos los nodos del clúster. Si se permitiera la asignación automática de UID, las ACL y los montajes de CephFS fallarían cuando Slurm despache un trabajo a un nodo de cómputo que no reconozca al usuario. Segundo, se crea el directorio home del usuario en CephFS (/mnt/cephfs/orgs/home/) con propiedad asignada a www-data:nextcloudfs debe ser www-data para permitir que nextcloud pueda crear y borrar archivos, y el bit SetGID activado, lo que garantiza que los archivos creados hereden el grupo correcto para la integración con Nextcloud. Tercero, se configuran las ACL para otorgar al usuario control total sobre su directorio y se establece una cuota de almacenamiento de 500 GB en CephFS. Cuarto, se crea un buzón de correo institucional y se generan las llaves SSH del usuario.

Una vez completado el alta en el nodo maestro, la identidad del usuario se propaga a los nodos de cómputo con scripts de sincronización ejecutados con ClusterShell. Estos scripts replican el UID, GID y el hash de contraseña en los nodos MCPS-HPC02 a MCPS-HPC05, garantizando que Slurm pueda despachar trabajos del nuevo usuario a cualquier nodo sin errores de identidad huérfana.

4.5 Consideraciones de seguridad y gestión de usuarios

En la fase inicial del proyecto, el clúster operaba con una dirección IP pública proporcionada por la UNAM. Esta conexión presentaba latencia elevada y cortes frecuentes del servicio. Se detectaron intentos sostenidos de acceso por fuerza bruta contra el servicio SSH, lo que evidenció la necesidad de un modelo de acceso externo más robusto.

Para resolver estos problemas se implementó un enlace VPN WireGuard entre el nodo maestro y una instancia EC2 de AWS. Esta instancia actúa como punto de entrada público estable, con una IP elástica fija y certificados TLS gestionados por Caddy. Las solicitudes externas llegan a EC2 y se reenvían al nodo maestro a través del túnel VPN. El servidor EC2 también aloja el servicio de correo electrónico institucional bajo el dominio mcps-epcm.org, gestionado a través de Route 53 de AWS.

Cuando se requiere añadir un nuevo servicio web, el procedimiento consiste en crear un subdominio (por ejemplo, jupyter.mcps-epcm.org) en Route 53. Si el servicio es interno, el subdominio se resuelve a la dirección del nodo maestro en la VPN (10.8.0.18). Si el servicio debe ser público, se resuelve a la IP elástica de EC2, y la cadena de proxy inverso se encarga de enrutar la solicitud: EC2 Caddy recibe la solicitud con TLS, la reenvía por WireGuard al nodo maestro, donde un segundo servidor Caddy la distribuye al contenedor correspondiente en la red mcps-net.

El acceso al clúster se organiza en tres vías, cada una con su propio mecanismo de autenticación y nivel de exposición:

Vía de acceso	Autenticación	Requiere VPN	Uso
SSH al nodo maestro	Llave privada Ed25519 (contraseña deshabilitada en sshd config)	Sí	Envío de trabajos, administración, acceso a terminal
JupyterHub / RStudio Server	Usuario y contraseña del sistema (PAM)	Sí	Análisis interactivo, depuración de código
Nextcloud	Usuario y contraseña propios (creados manualmente)	No (acceso público vía EC2)	Almacenamiento colaborativo, intercambio de archivos

Los nodos de cómputo no son accesibles desde internet ni desde la VPN. Solo el nodo maestro puede comunicarse con ellos a través de la red interna. Esta restricción se aplica tanto a nivel de red (los nodos de cómputo están detrás del NAT sin puertos expuestos) como a nivel de servicio (SSH en los nodos de cómputo solo acepta conexiones desde la red 10.42.42.0/24).

Se implementó Fail2ban en el nodo maestro con una política estricta: cualquier IP pública que falle un solo intento de conexión SSH queda bloqueada permanentemente. El sistema únicamente confía en la red interna del clúster (10.42.42.0/24) y en la interfaz de loopback (127.0.0.1/8).

La autenticación entre los nodos del clúster para la comunicación interna de Slurm va por Munge, un servicio que genera y valida credenciales criptográficas compartidas. Todos los nodos comparten la misma llave Munge, de modo que el demonio de control de Slurm (slurmctld) en el nodo maestro y los demonios de ejecución (slurmd) en los nodos de cómputo se autentican mutuamente sin transmitir contraseñas por la red.

El acceso administrativo con privilegios de superusuario (sudo) está restringido exclusivamente al administrador del clúster. Los investigadores no tienen acceso a sudo en ningún nodo. La instalación de software a nivel de sistema es responsabilidad del administrador; los usuarios ejecutan sus herramientas dentro de contenedores Apptainer, que no requieren privilegios elevados.

4.6 Diseño del subsistema de almacenamiento distribuido (Ceph)

4.6.1 Rol del almacenamiento en el flujo de trabajo científico

Las herramientas bioinformáticas no tienen integración con objetos o bloques, solo interactúan con la capa de filesystem. Los resultados se escriben en ceph y todos los usuarios tienen su home en ceph. Los usuarios tienen acceso a una página web <https://docs.mcps-epcm.org/> que sirve como documentación para lograr usar slurm con pipiles que sirven meramente como ejemplo y nada más. Los datos genómicos se encuentran dentro de /mnt/cephfs/hot_nvme/. Se aclara que a pesar de que la ruta hace referencia a unidades nvme, el espacio de estas unidades fue rápidamente sobrepasado por la cantidad de información del proyecto, se toma la decisión de que esta ruta almacene en los dispositivos HDD, esto se explica a mayor detalle en la sección 5.5.1.

4.6.2 Componentes y servicios principales considerados

La arquitectura de Ceph desplegada en el clúster se compone de cuatro tipos de demonios distribuidos entre los cinco nodos: monitores (MON), gestores (MGR), servidores de metadatos (MDS) y demonios de almacenamiento de objetos (OSD). La distribución de cada componente responde a los requisitos de tolerancia a fallos, rendimiento y las restricciones de hardware disponible.

Se desplegaron cinco monitores, uno por nodo. Los monitores mantienen el estado global del clúster incluyendo el mapa de OSD, el mapa de ubicación de datos y la membresía del clúster y toman decisiones por consenso mediante el algoritmo Paxos. Paxos requiere que una mayoría de monitores esté operativa para que el clúster funcione; esta mayoría se denomina quórum. Con cinco monitores, el quórum se alcanza con tres, lo que significa que el clúster tolera la caída simultánea de hasta dos nodos sin perder la capacidad de operar (Johnson, 2025). Se eligieron cinco en lugar de tres para maximizar la tolerancia a fallos en un clúster donde los nodos son también nodos de cómputo y podrían reiniciarse por mantenimiento o por fallas de hardware. Con tres monitores, en

cambio, la caída de dos nodos detendría todo el clúster; un escenario plausible si se actualiza el sistema operativo de dos máquinas al mismo tiempo.

Se configuraron dos gestores (MGR), uno activo en el nodo maestro y otro en espera (standby) en un nodo de cómputo. Los gestores proveen funciones de administración como la recolección de telemetría, la interfaz de monitoreo y la integración con herramientas como Prometheus y Grafana. Al mantener un MGR en espera, si el activo falla, el de respaldo asume el rol automáticamente sin intervención del administrador (Johnson, 2025).

Se desplegaron tres servidores de metadatos (MDS), dos activos y uno en espera. Los MDS gestionan las operaciones de metadatos de CephFS; listado de directorios, verificación de permisos, renombrado de archivos y son indispensables para que el sistema de archivos funcione con semántica POSIX. Se optó por dos MDS activos para distribuir la carga de operaciones de metadatos. Cuando múltiples trabajos de Slurm acceden simultáneamente al mismo directorio de datos genómicos, CephFS puede repartir la atención de subdirectorios entre ambos MDS, evitando que uno solo se convierta en cuello de botella. El tercer MDS en espera garantiza que si uno de los activos falla, el servicio de metadatos continúe sin interrupción (Johnson, 2025).

El clúster cuenta con 20 OSD, distribuidos entre los cinco nodos. Cada nodo aporta sus discos como OSD individuales a la agrupación lógica de almacenamiento (pool). Las reglas CRUSH clasifican los OSD en dos clases de dispositivo: hdd para los discos mecánicos y ssd para los dispositivos NVMe. Esta clasificación hace posible crear pools de almacenamiento con características de rendimiento distintas sobre el mismo clúster.

Se definieron cuatro pools de almacenamiento:

Pool	Clase de dispositivo	Propósito
.mgr	HDD	Datos internos de gestión de Ceph (telemetría, módulos del MGR)
ssd_metadata	SSD/NVMe	Metadatos de CephFS (journals de los MDS,

		inodos, entradas de directorio)
hdd_capacity	HDD	Datos de archivos de CephFS (genotipos, resultados de análisis, archivos de usuario, datos genómicos)
nvme_hot	SSD/NVMe	Datos de acceso frecuente

La separación del pool de metadatos (`ssd_metadata`) en dispositivos NVMe es una decisión de rendimiento crítica. Las operaciones de metadatos; listar directorios con miles de archivos, verificar permisos, gestionar bloqueos, generan un volumen alto de operaciones pequeñas y aleatorias (IOPS) que los discos mecánicos resuelven con lentitud debido al movimiento físico de sus cabezales. Al colocar los metadatos en NVMe, que opera sin partes móviles y accede directamente al bus PCIe, estas operaciones se resuelven en microsegundos en lugar de milisegundos. Sin esta separación, un simple comando `ls` en un directorio con miles de archivos genómicos podría tardar minutos que es el problema que se describió en la sección 2.5 como limitación del almacenamiento centralizado.

Cada disco HDD tiene además una partición de su NVMe local asignada como BlueStore DB. BlueStore es el backend de almacenamiento de los OSD, y su DB almacena los índices internos y el journal de transacciones de cada OSD. Al colocar esta base de datos en NVMe, las operaciones de escritura en los HDD se confirman más rápidamente porque el journal se escribe en un medio rápido, mientras que los datos en sí se escriben en el HDD asincrónicamente. Esta configuración híbrida hace que los discos mecánicos de gran capacidad operen con latencias de escritura cercanas a las de dispositivos de estado sólido, sin requerir NVMe de la misma capacidad.

4.6.3 Esquema general de acceso y permisos sobre datos

Ruta	Propósito	Permisos
/mnt/cephfs/orgs/home/	Directorios personales de los usuarios (500 GB c/u)	Propietario: <code>www-data:nextcloudfs</code> , <code>SetGID</code> , <code>ACL</code> por usuario
/mnt/cephfs/hot_nvme/	Datos genómicos de referencia y del MCPS (acceso frecuente)	Lectura para todos, escritura solo admin y usuarios autorizados
/mnt/cephfs/hot/	Datos persistentes de servicios (Caddy, PostgreSQL, etc.)	Acceso restringido al grupo gen y al admin
/mnt/cephfs/scratch/	Archivos temporales de los usuarios (sin cuota, sticky bit)	Lectura y escritura para todos, cada usuario responsable de sus datos
/mnt/cephfs/biocontainers/	Imágenes de contenedores Apptainer (.sif)	Ejecución para todos, escritura solo admin

5. Implementación del clúster

5.1 Instalación y configuración del sistema operativo

El sistema operativo seleccionado para todos los nodos del clúster fue Ubuntu Server 24.04 LTS con kernel 6.8.0. La elección inicial había sido Rocky Linux, una distribución derivada de Red Hat Enterprise Linux ampliamente utilizada en entornos HPC. Sin embargo, los nodos 2 a 5, equipados con procesadores Intel Core Ultra 9 de generación reciente y con Intel Core décimo cuarta generación, presentaban congelamientos aleatorios bajo Rocky Linux que resultaron ser incompatibilidades entre el kernel de esa distribución y el hardware de última generación. Se migró a Ubuntu Server por dos razones: su kernel incluía soporte más actualizado para el hardware disponible y su comunidad de usuarios es significativamente más amplia, lo que facilita la resolución de problemas. En la figura 4 se muestran de forma general las fases para la Implementación del sistema.

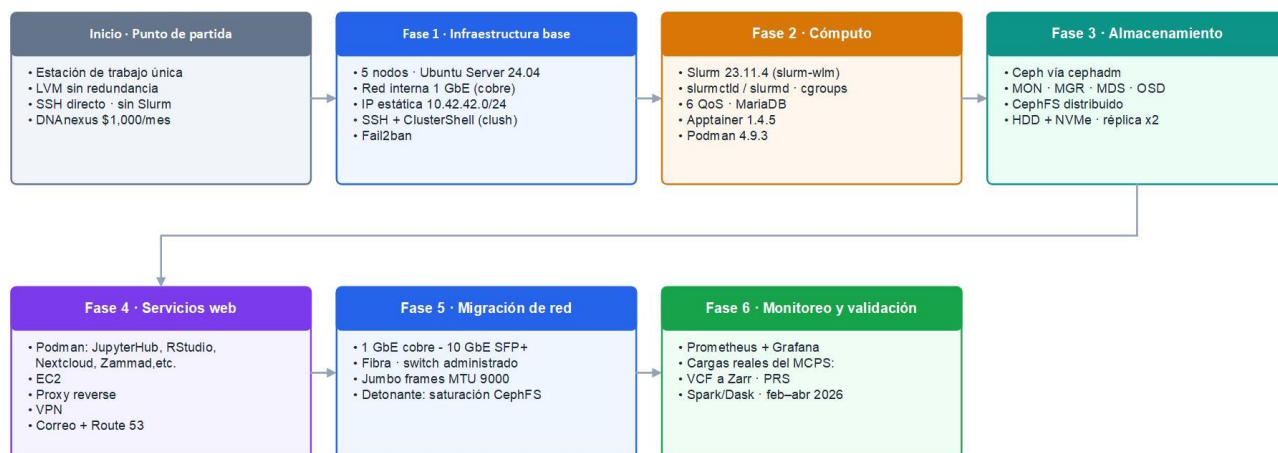


Figura 4. Fases de implementación del clúster

Aun con Ubuntu, los nodos 2 a 5 requerían ajustes en la configuración del BIOS para operar con estabilidad. Los módulos de memoria RAM, al operar con los perfiles XMP habilitados de fábrica, provocaban inestabilidad bajo cargas sostenidas de cómputo. Se aplicó la siguiente configuración en el BIOS de estos nodos:

Parámetro	Valor configurado
AI Overclock Tuner	Manual (XMP deshabilitado)
Memory Frequency	4000 MHz
DRAM VDD/VDDQ Voltage	1.25 V
Command Rate	2N
PCIe ASPM	Deshabilitado
C-States	Deshabilitados

La deshabilitación de XMP y la fijación manual de la frecuencia de memoria a 4000 MHz garantizaron la estabilidad del sistema bajo carga prolongada. Los modos de ahorro de energía PCIe ASPM y los C-States del procesador se desactivaron porque introducían latencias variables en las operaciones de entrada/salida, particularmente en la comunicación con los dispositivos NVMe y la red. Se evaluó inicialmente un perfil de rendimiento agresivo (performance mode), pero esta configuración causó inestabilidad severa que terminó corrompiendo el sistema operativo, por lo que se descartó en favor de los valores conservadores descritos.

El particionamiento de los discos sigue un esquema uniforme en todos los nodos. Cada nodo cuenta con un disco NVMe destinado al sistema operativo, particionado con una partición EFI de 1 GB, una partición de arranque de 2 GB y el resto asignado a un volume group LVM que contiene la raíz del sistema y el directorio /home local. Los discos HDD y los NVMe adicionales se reservan para los OSD de Ceph y no contienen particiones del sistema operativo. El detalle de esta configuración se describe en la sección 5.5.

La configuración de red de cada nodo consiste en una dirección IP estática dentro del segmento 10.42.42.0/24, con el nodo maestro (10.42.42.1) como puerta de enlace predeterminada y servidor DNS. El archivo /etc/hosts del nodo maestro, que contiene las asociaciones entre IP y nombre de cada nodo, se copia a todos los nodos de cómputo para garantizar la resolución de nombres interna sin depender de un servidor DNS.

Para garantizar la homogeneidad del software base entre los nodos, se estableció un procedimiento de replicación desde el nodo maestro. Los paquetes base instalados en todos los nodos incluyen Munge (autenticación para Slurm), las bibliotecas de detección de topología de hardware (libhwloc), el cliente de CephFS (ceph-common), los drivers propietarios de NVIDIA (versión 535) y Java Runtime Environment, OpenJDK (requerido por PySpark). Los binarios de Slurm, sus bibliotecas compartidas y el directorio completo de plugins se copian directamente desde el nodo maestro en lugar de instalarse desde los repositorios en cada nodo, lo que garantiza que todos los nodos ejecuten exactamente la misma versión y compilación de Slurm (23.11.4). Este procedimiento de replicación por copia directa se adoptó tras verificar que la instalación desde paquetes en nodos diferentes podía resultar en versiones o configuraciones ligeramente distintas que causaban fallos de comunicación entre slurmctld y slurmd.

La instalación de los drivers de NVIDIA requiere un paso previo que es desactivar el módulo de kernel de código abierto nouveau con un archivo de blacklist en /etc/modprobe.d/, seguido de la regeneración del initramfs y un reinicio del nodo. Una vez cargado el driver propietario, el dispositivo /dev/nvidia0 queda disponible para que Slurm lo registre como recurso GPU a través de la configuración de gres.conf.

El orden de operaciones durante la instalación de un nodo resultó ser crítico. La experiencia acumulada durante reinstalaciones de nodos (documentada en la sección 8 de este trabajo) demostró que el usuario del sistema slurm (UID 999) debe crearse antes de instalar cualquier

otro paquete, ya que Ubuntu 24.04 asigna ese mismo UID al servicio dnsmasq durante la instalación inicial. Si dnsmasq toma el UID 999 primero, Slurm no puede arrancar. El procedimiento estandarizado para la instalación o reinstalación de un nodo sigue el siguiente orden:

- Configuración de red (IP estática, /etc/hosts)
- Configuración de acceso SSH desde el nodo maestro
- Creación del usuario slurm con UID 999 y GID 988
- Instalación de dependencias (Munge, libhwloc, ceph-common)
- Copia de la llave Munge y reinicio del servicio
- Copia de binarios, bibliotecas y plugins de Slurm
- Copia de archivos de configuración (slurm.conf, cgroup.conf, gres.conf)
- Instalación de drivers NVIDIA y reinicio
- Configuración de CephFS y montaje
- Creación de directorios de Slurm y arranque de slurmd
- Reintegración del nodo al clúster desde el nodo maestro

5.2 Implementación del gestor de recursos y trabajos

Se instaló Slurm 23.11.4 desde el repositorio universe de Ubuntu 24.04 LTS, usando el metapaquete slurm-wlm. No se compiló desde código fuente. La versión 23.11.4-1.2ubuntu5 incluye parches de Canonical para integración con systemd y rutas del sistema Debian, lo que simplifica las actualizaciones y evita conflictos con bibliotecas del sistema operativo.

En el nodo controlador (mcps-hpc-01) se instalaron los paquetes slurmctld, slurmdbd, slurm-client y el conjunto completo de plugins, incluyendo slurm-wlm-mysql-plugin para la comunicación entre slurmdbd y MariaDB. En los nodos de cómputo se instalaron slurmd y slurm-client, distribuidos con ClusterShell:

```
clush -w MCPS-HPC[02-05] "apt install -y slurmd slurm-client slurm-wlm-basic-plugins"
```

En una fase anterior del proyecto, el clúster operaba con binarios de Slurm 24.05.5 compilados en Rocky Linux y portados a mano a Ubuntu copiando los archivos entre nodos. Este esquema funcionaba, pero cada actualización de paquetes del sistema rompía dependencias de

bibliotecas dinámicas. La migración a la versión empaquetada por Ubuntu eliminó ese problema.

El archivo `/etc/slurm/slurm.conf` centraliza toda la configuración del clúster. El nombre del clúster se definió como `mcps-hpc` y el controlador apunta a `mcps-hpc-01` (`10.42.42.1`). Los puertos se cambiaron a valores no estándar (`SlurmctldPort=6871`, `SlurmdPort=6868`) para coexistir con otros servicios que ocupaban los puertos por defecto.

Para el planificador se eligió `sched/backfill`. `Backfill` no solo ejecuta trabajos en orden de prioridad FIFO; también rellena huecos de la cola con trabajos menores que quepan sin retrasar al siguiente en prioridad. Con 104 cores distribuidos en cuatro nodos de cómputo, maximizar la ocupación es importante. Un planificador FIFO puro dejaría cores vacíos mientras trabajos pequeños esperan en cola detrás de uno grande que todavía no tiene hueco.

La selección de recursos usa `select/cons_tres` con `CR_Core_Memory`, de modo que Slurm asigne cores y memoria con granularidad fina en lugar de reservar nodos enteros. Si un trabajo solicita 8 cores y 32 GB, solo esos recursos se marcan como ocupados; el resto queda disponible para otros trabajos en el mismo nodo. La directiva `MaxMemPerNode=160000` en la definición de la partición establece un límite de 160 GB de RAM asignable por nodo, independientemente de la memoria física instalada. Los aproximadamente 30 GB restantes quedan reservados para el sistema operativo, los demonios de Ceph, los procesos de monitoreo y el propio `slurmd`. De esta forma, aunque un usuario solicite toda la memoria disponible de la partición, el nodo conserva un margen operativo que evita condiciones de agotamiento de memoria a nivel del kernel. La directiva `GresTypes=gpu` declara las GPUs NVIDIA como recurso adicional programable; cada nodo tiene una RTX 3060 con 12 GB de VRAM registrada en `gres.conf`.

Hay tres parámetros merecen atención particular. `ReturnToService=2` hace que un nodo reiniciado vuelva automáticamente al estado disponible sin intervención del administrador. En un entorno donde los cortes de energía del edificio son frecuentes, esta directiva ahorra horas de gestión manual. `SlurmdParameters=config_overrides` tolera diferencias menores entre el hardware declarado en `slurm.conf` y lo que `slurmd` detecta al arrancar; los procesadores Intel Core Ultra 9 reportan topologías de `sockets/cores/threads` que varían entre versiones de

firmware, y sin este parámetro slurmd rechaza el arranque. PrivateData=jobs restringe la visibilidad de la cola: cada usuario solo ve sus propios trabajos en squeue.

El rastreo de procesos se apoya en cgroups v2 del kernel de Linux. ProctrackType=proctrack/cgroup garantiza que todos los procesos hijos de un trabajo queden contenidos en el cgroup correcto, y JobAcctGatherType=jobacct_gather/cgroup muestrea uso de CPU y memoria cada 30 segundos. De esta forma, sacct puede reportar cuántos core-seconds y cuánta memoria consumió cada trabajo después de su ejecución.

Slurm exige que todos los nodos lean el mismo slurm.conf. El demonio slurmd extrae del archivo compartido los parámetros que le corresponden según su hostname. La distribución del archivo se realiza con ClusterShell:

```
clush -w MCPS-HPC[02-05] --copy /etc/slurm/slurm.conf --dest /etc/slurm/
```

```
clush -w MCPS-HPC[02-05] --copy /etc/slurm/gres.conf --dest /etc/slurm/
```

```
clush -w MCPS-HPC[02-05] --copy /etc/slurm/cgroup.conf --dest /etc/slurm/
```

El archivo cgroup.conf aplica tres restricciones de aislamiento: ConstrainCores=yes impide que un trabajo use cores no asignados, ConstrainRAMSpace=yes limita la memoria al monto solicitado, y ConstrainDevices=yes bloquea el acceso a GPUs no reservadas. Sin estas directivas, un trabajo que no liberara memoria podría consumir los 190 GB del nodo, provocar que el kernel elimine procesos ajenos (OOM-kill) y dejar a otros usuarios con sus pipelines interrumpidos a mitad de ejecución.

Las definiciones de nodos en slurm.conf declaran los recursos de cada máquina. Los valores de RealMemory se establecen por debajo de la memoria física instalada, reservando un margen para el sistema operativo:

Nodo	CPUs	RealMemory (MB)	GPU
MCPS-HPC02	24	160,000	1× RTX 3060
MCPS-HPC03	32	160,000	1× RTX 3060
MCPS-HPC04	24	160,000	1× RTX 3060
MCPS-HPC05	24	160,000	1× RTX 3060

Munge genera credenciales criptográficas efímeras: el nodo emisor cifra un token con una llave simétrica compartida y el receptor lo descifra con la misma llave. La llave se generó en el controlador con 1024 bytes de /dev/urandom y se distribuyó a los nodos de cómputo con ClusterShell. Los permisos de la llave son 400 con propietario munge:munge.

La verificación se realiza codificando un token en un nodo y decodificándolo en otro:

```
munge -n | ssh MCPS-HPC02 unmunge
```

La salida confirma cifrado AES-128, MAC SHA-256 y un TTL de 300 segundos. Si los relojes de los nodos difieren por más de ese intervalo, Munge rechaza el token.

Un clúster multiusuario sin contabilidad no puede justificar el uso de sus recursos ni detectar ineficiencias. slurmdbd actúa como intermediario entre slurmctld y una base de datos relacional donde se registra cada trabajo: recursos solicitados, recursos consumidos, tiempo de ejecución y código de salida.

La base de datos es MariaDB, ejecutándose dentro de un contenedor Podman en la red mcps-net (10.89.0.144, puerto 3306). Se eligió un contenedor para aislar la base de datos del sistema operativo del nodo controlador y facilitar respaldos. La base se denomina slurm_acct_db.

La directiva AccountingStorageEnforce=associations,limits,qos en slurm.conf activa el cumplimiento estricto: un usuario sin asociación en la base de datos no puede lanzar trabajos, y los límites de QoS se aplican en tiempo real. Los recursos rastreables configurados (AccountingStorageTRES=cpu,mem,gres/gpu) habilitan la consulta con sacct del consumo detallado de CPU, memoria y GPU por trabajo y por usuario.

Las particiones y QoS cuyo diseño se describió en la sección 4.2 se implementaron con sacctmgr. Los usuarios se agrupan bajo la cuenta researchers y cada uno tiene asignadas las QoS a las que puede acceder según su nivel de experiencia y las necesidades de su proyecto:

```
sacctmgr add user angel.pacheco Account=researchers QOS=normal,large,vip
```

La QoS vip se reservó para los administradores del clúster y los investigadores principales que ejecutan pipelines de larga duración. La construcción del panel de referencia MCPS10k con SHAPEIT5 e IMPUTE5, por ejemplo, lanza 23 trabajos por etapa, uno por cromosoma y cada uno puede ocupar un nodo entero durante más de 48 horas.

La verificación se realizó en tres pasos. Primero, confirmar que todos los nodos aparecen en sinfo con estado idle, mixed o allocated. Segundo, enviar un trabajo trivial que reporte el hostname, los cores disponibles y la memoria del nodo:

```
sbatch --wrap="hostname; nproc; free -h; sleep 10" \  
  
--partition=normal --ntasks=1 --cpus-per-task=4
```

Tercero, verificar que el trabajo quedó registrado en la base de datos contable:

```
sacct -j <JOBID> --format=JobID,JobName,State,Elapsed,MaxRSS,NCPUS
```

Si sacct devuelve los campos con datos reales, el ciclo completo funciona, slurmctld aceptó el trabajo, lo planificó según la disponibilidad de recursos, slurmd lo ejecutó dentro del cgroup correspondiente, jobacct_gather/cgroup midió los recursos durante la ejecución, y slurmdbd persistió todo en MariaDB. Cuando algún eslabón de esa cadena falla, sacct devuelve campos vacíos o el trabajo nunca sale del estado PENDING.

5.3 Integración de tecnologías de contenedores

El clúster utiliza dos tecnologías de contenedores con propósitos distintos; Apptainer para la ejecución de aplicaciones científicas en los nodos de cómputo a través de Slurm, y Podman para el despliegue de servicios web en el nodo maestro. Ambas coexisten sin conflicto porque operan en capas diferentes del sistema.

Se instaló Apptainer 1.4.5 en los cinco nodos del clúster desde el PPA oficial del proyecto (ppa:apptainer/ppa), ya que el paquete no está disponible en los repositorios estándar de Ubuntu 24.04. La instalación se distribuyó con ClusterShell:

```
clush -w 10.42.42.[1-5] "add-apt-repository -y ppa:apptainer/ppa"  
  
clush -w 10.42.42.[1-5] "apt update && apt install -y apptainer"
```

Se instaló en todos los nodos, incluyendo el maestro, porque los usuarios necesitan construir imágenes desde el nodo al que se conectan por SSH, y los nodos de cómputo necesitan ejecutarlas cuando Slurm les asigna un trabajo. Apptainer no requiere un demonio ni privilegios de root para ejecutar contenedores, lo que lo hace compatible con el modelo de seguridad descrito en la sección 4.5.

Las imágenes de contenedor de uso general se almacenan en `/mnt/cephfs/biocontainers/images/`, un directorio accesible desde todos los nodos a través de CephFS con permisos de ejecución para todos los usuarios y escritura restringida al administrador. Los archivos de definición (`.def`) utilizados para construir las imágenes se preservan en `/mnt/cephfs/biocontainers/def_files/`. Cuando un usuario construye una imagen para su uso personal, el archivo `.sif` y `.def` resultantes se almacenan en su directorio `home`.

Los servicios web del clúster como JupyterHub, RStudio Server, Nextcloud, Zammad, MariaDB, PostgreSQL, Grafana, Prometheus, entre otros, se ejecutan como contenedores Podman en el nodo maestro. Cada servicio está definido como una unidad de `systemd` (archivos `container-*.service` en `/etc/systemd/system/`) que gestiona el ciclo de vida del contenedor: arranque, reinicio automático ante fallos y parada ordenada. Todos los contenedores operan en la red virtual `mcps-net` (`10.89.0.0/24`) con direcciones IP fijas que se asignan con el parámetro `--ip`, de modo que Caddy los identifique como destinos de su proxy inverso.

Los contenedores Podman dependen de CephFS para la persistencia de sus datos. Cada unidad de servicio incluye una directiva `ExecStartPre` que espera a que el punto de montaje de CephFS esté disponible antes de arrancar el contenedor. Los datos de configuración y estado persistente se almacenan en subdirectorios de `/mnt/cephfs/hot/` por ejemplo, `/mnt/cephfs/hot/caddy/` para la configuración de Caddy, `/mnt/cephfs/hot/mariadb/data/` para la base de datos de MariaDB, y `/mnt/cephfs/hot/zammad-storage/` para los archivos adjuntos del sistema de tickets. Esta estrategia garantiza que si el nodo maestro requiere una reinstalación del sistema operativo, los servicios pueden restaurarse recreando las unidades de `systemd` sin pérdida de datos.

Las credenciales sensibles de los contenedores, contraseñas de bases de datos, llaves API, tokens JWT se gestionan con variables de entorno en los archivos de servicio o con el mecanismo de `secrets` de Podman. Los archivos de servicio tienen permisos restringidos (`600`) para evitar la exposición de estas credenciales.

El clúster mantiene dos flujos de construcción de contenedores según el tipo de imagen.

Para contenedores científicos especializados como `sbayesrc-0.2.6.sif` o `imputation-tools-1.1.sif` se parte de un archivo de definición `.def` que describe la imagen base, las dependencias y las

herramientas a instalar. La construcción se ejecuta con `apptainer build` y produce un archivo `.sif` de solo lectura:

```
apptainer build /mnt/cephfs/biocontainers/images/sbayesrc-0.2.6.sif \  
/mnt/cephfs/biocontainers/def_files/sbayesrc.def
```

Para el contenedor principal del clúster (`jupyter-biotools`), que contiene múltiples entornos de Conda con herramientas de bioinformática y estadística genómica, etc. El flujo es diferente. La imagen base se ejecuta como contenedor Podman en el nodo maestro (el servicio JupyterHub), y las actualizaciones de paquetes se realizan directamente dentro del contenedor en ejecución con `podman exec`. Una vez verificado que los cambios funcionan correctamente, el estado del contenedor se congela como nueva imagen, se exporta a formato tar y se convierte a formato SIF para su uso en los nodos de cómputo a través de Slurm:

```
podman commit jupyterhub localhost/jupyter-biotools:1.5  
  
podman save localhost/jupyter-biotools:1.5 -o /mnt/cephfs/biocontainers/images/jupyter-  
biotools-1.5.tar  
  
apptainer build /mnt/cephfs/biocontainers/images/jupyter-biotools-1.5.sif \  
docker-archive:///mnt/cephfs/biocontainers/images/jupyter-biotools-1.5.tar  
  
rm /mnt/cephfs/biocontainers/images/jupyter-biotools-1.5.tar
```

El flujo: `podman commit` → `podman save` → `apptainer build docker-archive`, hace que la misma imagen sirva tanto para el servicio web interactivo (Podman en el maestro) como para los trabajos de Slurm en los nodos de cómputo (Apptainer). La imagen `jupyter-biotools 1.5` ocupa aproximadamente 80 GB y contiene entornos de Conda para Python (con NumPy, PyArrow, Dask, TensorFlow, JAX), R (con Bioconductor, bigsnpr, qqman), herramientas de alineamiento (BWA, SAMtools, BCFtools) y herramientas de GWAS (REGENIE, PLINK, PLINK2).

Las versiones anteriores de las imágenes se eliminan para conservar espacio de almacenamiento. Dado el tamaño de las imágenes, mantener múltiples versiones simultáneas consumiría cientos de gigabytes de CephFS. La decisión de diseño fue priorizar la capacidad de almacenamiento sobre el historial de versiones, confiando en que los archivos `.def` y el

registro de los comandos de actualización bastan para reconstruir cualquier versión anterior si fuera necesario.

Un trabajo típico que utiliza Apptainer se envía con sbatch y ejecuta herramientas dentro del contenedor con apptainer exec. El acceso a los datos de CephFS requiere un montaje enlazado (bind mount) explícito (--bind /mnt/cephfs:/mnt/cephfs) que hace visible el sistema de archivos distribuido dentro del contenedor. Los datos de entrada de solo lectura y los directorios de salida pueden montarse por separado para mayor control de permisos. Cuando el trabajo requiere un entorno de Conda específico, se invoca directamente el intérprete del entorno mediante su ruta completa dentro del contenedor, sin necesidad de activar el entorno con conda activate:

```
#!/bin/bash

#SBATCH --job-name=gwas_analysis

#SBATCH --account=researchers

#SBATCH --qos=normal

#SBATCH --partition=normal

#SBATCH --nodes=1

#SBATCH --ntasks=1

#SBATCH --cpus-per-task=16

#SBATCH --mem=64G

#SBATCH --time=24:00:00

#SBATCH --output=/mnt/cephfs/orgs/home/%u/logs/%x_%j.out

#SBATCH --error=/mnt/cephfs/orgs/home/%u/logs/%x_%j.err

set -euo pipefail
```

```
SIF=/mnt/cephfs/biocontainers/images/jupyter-biotools-1.5.sif
```

```
DATA=/mnt/cephfs/hot_nvme
```

```
OUT=/mnt/cephfs/orgs/home/$USER/results/$SLURM_JOB_ID
```

```
mkdir -p /mnt/cephfs/orgs/home/$USER/logs
```

```
mkdir -p "$OUT"
```

```
apptainer exec \
```

```
--bind /mnt/cephfs:/mnt/cephfs \
```

```
--bind "$DATA:/data:ro" \
```

```
--bind "$OUT:/out:rw" \
```

```
"$SIF" \
```

```
/opt/miniforge3/envs/r-bio/bin/Rscript mi_analisis_gwas.R
```

Para trabajos que requieren aceleración por GPU, se añade la bandera `--nv`, que inyecta automáticamente las bibliotecas de NVIDIA del sistema anfitrión dentro del contenedor además, de añadir `#SBATCH --gres=gpu:1`, cambiar la partición y la QoS a `gpu`.

```
apptainer exec --nv \
```

```
--bind /mnt/cephfs:/mnt/cephfs \
```

```
"$SIF" \
```

```
/opt/miniforge3/envs/jax-gpu/bin/python gpu_script.py
```

La documentación completa de los entornos de Conda y las plantillas de scripts de Slurm se mantiene en un sitio web estático accesible en <https://docs.mcps-epcm.org>, desplegado como contenedor Nginx en el nodo maestro y actualizado por el administrador del clúster.

5.4 Configuración de red y servicios base

El nodo maestro actúa como puerta de enlace NAT para los nodos de cómputo. El reenvío de paquetes se habilitó en el arranque en el kernel (`net.ipv4.ip_forward = 1` en `/etc/sysctl.conf`). La traducción de direcciones se implementó con una regla MASQUERADE en la tabla NAT de iptables, que reescribe la dirección de origen de los paquetes provenientes de la red interna (10.42.42.0/24) con la dirección de la interfaz de salida (`enp68s0`) antes de enviarlos al exterior. Netavark, el backend de red de Podman, genera automáticamente reglas adicionales de MASQUERADE y DNAT para la red de contenedores mcps-net (10.89.0.0/24), de modo que los contenedores se comuniquen entre sí y con el exterior sin configuración manual adicional.

La red interna del clúster operó inicialmente a 1 Gbps con cable de cobre y un switch no administrado. Durante la ejecución de los primeros pipelines genómicos a escala — en particular la conversión de VCF a Zarr del Freeze 150k y los primeros pasos de la construcción del panel de referencia MCPS10k; se observó que el throughput de CephFS se degradaba al ejecutar múltiples trabajos simultáneamente. Tres nodos leyendo y escribiendo archivos de cientos de gigabytes saturaban el enlace de 1 Gbps.

Se migró la red interna a fibra óptica SFP+ de 10 Gbps. La migración requirió configuración adicional del stack de red en todos los nodos para aprovechar el ancho de banda disponible. Los ring buffers de las NIC Intel X540 se expandieron de 512 a 8,192 entradas para absorber ráfagas de tráfico sin descartar paquetes. Los buffers TCP del kernel se ampliaron a 16 MB (`net.core.rmem_max`, `net.core.wmem_max`, `net.ipv4.tcp_rmem`, `net.ipv4.tcp_wmem`) para sostener transferencias de 10 Gbps. El backlog de paquetes del kernel se incrementó a 30,000 entradas (`net.core.netdev_max_backlog`) con un presupuesto de 600 paquetes por ciclo de poll (`net.core.netdev_budget`) para evitar descartes cuando los cores están bajo carga intensa de cómputo. Todos estos parámetros se escribieron en `/etc/sysctl.d/99-10gbe-tuning.conf` y en un archivo `.link` de `systemd-networkd` para los ring buffers, garantizando su persistencia tras reinicios.

La configuración del switch administrado TP-Link de 10 GbE requirió dos ajustes. Primero, se habilitaron los jumbo frames en los 8 puertos con un tamaño máximo de 9,216 bytes — valor que cubre holgadamente el MTU de 9,000 configurado en las interfaces de los nodos. Sin este ajuste en el switch, los paquetes jumbo serían fragmentados o descartados en tránsito, anulando la ventaja de reducir la sobrecarga de encabezados TCP/IP en transferencias

grandes. Segundo, se activó el control de flujo (flow control) en todos los puertos. El control de flujo permite que un puerto del switch señale a la NIC emisora que reduzca temporalmente la velocidad de transmisión cuando sus buffers internos están cerca de saturarse. Sin este mecanismo, las ráfagas de tráfico generadas por múltiples nodos escribiendo simultáneamente sobre CephFS provocan descartes de paquetes en el switch que se manifiestan como retransmisiones TCP y degradación del throughput.

Un componente crítico de la configuración de red fue irqbalance, el demonio que distribuye las interrupciones de hardware de la NIC entre los cores disponibles del procesador. Sin irqbalance, las 24 colas de interrupción de la NIC de 10 GbE caen en un solo core por defecto. Si ese core está saturado con un proceso de análisis genómico, los paquetes de red entrantes no se atienden a tiempo y se descartan. La importancia de este componente se descubrió durante un incidente operativo descrito en la sección 8.3.

UFW (*Uncomplicated Firewall*) gestiona las reglas de filtrado del nodo maestro con una política por defecto de denegar tráfico entrante y permitir tráfico saliente y enrutado. Las reglas permiten el tráfico entrante desde la red interna del clúster (10.42.42.0/24) y desde la red de contenedores (10.89.0.0/16) sin restricción. Desde el exterior se permiten exclusivamente los puertos de SSH (22), HTTP/HTTPS (80, 443), correo (25, 587, 993, 995) y la VPN WireGuard. Los puertos de Slurm (6868, 6871, 7320), MariaDB (3306) y Spark (7077, 8080, 8081) solo aceptan conexiones desde la red interna. El tráfico proveniente de la VPN (10.8.0.0/24 e interfaz wg0) se admite sin restricciones, ya que los usuarios autenticados por VPN deben acceder a los servicios internos del clúster.

El clúster opera con dos instancias de Caddy, una en la instancia EC2 que gestiona los certificados TLS de Let's Encrypt para los dominios públicos, y otra en el nodo maestro que distribuye el tráfico a los contenedores internos. El Caddy del nodo maestro se ejecuta como contenedor Podman en la red mcps-net (IP 10.89.0.14), con los puertos 80, 443 y 443/UDP publicados al host. Su archivo de configuración (Caddyfile) reside en CephFS (/mnt/cephfs/hot/caddy/Caddyfile) y define un bloque de proxy inverso por cada servicio:

Subdominio	Destino	Servicio
jupyter.mcps-epcm.org	10.89.0.29:8000	JupyterHub
bio-r.mcps-epcm.org	10.89.0.20:8787	RStudio Server
files.mcps-epcm.org	10.89.0.205:80	Nextcloud
soporte.mcps-epcm.org	10.89.0.17:3000 /10.89.0.18:6042	Zammad (Rails + WebSocket)
postgres.mcps-epcm.org	10.89.0.12:80	pgAdmin
ceph.mcps-epcm.org	10.42.42.2:9443	Dashboard de Ceph
grafana.mcps-epcm.org	10.89.0.211:3000	Grafana
docs.mcps-epcm.org	10.89.0.33:80	Documentación
inventario.mcps-epcm.org	10.89.0.40:80	Snipe-IT (inventario del MCPS)
office.mcps-epcm.org	10.89.0.206:80	OnlyOffice
pgscat.mcps-epcm.org	10.89.0.34:8080	PGS Catalog Dashboard

Los servicios internos (JupyterHub, RStudio, pgAdmin, Ceph, inventario, PGS Catalog) se configuran con `tls internal`, lo que genera certificados autofirmados válidos dentro de la red VPN. Los servicios públicos (Nextcloud, GTEx, documentación, OnlyOffice, Grafana) se sirven por HTTP a través del proxy de EC2, que se encarga de la terminación TLS con certificados de Let's Encrypt. Para agregar un nuevo servicio, basta con añadir un bloque al Caddyfile, crear el subdominio correspondiente en Route 53, y recargar la configuración con `podman exec caddy caddy reload`.

Se configuraron tres jails de protección contra ataques de fuerza bruta: `sshd` para intentos fallidos de SSH, `sshd-country` para bloqueo por país de origen, y `recidive` para reincidentes que acumulan baneos en múltiples jails. La política de la jail principal (`sshd`) permite un máximo de tres intentos fallidos antes de bloquear la IP. Las redes confiables excluidas del bloqueo incluyen la interfaz de loopback (127.0.0.0/8), la red interna del clúster (10.42.42.0/24) y la red de la VPN (10.8.0.0/24).

El archivo `/etc/hosts` del nodo maestro asocia las direcciones IP con los nombres de los cinco nodos del clúster y con el dominio de correo electrónico (`mail.mcps-epcm.org` apuntando a la IP

del servidor EC2 en la VPN). Este archivo se copia a todos los nodos de cómputo para garantizar la resolución de nombres sin depender de un servidor DNS.

La gestión remota de los nodos se realiza con ClusterShell (clush), configurado con un grupo all que incluye los cinco nodos del clúster. Esto hace posible ejecutar comandos en paralelo en todos los nodos desde el nodo maestro, lo que simplifica operaciones recurrentes como la distribución de archivos de configuración, la verificación de estado y las actualizaciones de paquetes.

Se configuró un mecanismo de notificaciones por correo para que los usuarios reciban avisos cuando sus trabajos cambian de estado (inicio, finalización, error). Slurm no envía correos directamente; invoca un programa externo definido en la directiva MailProg de slurm.conf. Se implementó un script (/usr/local/bin/slurm-mailprog) que recibe el destinatario y el cuerpo de la notificación de Slurm, construye un mensaje con los datos del trabajo (identificador, host, fecha) y lo envía mediante msmtplib, un cliente SMTP ligero que no requiere un servidor de correo local. msmtplib está configurado para conectarse al servidor de correo institucional (mail.mcps-epcm.org) en el puerto 587 con autenticación STARTTLS, utilizando la cuenta de soporte del clúster como remitente. De esta forma, cuando un usuario incluye la directiva #SBATCH --mail-user=usuario@mcps-epcm.org en su script de trabajo, recibe notificaciones en su buzón institucional sin necesidad de configurar nada adicional.

5.5 Implementación del almacenamiento distribuido Ceph

5.5.1 Despliegue inicial y configuración general

El clúster Ceph se desplegó mediante cephadm, la herramienta oficial de orquestación para Ceph Squid. El bootstrap inicial se ejecutó en el nodo maestro (mcps-hpc-01), que se convirtió en el primer monitor (MON). Los cuatro nodos restantes se incorporaron progresivamente al clúster, cada uno aportando sus discos como OSD y su demonio de monitor. La creación del sistema de archivos siguió una secuencia definida: primero las reglas CRUSH para separar los dispositivos por clase, luego los pools de almacenamiento, y finalmente el filesystem CephFS. Se definieron dos reglas CRUSH para dirigir los datos al tipo correcto de dispositivo:

```
ceph osd crush rule create-replicated ssd_rule default host ssd
```

```
ceph osd crush rule create-replicated hdd_rule default host hdd
```

Se crearon los pools principales con factor de replicación 2 y tamaño mínimo 1:

```
ceph osd pool create ssd_metadata 64 64 replicated
ceph osd pool create hdd_capacity 128 128 replicated
ceph osd pool set ssd_metadata crush_rule ssd_rule
ceph osd pool set hdd_capacity crush_rule hdd_rule
ceph osd pool set ssd_metadata size 2
ceph osd pool set ssd_metadata min_size 1
ceph osd pool set hdd_capacity size 2
ceph osd pool set hdd_capacity min_size 1
```

Se eligió un factor de replicación de 2 en lugar del valor recomendado de 3 por una restricción práctica: con solo 5 nodos y 2 HDD por nodo (10 HDD en total), un factor de 3 consumiría el 75% del espacio bruto en réplicas, dejando apenas 52 TB útiles de los 208 TB de HDD disponibles. Con replicación 2, el clúster ofrece aproximadamente 104 TB útiles, suficientes para los 46 TB de datos actuales con margen de crecimiento. El parámetro `min_size=1` hace que el clúster continúe operando (en modo degradado) aunque una de las dos réplicas no esté disponible, lo que evita interrupciones durante reinicios de nodos o fallos temporales de disco.

El filesystem se creó asignando `ssd_metadata` como pool de metadatos y `hdd_capacity` como pool de datos:

```
ceph fs new cephfs_hpc ssd_metadata hdd_capacity
ceph fs set cephfs_hpc allow_standby_replay true
ceph fs set cephfs_hpc max_file_size 1099511627776
```

El parámetro `max_file_size` se estableció en 1 TB para permitir archivos genómicos de gran tamaño sin restricción. La directiva `allow_standby_replay` hace que el MDS en espera mantenga una copia activa del journal de metadatos, lo que acelera la

conmutación ante falla de un MDS activo. Posteriormente se añadió un segundo pool SSD (nvme_hot) para datos de acceso frecuente y se incrementaron los placement groups (PG) del pool hdd_capacity a 512, como se describe más adelante.

La configuración inicial del clúster asignaba los NVMe como OSD independientes junto a los HDD, sin separación de roles. Bajo esta configuración, los HDD almacenaban tanto los datos como los índices internos de BlueStore (RocksDB), lo que provocaba latencias elevadas en las operaciones de metadatos y escritura. La migración a una arquitectura híbrida, donde los NVMe almacenan los índices de RocksDB y los HDD se dedican exclusivamente a los datos, se realizó nodo por nodo para evitar interrupciones del servicio.

El procedimiento por nodo consistió en: sacar el OSD NVMe del clúster (`ceph osd out`), esperar a que los datos se redistribuyeran (`active+clean`), eliminar el OSD (`ceph orch osd rm --zap`), reparticionar el NVMe en volúmenes lógicos LVM, una partición de 1 TB para el DB de cada HDD (`db_sda`, `db_sdb`) y el espacio restante como OSD rápido (`osd_nvme`), inyectar las particiones DB en los OSD de los HDD con `ceph-volume lvm new-db` y migrar los metadatos existentes con `ceph-volume lvm migrate`, y finalmente levantar los OSD y crear un nuevo OSD sobre la partición NVMe restante. Cada paso requería esperar a que el clúster alcanzara el estado `active+clean` antes de continuar.

Un problema específico de Ubuntu 24.04 requirió atención, LVM bloqueaba los dispositivos con `use_devicesfile`, lo que impedía que `ceph-volume` accediera a los volúmenes. La solución fue desactivar esta funcionalidad en `/etc/lvm/lvm.conf` en todos los nodos antes de comenzar la migración.

El problema más crítico encontrado durante la operación del clúster fue la determinación de la capacidad de IOPS de los HDD por parte de mClock, el planificador de QoS de Ceph. Durante la inicialización de cada OSD, Ceph ejecuta un benchmark local (`osd bench`) que mide las escrituras aleatorias a 4 KiB directamente sobre el disco, sin tráfico de red. Si el resultado es inferior al umbral de plausibilidad de 500 IOPS (`osd_mclock_iops_capacity_threshold_hdd`), Ceph lo acepta como la capacidad real del disco. Si lo supera, lo descarta y aplica el valor por defecto de 315 IOPS (Ceph

Foundation, s.f.). Con esa cifra, mClock calcula cuántas operaciones puede aceptar cada OSD antes de encolar las siguientes.

El problema es que ese benchmark no refleja las condiciones reales de operación: cuando cada escritura se replica a un segundo nodo por un enlace de 1 Gbps compartido entre cinco nodos, la capacidad efectiva cae muy por debajo de la medición local. Los discos mecánicos de 20 TB, con alta densidad de platinas y cabezales de largo recorrido, sostenían en la práctica alrededor de 100 IOPS con la red de cobre. La discrepancia provocaba que mClock saturara los discos, generando latencias superiores a 400 ms y marcando datos como inconsistentes durante las operaciones de scrubbing. La corrección inicial consistió en fijar manualmente el límite:

```
ceph config set osd/class:hdd osd_mclock_max_capacity_iops_hdd 100
```

El ajuste estabilizó la latencia de escritura y eliminó los falsos reportes de inconsistencia. Tras la migración a fibra óptica SFP+ a 10 Gbps, el cuello de botella de red desapareció y los mismos discos sostuvieron IOPS más altos sin degradación. El límite se reajustó a 215 IOPS como punto de equilibrio entre rendimiento y estabilidad:

```
ceph config set osd/class:hdd osd_mclock_max_capacity_iops_hdd 215
```

La configuración inicial asignó 128 PG al pool hdd_capacity. A medida que el volumen de datos alcanzó los 46 TB y 17.4 millones de objetos, la proporción de datos por PG se volvió excesiva, provocando un desbalance en la distribución física entre los OSD. En una primera fase, se incrementaron los PG a 512 para mejorar la granularidad de la carga. Para refinar esta distribución y lograr una uniformidad óptima, se implementó el algoritmo de balanceo más estricto del gestor, mgr/balancer/mode upmap. Al generar mapeos explícitos para PGs individuales sobre el algoritmo CRUSH, el modo upmap permitió una optimización de la topología de almacenamiento, reduciendo la varianza de utilización entre discos de >2.0 a 1.16.

Se aplicaron tres ajustes complementarios en todos los nodos. Primero, se configuró el kernel para utilizar la RAM como buffer de escritura (vm.dirty_ratio=40, vm.dirty_background_ratio=10), de modo que hasta el 40% de la RAM absorba ráfagas de escritura antes de forzar la escritura a disco. Segundo, se desactivó la caché de

escritura de los HDD (`smartctl -s wcache,off`) para garantizar la consistencia de los datos. Sin esta medida, una pérdida de energía podría corromper datos que el disco reportó como escritos pero que solo residían en su caché volátil. Tercero, se incrementó el umbral de BlueStore para aceptar escrituras concurrentes (`bluestore_throttle_bytes` y `bluestore_throttle_deferred_bytes` a 256 MB), reduciendo los cuellos de botella durante ráfagas de escritura de los pipelines bioinformáticos.

Las lecturas de archivos genómicos de cientos de gigabytes son predominantemente secuenciales. El read-ahead por defecto del kernel (128 KB) obligaba a los HDD a realizar seeks innecesarios entre lecturas pequeñas. Se incrementó a 8 MB (`blockdev --setra 16384`) para que el kernel lea bloques grandes por adelantado, aprovechando la velocidad secuencial de los platos magnéticos. La desactivación de la write cache de los HDD, descrita anteriormente, se hizo persistente con una regla udev (`/etc/udev/rules.d/61-disable-wcache.rules`) que ejecuta `smartctl -s wcache,off` en cada arranque, ya que la caché se rehabilita sola tras un ciclo de energía.

Los dispositivos NVMe del clúster tenían habilitada la transición autónoma de estados de energía (APST), que permite al firmware entrar en modos de bajo consumo si no recibe operaciones durante unos instantes. Al despertar, la primera operación sufre una latencia de hasta 100 ms — un valor que para Ceph es inaceptable, dado que los NVMe normalmente responden en 50 microsegundos. BlueStore interpreta esos picos como un fallo del dispositivo y puede abortar transacciones, lo que en el caso de un NVMe que aloja el WAL/DB de dos OSDs mecánicos provoca la caída simultánea de ambos. Se deshabilitó el APST en todos los nodos con el parámetro `nvme_core.default_ps_max_latency_us=0` en la línea de comandos del kernel (GRUB).

Al momento de esta documentación, el clúster Ceph opera con 20 OSD distribuidos en 5 nodos

Nodo	OSD HDD	OSD SSD/NVMe	Almacenamiento HDD	Almacenamiento SSD
mcps-hpc-01	osd.13, osd.14	osd.15	38.2 TB	1.8 TB
MCPS-HPC02	osd.1, osd.2	osd.0, osd.16	40.0 TB	5.4 TB
MCPS-HPC03	osd.4, osd.5	osd.3, osd.12	40.0 TB	5.5 TB
MCPS-HPC04	osd.7, osd.8	osd.6, osd.18	40.0 TB	5.4 TB
MCPS-HPC05	osd.10, osd.11	osd.9, osd.17, osd.19	40.0 TB	6.3 TB

La capacidad total bruta es de 232 TB (208 TB en HDD, 24 TB en SSD/NVMe), con 103 TB en uso y 130 TB disponibles. Los cuatro pools almacenan

Pool	Clase	Datos almacenados	Objetos	PGs
.mgr	HDD	92 MB	24	1
ssd_metadata	SSD	1.7 GB	2.79M	64
hdd_capacity	HDD	46 TB	17.46M	512
nvme_hot	SSD	0 B	0	512

5.5.2 Integración con el clúster y montaje para usuarios/servicios

CephFS se monta en todos los nodos con el cliente de kernel de Ceph (no FUSE), configurado en /etc/fstab con la lista de los cinco monitores:

```
10.42.42.1:6789,...,10.42.42.5:6789:/mnt/cephfs          ceph
name=admin,secret=<key>,noatime,_netdev 0 0
```

La opción `_netdev` indica al sistema que el montaje depende de la red, evitando que el arranque se bloquee si CephFS no está disponible inmediatamente. La opción `noatime` desactiva la actualización de la marca de tiempo de acceso en cada lectura, lo que reduce la carga de operaciones de metadatos, en un entorno donde miles de archivos genómicos se leen concurrentemente, actualizar `atime` en cada lectura generaría millones de escrituras de metadatos innecesarias.

Los contenedores Podman del nodo maestro acceden a CephFS con bind mounts en sus archivos de servicio systemd. Cada unidad de servicio incluye una directiva `ExecStartPre` que espera en un bucle hasta que el directorio de CephFS esté montado antes de arrancar el contenedor. JupyterHub y RStudio Server montan los directorios `home` de los usuarios (`/mnt/cephfs/orgs/home/`) y los datos genómicos de referencia (`/mnt/cephfs/hot_nvme/`) como volúmenes de solo lectura, mientras que Nextcloud monta el directorio `home` con permisos de escritura para permitir la sincronización de archivos.

Se configuraron parámetros de lectura anticipada (`readahead`) en el cliente CephFS para mejorar el rendimiento de lecturas secuenciales de archivos genómicos grandes:

```
client_readahead_max_bytes = 134217728 # 128 MB
```

```
client_readahead_min = 524288 # 512 KB
```

Estos valores hacen que el cliente lea bloques de hasta 128 MB por adelantado cuando detecta un patrón de lectura secuencial, lo que es frecuente en el procesamiento de archivos PLINK, BAM y BGEN.

5.5.3 Consideraciones de operación: cuotas, permisos y respaldos

Las cuotas de almacenamiento se implementan a nivel de directorio con el atributo extendido `ceph.quota.max_bytes` de CephFS. Cada directorio `home` de usuario tiene asignado un límite de 500 GB. Cuando un usuario alcanza su cuota, las operaciones de escritura en su directorio fallan con un error de espacio insuficiente, sin afectar a otros usuarios ni al resto del sistema de archivos. El directorio `/mnt/cephfs/scratch/` no tiene cuota asignada y está disponible para archivos temporales de todos los usuarios, con el `sticky` bit activado para evitar que un usuario elimine archivos de otro.

Los permisos se gestionan combinando permisos POSIX estándar y listas de control de acceso (ACL). Los datos genómicos de referencia en `/mnt/cephfs/hot_nvme/` tienen permisos de lectura para todos los usuarios y escritura restringida al administrador y a los usuarios autorizados mediante ACL. Los directorios home en `/mnt/cephfs/orgs/home/` están configurados con el bit SetGID y propiedad de `www-data:nextcloudfs` para la integración con Nextcloud, y cada usuario accede a su directorio exclusivamente a través de ACL nominales.

La configuración de los servidores de metadatos (MDS) asigna 16 GB de caché de memoria (`mds_cache_memory_limit = 17179869184`) para mantener en RAM la mayor cantidad posible de metadatos del sistema de archivos. Esta caché reduce la necesidad de consultar los metadatos en disco durante operaciones frecuentes como listar directorios o verificar permisos, lo que beneficia directamente a los usuarios que trabajan con directorios que contienen miles de archivos genómicos.

En cuanto a la integridad de los datos, Ceph ejecuta procesos periódicos de verificación (scrubbing) que comparan los datos almacenados en cada réplica para detectar inconsistencias. El scrub ligero se ejecuta cada 5 a 7 días y el deep scrub cada 15 días, con un máximo de 4 operaciones de scrub simultáneas y una prioridad baja (5) para minimizar el impacto en las operaciones de los usuarios. Las ventanas de scrub están configuradas para ejecutarse todos los días de la semana en horarios de baja demanda.

No se implementó un sistema de respaldos externo automatizado para CephFS. La protección contra pérdida de datos se basa en la replicación inherente de Ceph (factor 2), que tolera la falla de un disco o un nodo sin pérdida de información. Los datos genómicos originales del MCPS existen en copias externas mantenidas en Globus por los colaboradores de Oxford. Esta limitación se identifica como una línea de trabajo futuro en la sección 9.2.

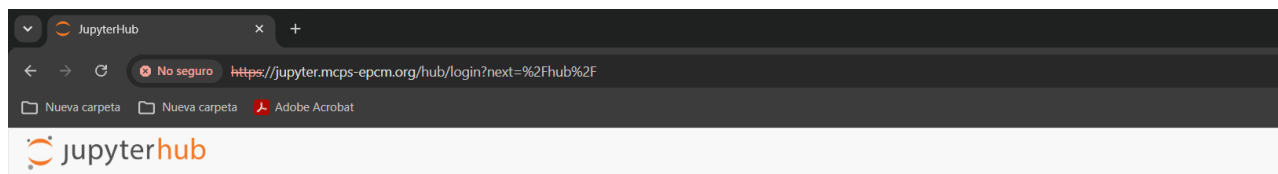
6. Integración de servicios y soporte a usuarios

6.1 Servicios de acceso a entornos de análisis

El clúster ofrece dos entornos interactivos accesibles desde el navegador, JupyterHub para Python y RStudio Server para R. Ambos corren como contenedores Podman en el nodo maestro, gestionados por unidades systemd que los reinician automáticamente ante cualquier falla.

JupyterHub usa la imagen `jupyter-biotools1.5`, la misma que se convierte a formato SIF para los trabajos de Slurm. Corre en la red `mcps-net` con IP `10.89.0.29` y escucha en el puerto `8000`, accesible desde `jupyter.mcps-epcm.org` a través de la VPN. La autenticación funciona con PAM. El contenedor monta `/etc/passwd`, `/etc/shadow` y `/etc/pam.d` del host como volúmenes de solo lectura, así que el usuario entra con las mismas credenciales que usa para SSH.

Lo que hace a JupyterHub útil en el clúster no es solo el notebook. El contenedor monta los directorios home de CephFS (`/mnt/cephfs/orgs/home/`) y los datos genómicos y epidemiológicos (`/mnt/cephfs/hot_nvme/` en solo lectura). También monta la configuración de Slurm y el socket de Munge, lo que significa que un investigador puede abrir un terminal dentro de JupyterHub y ejecutar `sbatch` directamente. La idea es que JupyterHub sirva como entorno de desarrollo y depuración. El usuario puede probar su código con un subconjunto pequeño de datos en el notebook, y cuando funciona, lo empaqueta en un script `sbatch` para ejecutarlo a escala en los nodos de cómputo.



Sign in

Username:

Password:

Sign in

Figura 5. Login de ingreso de jupyterhub

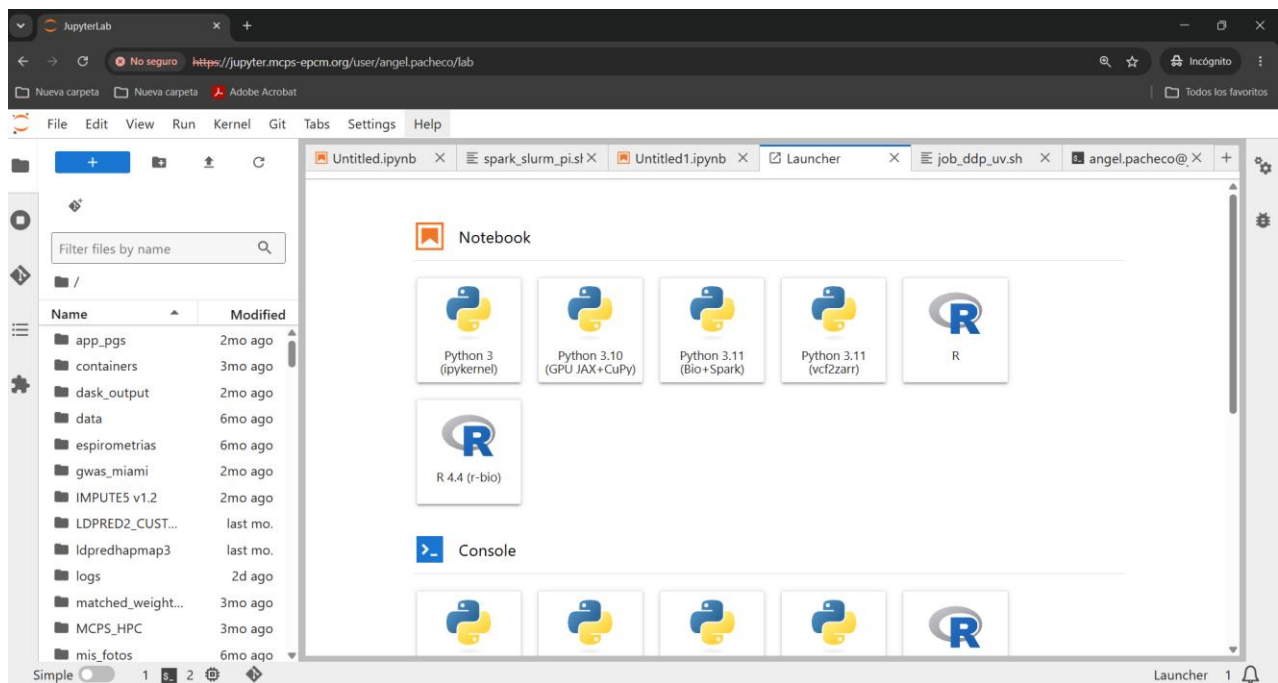


Figura 6. Panel principal de jupyterhub

RStudio Server sigue el mismo patrón. Corre con la imagen `rstudio-biotools:latest` en la IP 10.89.0.20, accesible desde `bio-r.mcps-epcm.org`. Monta los mismos volúmenes: homes de CephFS, datos genómicos, configuración de Slurm y socket de Munge. La diferencia principal es el público, los investigadores que trabajan con R y Bioconductor prefieren RStudio sobre

JupyterHub. Tener ambos servicios disponibles evita forzar a los usuarios a cambiar de herramienta.

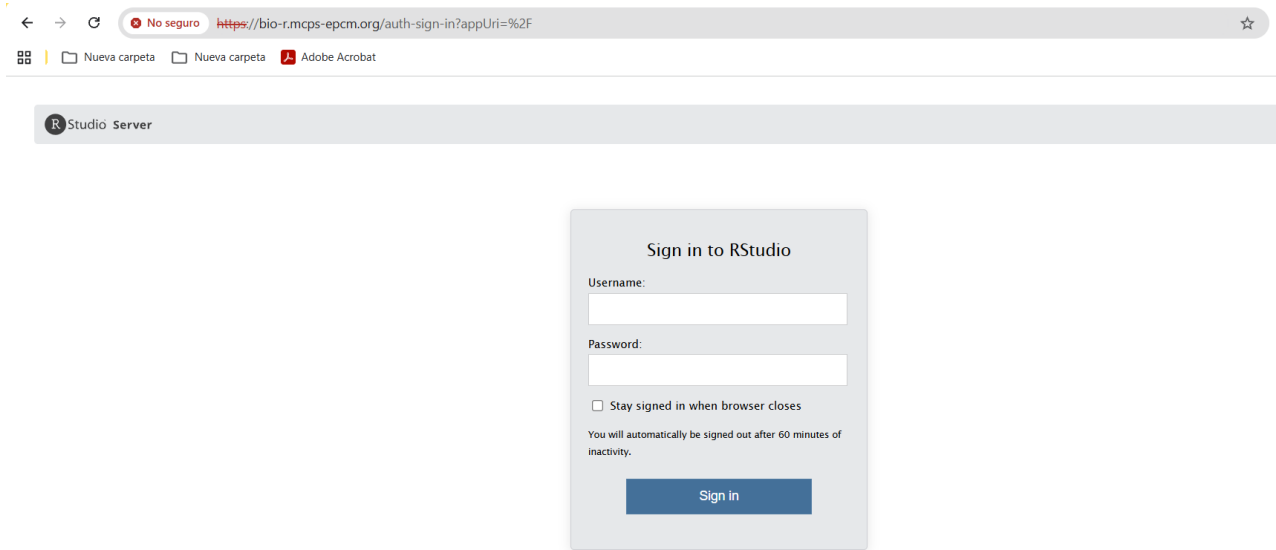


Figura 7. Login de ingreso rstudio

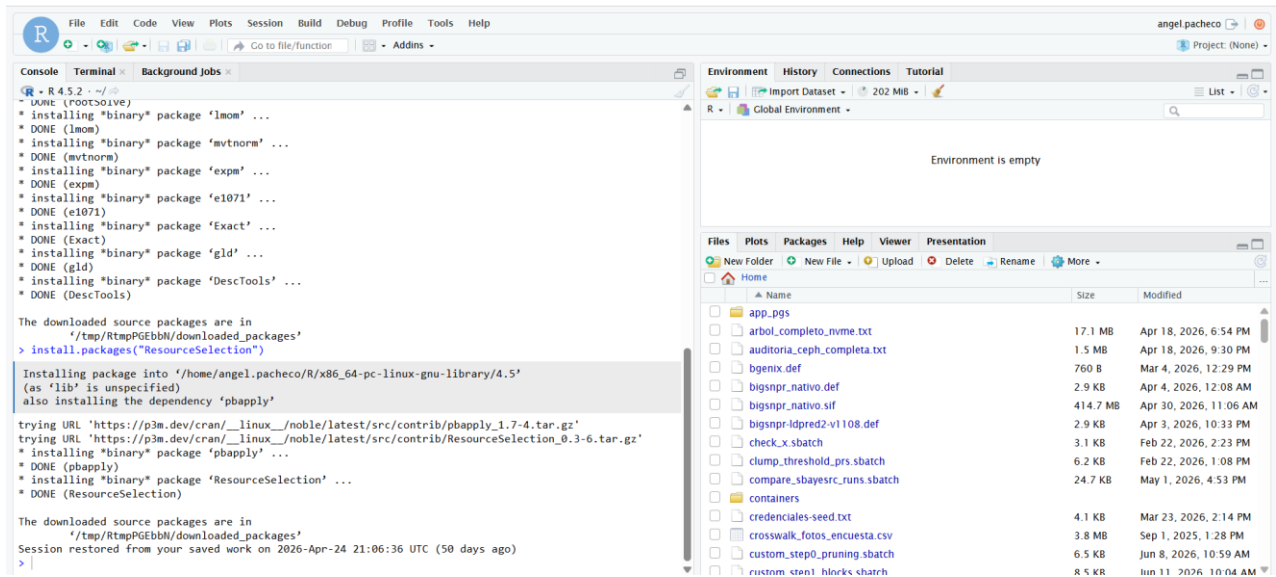


Figura 8. Panel principal de rstudio

6.2 Servicios de almacenamiento y colaboración

Nextcloud funciona como la plataforma de almacenamiento colaborativo del clúster. Corre en la IP 10.89.0.205 y a diferencia de JupyterHub y RStudio, es accesible públicamente desde `files.mcps-epcm.org` sin requerir VPN. El DNS apunta a la IP elástica de EC2, que reenvía el tráfico al nodo maestro.

La integración con CephFS es lo que distingue a esta instalación de un Nextcloud convencional. El contenedor monta `/mnt/cephfs/orgs/home/` como volumen, lo que significa que los archivos que un investigador ve en Nextcloud son los mismos que ve por SSH o en JupyterHub. No hay duplicación. El truco que hace posible esta integración es la configuración de permisos descrita en la sección 4.4: los directorios `home` pertenecen a `www-data:nextcloudfs` con el bit `SetGID` activado, y cada usuario tiene acceso a su directorio a través de ACLs nominales. Nextcloud (que corre como `www-data`) puede leer y escribir en todos los `homes`, pero cada usuario solo ve el suyo.

Nextcloud depende de dos servicios adicionales. PostgreSQL (IP 10.89.0.11) sirve como base de datos para la metadata de archivos, usuarios y actividad. Redis (IP 10.89.0.13) actúa como caché en memoria para acelerar las consultas. Ambos corren como contenedores Podman con sus propios archivos de servicio `systemd` y almacenan sus datos en CephFS (`/mnt/cephfs/hot/postgres/data/` y en memoria, respectivamente).

OnlyOffice complementa a Nextcloud como servidor de edición de documentos. Corre en la IP 10.89.0.206, accesible desde `office.mcps-epcm.org`, y se conecta a Nextcloud a través de un token JWT compartido. Los investigadores pueden editar documentos de texto, hojas de cálculo y presentaciones directamente en el navegador sin instalar software local.

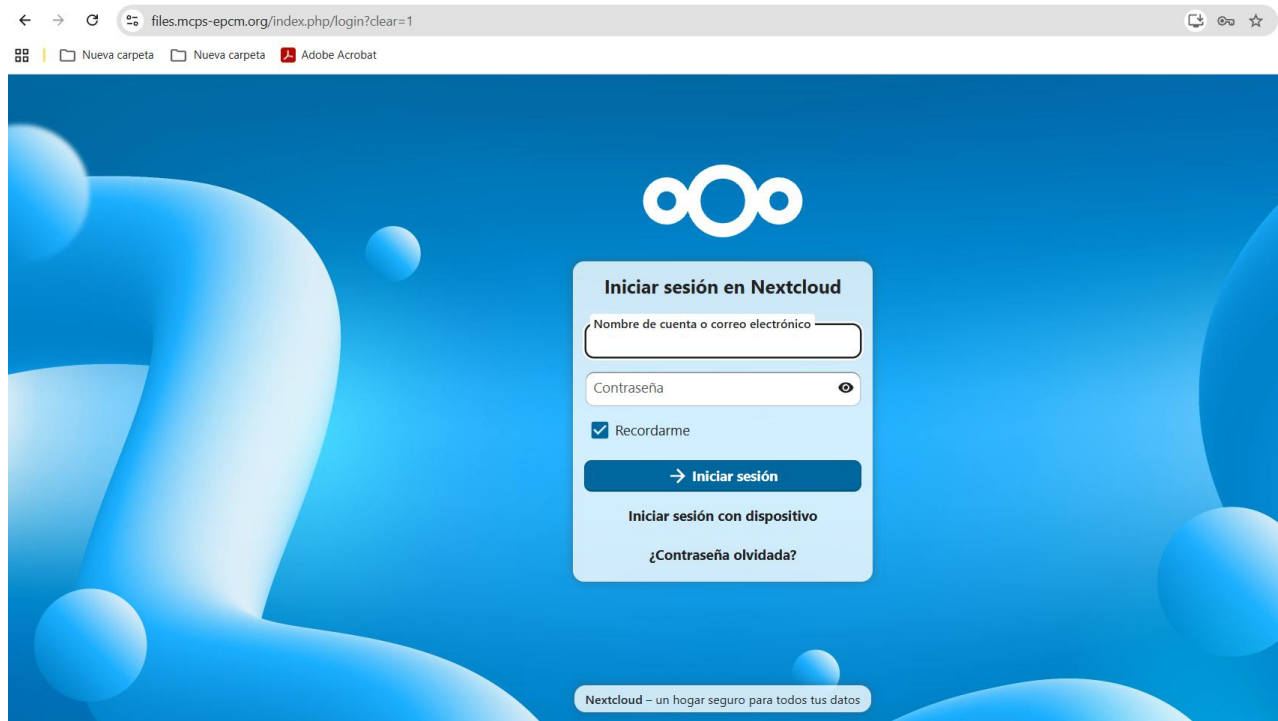


Figura 9. Login de ingreso a nextcloud

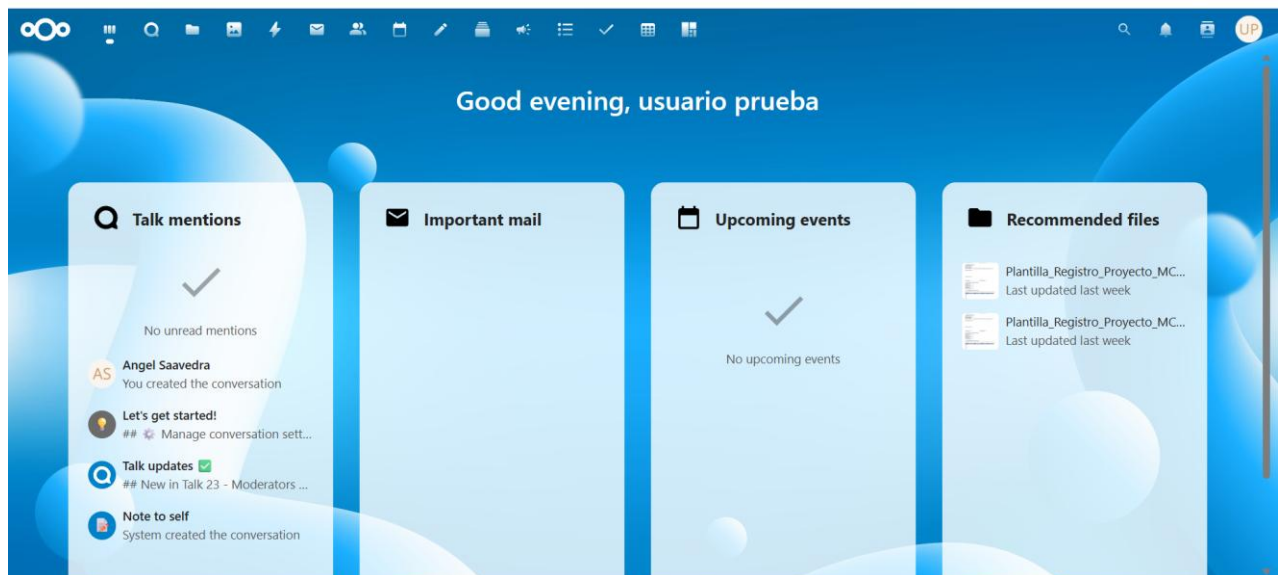


Figura 10. Panel principal de nextcloud

6.3 Sistema de soporte y atención a incidencias

En el capítulo 3 se documentó que la comunicación entre usuarios y administrador se realizaba por WhatsApp y conversaciones cara a cara, con solicitudes que se olvidaban o no se respondían a tiempo. Zammad resolvió ese problema. Es un sistema de mesa de ayuda accesible desde soporte.mcps-epcm.org (solo por VPN) que convierte cada correo enviado a soporte@mcps-epcm.org en un ticket rastreable.

El flujo es directo. Un usuario envía un correo reportando un problema o solicitando algo. Zammad lo recibe, crea un ticket, y lo asigna al administrador. Cada respuesta queda registrada en el historial del ticket. El usuario recibe las respuestas en su buzón de correo. No hay solicitudes que se pierdan porque todo queda documentado con fecha, hora y responsable.

La implementación de Zammad involucra tres contenedores que trabajan juntos. El servidor Rails (IP 10.89.0.17) maneja la lógica de la aplicación y la interfaz web. El contenedor WebSocket (IP 10.89.0.18) gestiona las notificaciones en tiempo real, cuando un ticket se actualiza, el cambio aparece instantáneamente en la pantalla del administrador sin recargar la página. El planificador (IP 10.89.0.19) ejecuta tareas programadas como la revisión periódica de correos entrantes y el envío de recordatorios. Memcached (IP 10.89.0.16) proporciona caché en memoria para la sesión de usuario. Los tres contenedores de Zammad comparten la misma base de datos PostgreSQL que usa Nextcloud, pero en una base separada (zammad), y usan Redis para la cola de tareas.

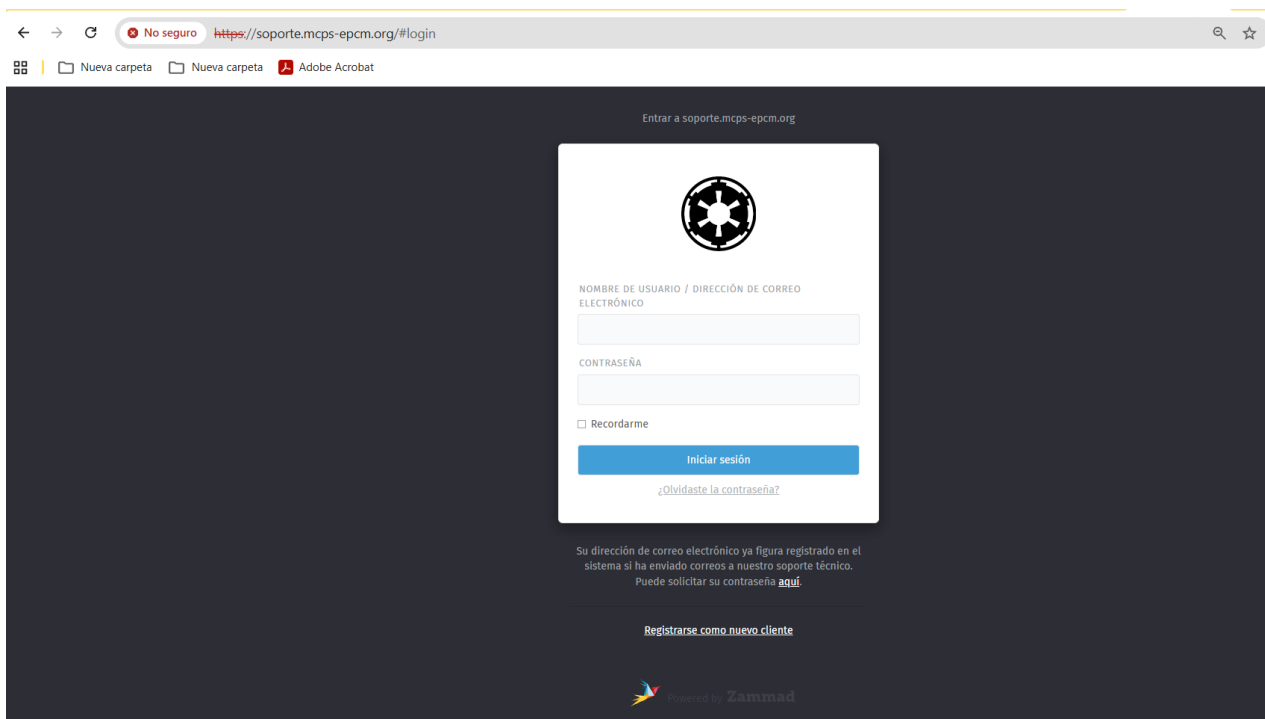


Figura 11. Login de ingreso a zammad

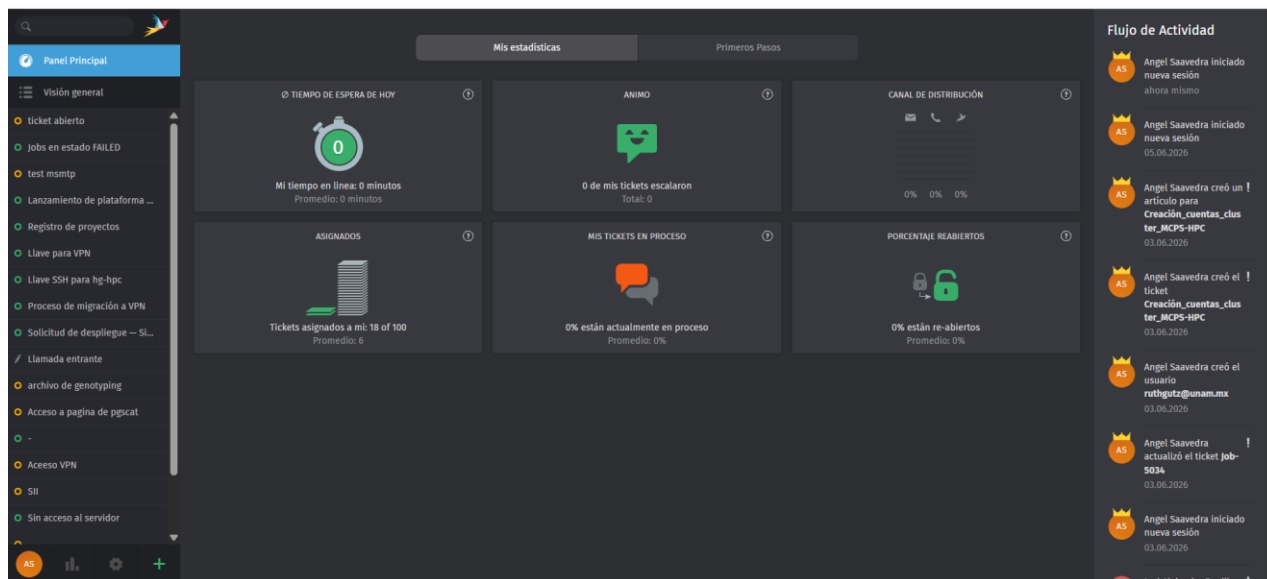


Figura 12. Panel principal de zammad para administrador

6.4 Servicios de comunicación y notificación

El correo electrónico institucional opera bajo el dominio mcps-epcm.org, gestionado en Route 53 de AWS. La configuración DNS incluye registros MX que apuntan a mail.mcps-epcm.org, registros SPF para autorizar Amazon SES como remitente válido, registros DKIM con firma RSA para verificación criptográfica de los correos salientes, registros DMARC con política de rechazo (p=reject) para proteger contra suplantación de identidad, y registros MTA-STS para forzar el uso de TLS en las conexiones SMTP entrantes.

El servidor de correo opera directamente en la instancia EC2 de AWS, no como contenedor dentro del nodo maestro del clúster. Se eligió esta ubicación porque el servicio de correo necesita una IP pública fija y puertos abiertos (25, 465, 587, 993, 995) que la red del proveedor externo no puede garantizar, los cortes de la conexión a internet y de electricidad afectarían la recepción de correo. Docker Mailserver corre con Postfix para SMTP y Dovecot para IMAP/POP3, con relay saliente a través de Amazon SES. Al momento de la documentación, el servidor aloja buzones para 33 usuarios entre investigadores mexicanos y personal administrativo del proyecto.

Los certificados TLS del servidor de correo los emite Let's Encrypt a través de Caddy en la misma instancia EC2. Un timer de systemd (sync-mail-cert.timer) ejecuta diariamente un script que compara el certificado vigente de Caddy con el que usa Docker Mailserver, valida que no esté próximo a vencer, verifica que certificado y llave correspondan, y solo si hay cambios copia los archivos y reinicia el servicio. La automatización se implementó después de un incidente en el que el certificado expiró sin que nadie lo detectara, lo que provocó que Gmail y otros clientes rechazaran las conexiones POP3S e IMAPS durante medio día.

Los servicios internos del clúster que necesitan enviar correo se conectan al servidor de EC2 a través de la VPN. Zammad apunta a mail.mcps-epcm.org en el puerto 587 con STARTTLS para enviar respuestas a tickets. Snipe-IT usa la misma ruta para alertas de inventario. Las notificaciones de Slurm se envían a través de msmtplib, un cliente SMTP ligero configurado en el nodo maestro que también se conecta a mail.mcps-epcm.org por el puerto 587. Cuando un usuario incluye #SBATCH --mail-user=usuario@mcps-epcm.org en su script, recibe un correo al iniciar, terminar o fallar el trabajo. El script personalizado /usr/local/bin/slurm-mailprog

construye el mensaje con el identificador del trabajo, el nodo de ejecución y la fecha antes de pasarlo a msmtpt.

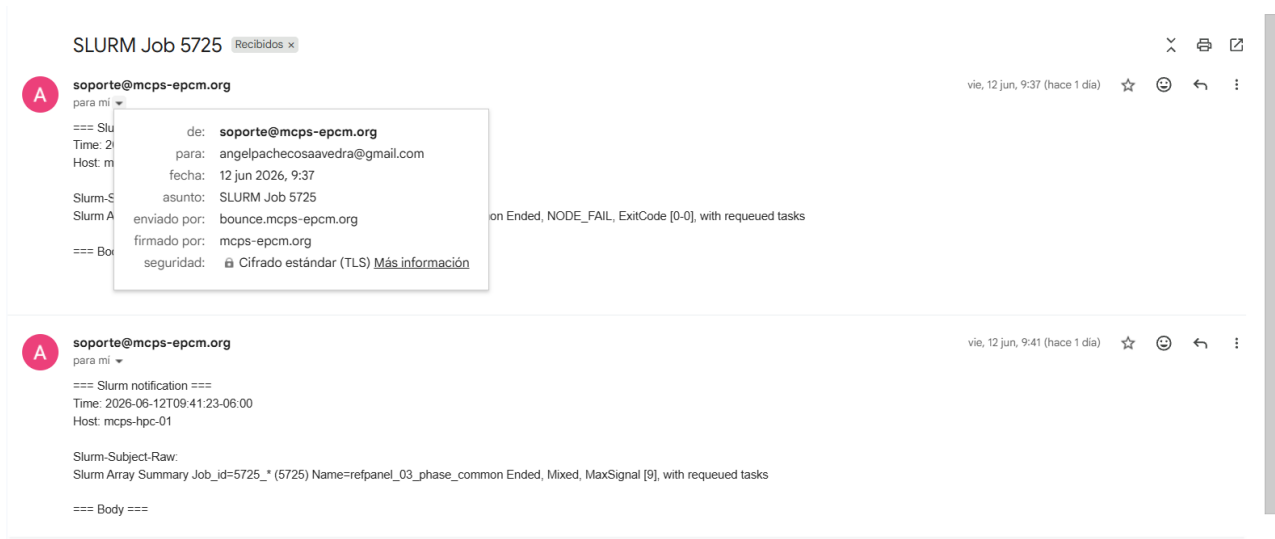


Figura 13. Correo de estados de job generador por slurm

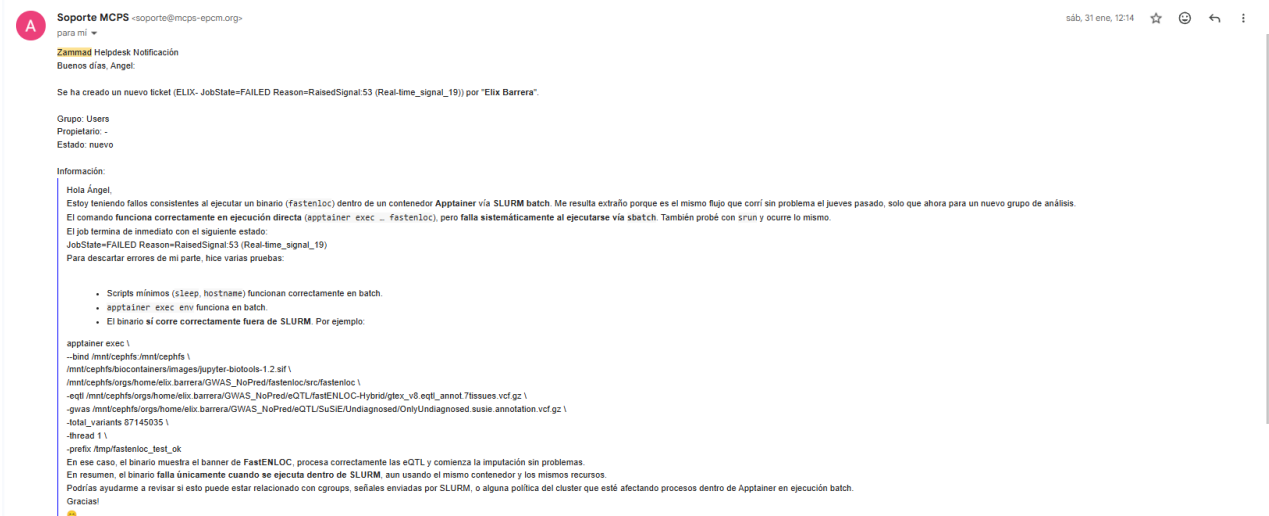


Figura 14. Notificación de creación de un ticket por un usuario

6.5 Uso de Ceph como base de almacenamiento para servicios y flujos de trabajo

Todos los servicios web del nodo maestro almacenan sus datos persistentes en CephFS, dentro de la estructura de `/mnt/cephfs/hot/`. Cada servicio tiene su propio subdirectorio:

Servicio	Ruta en CephFS	Contenido
Caddy	<code>/mnt/cephfs/hot/caddy/</code>	Caddyfile, certificados TLS internos, logs de acceso
MariaDB	<code>/mnt/cephfs/hot/mariadb/data/</code>	Base de datos de Slurm (<code>slurm_acct_db</code>) y Snipe-IT
PostgreSQL	<code>/mnt/cephfs/hot/postgres/data/</code>	Bases de datos de Nextcloud, Zammad, PGS Dashboard, SII-UNAM
Nextcloud	<code>/mnt/cephfs/hot/nextcloud/</code>	Configuración, datos internos, aplicaciones
Zammad	<code>/mnt/cephfs/hot/zammad-storage/</code>	Adjuntos de tickets, logs de actividad
Grafana	<code>/mnt/cephfs/hot/grafana/</code>	Dashboards, fuentes de datos, configuración de alertas
Prometheus	<code>/mnt/cephfs/hot/prometheus/</code>	Series de tiempo (retención de 90 días)
OnlyOffice	<code>/mnt/cephfs/hot/onlyoffice/</code>	Documentos en edición, logs, base de datos interna
Snipe-IT	<code>/mnt/cephfs/hot/snipeit/data/</code>	Inventario de hardware del clúster

La consecuencia práctica de esta arquitectura es si el nodo maestro requiere una reinstalación completa del sistema operativo como ocurrió con los nodos de cómputo documentados en la sección 5.1, ningún dato de servicio se pierde. Los contenedores se reconstruyen recreando las unidades `systemd` y apuntándolas a los mismos directorios de CephFS. La configuración de cada servicio vive dentro de sus volúmenes, no en el sistema de archivos local del nodo.

Cada unidad de servicio `systemd` incluye una directiva `ExecStartPre` que espera en un bucle hasta que el directorio correspondiente de CephFS esté disponible. Si el nodo arranca y CephFS tarda unos segundos en montarse, algo normal después de un reinicio, los contenedores no

fallan al inicio sino que esperan pacientemente. Sin esta precaución, un reinicio del nodo maestro provocaría que todos los servicios arrancaran contra un punto de montaje vacío, corrompiendo sus bases de datos al crear archivos nuevos sobre un directorio que debería contener datos existentes.

El clúster también aloja servicios complementarios que no se documentan en detalle pero que forman parte del ecosistema, Grafana y Prometheus para monitoreo del clúster (desarrollado en la sección 7 de validación), pgAdmin para administración visual de PostgreSQL, Cockpit para administración web del nodo maestro, un sitio estático de documentación (docs.mcps-epcm.org) construido con Hugo y servido por Nginx, un dashboard del PGS Catalog para visualización de puntajes de riesgo poligénico, Snipe-IT para inventario de hardware, Coturn y Janus como servidores WebRTC para videollamadas integradas en Nextcloud Talk, y aplicaciones web de investigación como la encuesta de tianguis y el sistema SII-UNAM de cuestionarios.

7. Validación operativa del sistema

7.1 Metodología de validación

La validación del clúster se estructuró en tres niveles. El primero verificó el funcionamiento correcto de cada componente individual: que Slurm asigne recursos según las QoS definidas, que CephFS sea accesible y consistente desde todos los nodos, y que los servicios web respondan correctamente. El segundo nivel evaluó la capacidad del sistema para operar en condiciones multiusuario, verificando el aislamiento entre cuentas, el cumplimiento de cuotas y la ejecución concurrente de trabajos sin interferencia. El tercero y el más relevante para los objetivos de este trabajo, sometió al clúster a cargas de trabajo genómicas reales del MCPS, validando que la infraestructura es capaz de ejecutar los análisis que motivaron su construcción.

La validación fue operativa, el clúster se consideró funcional cuando los investigadores pudieron ejecutar sus pipelines de producción sin errores atribuibles a la infraestructura. Los resultados que se presentan en las secciones siguientes corresponden a ejecuciones reales realizadas entre febrero y abril de 2026.

7.2 Validación del gestor de trabajos y ejecución en entorno multiusuario

La verificación de Slurm se realizó en dos etapas. La primera confirmó que los mecanismos de asignación de recursos funcionan según lo configurado. Se enviaron trabajos con distintas QoS y se verificó que Slurm rechazara solicitudes que excedieran los límites definidos, un usuario con QoS small no puede solicitar más de 8 cores ni más de 32 GB de memoria. Se confirmó que ReturnToService=2 funciona correctamente: después de un reinicio no planificado de un nodo de cómputo, el nodo volvió al estado idle sin intervención manual.

La segunda etapa verificó la ejecución concurrente. Durante la conversión masiva de VCF a Zarr (descrita en la sección 7.3), Slurm distribuyó simultáneamente hasta tres trabajos de array en los nodos HPC02, HPC03 y HPC04, cada uno solicitando 16 cores y 128 GB de RAM. Los logs de los trabajos confirman que cada job se ejecutó en un nodo distinto sin contención:

Job	Cromosoma	Nodo asignado	CPUs	RAM solicitada
738	chr10	MCPS-HPC04	16	128 GB
739	chr11	MCPS-HPC03	16	128 GB
740	chr12	MCPS-HPC02	16	128 GB

Los tres trabajos corrieron en paralelo y terminaron sin errores. Cuando los tres primeros finalizaron, Slurm lanzó automáticamente los siguientes cromosomas de la cola del array job en los nodos que se liberaron. El backfill funcionó según lo esperado: trabajos pequeños de otros usuarios se ejecutaron en los recursos libres de nodos donde el array job no ocupaba la totalidad de la memoria.

La contabilidad con slurmdbd registró correctamente el consumo de recursos de cada trabajo. Los datos de sacct permitieron verificar que los cores y la memoria asignados correspondían a lo solicitado, y que los tiempos de ejecución reales coincidían con los registrados en los logs de los jobs.

7.3 Validación de cargas de trabajo genómicas y epidemiológicas representativas

Se ejecutaron tres tipos de cargas de trabajo representativas de la investigación del MCPS para validar que el clúster soporta los análisis que motivaron su construcción.

Conversión VCF a Zarr (vcf2zarr). Se convirtieron los 23 archivos VCF del Freeze 150k del MCPS (22 autosomas + cromosoma X, ~150,000 muestras cada uno) al formato Zarr columnar, un requisito previo para los análisis de GWAS con sgkit. Cada cromosoma se procesó como un trabajo independiente dentro de un Slurm array job, solicitando 16 cores y 128 GB de RAM. El pipeline ejecutó dos etapas dentro de un contenedor Apptainer (jupyter-biotools-1.2.sif) vcf2zarr explora para descomponer el VCF en columnas intermedias, y vcf2zarr encode para generar el archivo Zarr final. Los datos se leían y escribían en CephFS (/mnt/cephfs/hot_nvme/mcps/imputed-topmed/).

Los 23 cromosomas se procesaron exitosamente. Los tiempos de ejecución variaron según el tamaño del cromosoma:

Cromosoma	Variantes	Datos procesados	Tiempo explora	Tiempo encode	Tiempo total	Nodo
chr1	3.31M	4.20 TB	2h 01m	1h 25m	3h 31m	MCPS-HPC02
chr2	3.56M	4.52 TB	4h 20m	2h 54m	7h 23m	MCPS-HPC04
chr3	2.99M	3.79 TB	1h 50m	1h 16m	3h 08m	MCPS-HPC02
chr11	2.04M	2.58 TB	1h 23m	0h 59m	2h 27m	MCPS-HPC03
chr21	545k	690 GB	0h 24m	0h 16m	0h 41m	MCPS-HPC03
chr22	548k	694 GB	0h 23m	0h 17m	0h 41m	MCPS-HPC02

chrX	1.62M	2.05 TB	1h 39m	0h 40m	2h 21m	MCPS- HPC02
-------------	-------	---------	--------	--------	--------	----------------

El cromosoma 2 fue el más demandante, 3.56 millones de variantes, 4.52 TB de datos procesados, 7 horas 23 minutos de ejecución total. El cromosoma 21 fue el más rápido, 41 minutos. El throughput de la etapa encode osciló entre 417 MB/s y 849 MB/s, dependiendo de la carga concurrente sobre CephFS en el momento de la ejecución. La conversión completa de los 23 cromosomas se realizó en aproximadamente 48 horas de tiempo de reloj, distribuidas en 3 nodos trabajando en paralelo. Los datos mostrados en la tabla anterior fueron el resultado de la conexión por cable ethernet.

Durante la ejecución con la red de cobre a 1 Gbps, varios cromosomas presentaron errores de lectura y escritura en CephFS que obligaron a reejecutarlos. Los errores no seguían un patrón reproducible: el mismo cromosoma fallaba en etapas distintas en cada intento. Las 48 horas de tiempo total incluyen esas reejecuciones.

Tras la migración de la red interna a fibra óptica SFP+ a 10 Gbps (descrita en la sección 4.3), se reejecutaron algunos de los cromosomas más demandantes para cuantificar el efecto del cambio de red sobre el pipeline. Los tiempos se redujeron entre 30 minutos y 1 hora por cromosoma respecto a los registrados con cobre, y no se presentó ningún error de lectura ni escritura. No se repitió la conversión completa de los 23 cromosomas porque los archivos Zarr generados con la red de cobre ya se habían validado y estaban en uso por los investigadores; la reejecución parcial fue suficiente para confirmar la mejora.

Se ejecutó un pipeline de cálculo de puntajes de riesgo poligénico en tres etapas; extracción de variantes del Zarr, matching contra el catálogo PGS, y cómputo del score con GPU. La primera etapa corrió en un solo nodo con 16 cores. La segunda y tercera utilizaron Spark distribuido en 4 nodos con 24 cores cada uno, coordinando la lectura paralela de los datos Zarr desde CephFS. La validación confirmó que Spark puede operar correctamente sobre el clúster cuando se lanza a través de Slurm, con los workers comunicándose entre nodos a través de la red interna.

Cálculo de PRS con Dask distribuido. Un pipeline alternativo usó Dask Distributed para orquestar el cálculo de PRS cromosoma por cromosoma en 4 nodos con 8 cores y 100 GB cada uno. El scheduler de Dask se lanzó en el nodo master del job, y los workers en los demás nodos. Cada worker procesó un cromosoma independientemente, leyendo los datos del Zarr en CephFS y escribiendo los resultados parciales. Los 23 cromosomas se procesaron en paralelo sin errores.

Las tres cargas de trabajo validaron que los componentes del clúster funcionan integradamente, Slurm asigna los recursos correctos, Apptainer ejecuta las herramientas dentro de contenedores con acceso a CephFS, los datos se leen y escriben concurrentemente desde múltiples nodos sin corrupción ni conflictos de permisos, y los frameworks de cómputo distribuido (Spark, Dask) operan correctamente sobre la infraestructura.

7.4 Validación del almacenamiento distribuido (Ceph)

7.4.1 Pruebas funcionales de disponibilidad y acceso concurrente

La ejecución simultánea de los array jobs de vcf2zarr en tres nodos constituyó la prueba de concurrencia más exigente del almacenamiento. Tres trabajos leyendo archivos VCF de decenas de gigabytes y escribiendo archivos Zarr de terabytes sobre el mismo filesystem CephFS, al mismo tiempo, sin errores de lectura ni de escritura. El throughput sostenido de hasta 849 MB/s en la etapa de encode confirma que CephFS, con la configuración híbrida HDD+NVMe documentada en la sección 5.5, responde adecuadamente a las demandas de I/O de las cargas bioinformáticas.

El estado del clúster Ceph durante la operación normal muestra 20 OSD activos, 5 monitores en quórum, 2 MDS activos con 1 en espera, y una distribución de datos con varianza de 1.16 entre los OSD (es decir, los discos más cargados tienen un 16% más de datos que los menos cargados, un valor aceptable para un clúster de 5 nodos con discos heterogéneos).

7.4.2 Verificación de permisos y aislamiento entre usuarios

Se verificó el aislamiento entre usuarios creando una cuenta de prueba (prueba) y ejecutando intentos de acceso a recursos protegidos. Los resultados confirmaron que las ACLs funcionan correctamente:

Operación intentada	Resultado
cd /mnt/cephfs/orgs/home/angel.pacheco/	Permission denied
cd /mnt/cephfs/hot/	Permission denied
cd /mnt/cephfs/hot_nvme/mcps/bases/Bases_DNAnexus/	Acceso concedido (lectura)
ls en Bases_DNAnexus	Lista archivos correctamente
rm ancestry-proportions-k4.txt en Bases_DNAnexus	Permission denied

El usuario prueba pudo navegar los datos genómicos de referencia y listar su contenido (lectura permitida), pero no pudo eliminar ni modificar archivos (escritura denegada). El acceso al directorio home de otro usuario fue rechazado. El acceso a /mnt/cephfs/hot/, donde residen los datos de los servicios web, también fue denegado. El directorio scratch fue accesible con permisos de lectura y escritura, como se diseñó.

7.5 Resultados de la validación

El clúster cumple con los requerimientos funcionales y no funcionales definidos en la sección 3.4. Las siguientes tablas resumen la correspondencia entre cada requerimiento y la evidencia obtenida durante la validación.

Requerimientos funcionales

Requerimiento	Evidencia de validación	Resultado
Gestión de trabajos usando Slurm	Ejecución de array jobs de vcf2zarr (23 cromosomas), distribución automática en 3 nodos simultáneos, cola con backfill funcionando, contabilidad registrada en slurmdbd	Cumple

Cuentas de usuario individuales con cuotas	Cuenta de prueba creada con UID explícito, directorio home en CephFS con cuota de 500 GB, ACL nominales verificadas	Cumple
Almacenamiento compartido visible desde todos los nodos	23 cromosomas leídos y escritos concurrentemente desde MCPS-HPC02, HPC03 y HPC04 sobre el mismo CephFS sin errores ni inconsistencias	Cumple
Entornos de análisis interactivos vía web	JupyterHub y RStudio Server accesibles por navegador con autenticación PAM, con acceso a datos de CephFS y capacidad de enviar trabajos a Slurm desde el entorno	Cumple
Plataforma de almacenamiento colaborativo	Nextcloud operativo con acceso público, integrado con los directorios home de CephFS, edición de documentos con OnlyOffice	Cumple
Sistema de atención a incidencias	Zammad operativo en soporte.mcps-epcm.org, recepción de tickets por correo electrónico, seguimiento y resolución registrados	Cumple
Documentación y guía de uso	Sitio web estático en docs.mcps-epcm.org con guías de uso de Slurm, plantillas de scripts sbatch, catálogo de 4,989 herramientas disponibles en el contenedor y documentación de los 6 entornos Conda	Cumple
Ejecución de aplicaciones por medio de contenedores	Los tres pipelines validados (vcf2zarr, PRS Spark, PRS Dask) se ejecutaron dentro de jupyter-biotools-1.2.sif con bind mounts a CephFS	Cumple
Soporte para cargas de trabajo genómicas y epidemiológicas	Conversión VCF→Zarr del Freeze 150k completo (23 cromosomas, ~150,000 muestras), cálculo de PRS con Spark distribuido en 4 nodos, cálculo de PRS con Dask distribuido en 4 nodos	Cumple

Requerimientos no funcionales

Requerimiento	Evidencia de validación	Resultado
Operación continua	El clúster opera continuamente; el nodo maestro acumula 71 días de uptime, HPC03 y HPC05 acumulan 47 días. La caída y reinstalación de HPC02 y HPC04 no interrumpió la operación del resto del sistema	Cumple
Aislamiento entre usuarios	El usuario prueba no pudo acceder al home de angel.pacheco (Permission denied), no pudo acceder a /mnt/cephfs/hot/ (Permission denied), pudo leer datos de referencia en hot_nvme pero no pudo eliminar archivos (Permission denied)	Cumple
Acceso externo restringido	Acceso SSH exclusivo por VPN + llave pública Ed25519, contraseña deshabilitada en sshd_config, Fail2ban con bloqueo tras 3 intentos, nodos de cómputo sin acceso directo desde internet	Cumple
Capacidad de procesamiento concurrente	3 trabajos de vcf2zarr ejecutándose simultáneamente en 3 nodos distintos, cada uno con 16 cores y 128 GB, sin interferencia ni degradación	Cumple
Crecimiento horizontal	La arquitectura de Ceph y Slurm soporta la adición de nodos sin reinstalación; se demostró con la reintegración de HPC02 y HPC04 tras reinstalación de SO	Cumple
Inmutabilidad de entornos	Los pipelines se ejecutaron dentro de imágenes .sif de solo lectura almacenadas en CephFS; la imagen jupyter-biotools-1.2.sif usada en febrero 2026 es la misma que se usó en abril 2026	Cumple
Protección contra pérdida de datos	CephFS con factor de replicación 2, 20 OSD activos, scrubbing periódico. No se implementaron respaldos externos automatizados	Cumple parcialmente

Administración con personal reducido	Todo el despliegue, configuración, mantenimiento y resolución de incidencias fue realizado por una sola persona (el autor de este trabajo)	Cumple
---	--	--------

La protección contra pérdida de datos se marca como cumplimiento parcial porque la replicación factor 2 de Ceph tolera la falla de un disco sin pérdida de información, pero no existe un respaldo externo automatizado que proteja ante un evento catastrófico que afecte al clúster completo (incendio, robo, falla eléctrica prolongada). Los datos genómicos originales del MCPS existen en copias externas mantenidas por los colaboradores de Oxford, lo que mitiga parcialmente el riesgo. La implementación de respaldos externos se identifica como línea de trabajo futuro en la sección 9.2.

8. Discusión

8.1 Análisis de los resultados obtenidos

El clúster transformó la dinámica de trabajo del grupo de investigación. La situación descrita en el capítulo 3, una estación de trabajo compartida que se apagaba cuando múltiples usuarios coincidían, sin aislamiento de recursos, sin gestión formal de cuentas, con un presupuesto de \$1,000 dólares mensuales en DNAnexus que se agotaba en una semana con una carga de trabajo alta, quedó resuelta. Los investigadores que usan el clúster confirman que la disponibilidad continua del sistema es lo que más valoran. Antes, un equipo que se apagaba a mitad de un análisis significaba días o semanas de retrabajo. Ahora, si el recurso existe y está disponible, Slurm lo asigna. Si no, el trabajo espera en cola. Nadie interrumpe el trabajo de nadie.

No todos los investigadores migraron. Algunos siguen usando DNAnexus porque su interfaz gráfica les resulta más accesible que la línea de comandos y Slurm. La curva de aprendizaje de un sistema Linux con gestor de trabajos por lotes no es trivial para usuarios acostumbrados a resolver todo con un clic. Sin embargo, quienes adoptaron el clúster dejaron de preocuparse por los costos de ejecución, contrario a la tensión constante que generaba el esquema anterior, donde el presupuesto mensual se repartía entre todos los investigadores y su agotamiento prematuro cancelaba los trabajos de todos.

El modelo del clúster cumplió su propósito, con matices. Los procesadores de consumo (Intel Core i9, Core Ultra 9, AMD Threadripper) ofrecen frecuencias de reloj altas y un número razonable de núcleos para cargas bioinformáticas que son predominantemente paralelas a nivel de datos (cada muestra o cromosoma se procesa por separado). Las 23 conversiones de VCF a Zarr del Freeze 150k se completaron en 48 horas repartidas en 3 nodos, un resultado funcional para el volumen de datos del MCPS.

Donde el hardware de consumo muestra sus limitaciones es en el subsistema de memoria y entrada/salida. Los procesadores empresariales (Intel Xeon, AMD EPYC) tienen entre 8 y 12 canales de memoria frente a los 2 canales de los procesadores de escritorio, lo que les da un ancho de banda hacia RAM del orden de 460 GB/s contra aproximadamente 100 GB/s en escritorio. Las líneas PCIe disponibles son también inferiores, 20-24 líneas en procesadores de consumo frente a 128 en un EPYC, lo que limita el rendimiento simultáneo de los NVMe y la GPU. La interconexión entre núcleos refleja la misma brecha. Los procesadores de escritorio usan topologías de anillo o tiles desagregados donde la latencia de comunicación entre núcleos es impredecible y se degrada conforme aumenta la carga, un núcleo que necesita un dato que está en la caché de otro debe "recorrer" el anillo parando en cada estación. Los procesadores empresariales usan topologías de malla con rutas deterministas y protocolos de coherencia de caché basados en directorio, lo que escala sin degradación. Los procesadores de escritorio tampoco soportan configuraciones multi-socket, lo que limita la capacidad máxima de un solo nodo.

La migración de la red interna de 1 Gbps a 10 Gbps con fibra óptica SFP+ eliminó el cuello de botella de red más evidente. Con la red de 1 Gbps, el throughput de CephFS se degradaba visiblemente cuando tres nodos accedían al almacenamiento al mismo tiempo. Con 10 GbE, el ancho de banda de red dejó de ser el factor limitante y las restricciones remanentes son las que impone el propio hardware de los procesadores y los discos: los canales de memoria, las líneas PCIe y la latencia mecánica de los HDD de 20 TB. Para las cargas actuales del MCPS, el rendimiento es suficiente. Para un crecimiento significativo del volumen de datos o del número de usuarios concurrentes, estas limitaciones requerirán atención.

8.2 Limitaciones del sistema implementado

Las limitaciones remanentes del clúster son de tres tipos: de hardware, de adopción por parte de los usuarios y de operación.

En hardware, el cuello de botella ya no es la red, la migración a 10 GbE resolvió eso, sino los procesadores de consumo. Dos canales de memoria frente a ocho o doce en procesadores empresariales limitan el ancho de banda hacia RAM. Pocas líneas PCIe (20-24) restringen el rendimiento simultáneo de los NVMe, la GPU y la NIC de 10 GbE, discos duros. Las GPUs RTX 3060 con 12 GB de VRAM son insuficientes para modelos de deep learning que requieren 24-80 GB. Estas son limitaciones inherentes a la elección de hardware de consumo y solo se resuelven reemplazando componentes.

La barrera de adopción por parte de los usuarios es una limitación no técnica pero igualmente real. Investigadores acostumbrados a interfaces gráficas encuentran intimidante trabajar con la terminal, escribir scripts sbatch y navegar un sistema de archivos Linux. La documentación en docs.mcps-epcm.org mitiga parcialmente el problema, pero no lo elimina. Una alternativa explorable es Open OnDemand, una plataforma web desarrollada por el Ohio Supercomputer Center que actúa como portal web para clústeres HPC, ofreciendo lanzamiento de trabajos, acceso a JupyterHub y RStudio, y gestión de archivos desde el navegador, todo integrado con Slurm sin que el usuario necesite escribir un solo comando en terminal.

Otra opción pendiente de evaluación es la integración de JupyterHub y RStudio Server con Slurm para asignar recursos por usuario. En la configuración actual, ambos servicios corren en el nodo maestro y consumen recursos dinámicamente sin control de Slurm. Si un usuario lanza un cálculo pesado desde JupyterHub, compite por CPU y memoria con los demás usuarios del servicio y con los contenedores de otros servicios web. Con un spawner de JupyterHub integrado con Slurm, cada sesión de notebook se lanzaría como un trabajo de Slurm en un nodo de cómputo, con recursos limitados por la QoS del usuario. El reto de este modelo es afinar las políticas de tiempo límite (Walltime) y las particiones en Slurm para evitar que sesiones interactivas inactivas monopolicen los recursos de los nodos de cómputo por días, garantizando al mismo tiempo que los investigadores puedan desconectarse y retomar sus análisis horas después sin perder el progreso.

La ausencia de respaldos externos automatizados es una limitación que el capítulo 7 identificó como cumplimiento parcial del requerimiento de protección contra pérdida de datos. La replicación factor 2 de Ceph protege contra fallas de disco individuales, pero un evento que afecte al clúster completo provocaría la pérdida de los datos derivados del procesamiento. Los datos genómicos originales del MCPS existen en copias externas, pero los resultados de los investigadores que tardan días en ejecutarse no tienen respaldo fuera del clúster.

8.3 Consideraciones operativas: mantenimiento, monitoreo y continuidad

El clúster es administrado por una sola persona. Todo el despliegue, la configuración, el mantenimiento, la resolución de incidencias y la atención a usuarios documentados en este trabajo fueron realizados por el autor. Si el administrador no está disponible por vacaciones, enfermedad o cambio de adscripción, no existe un segundo operador capacitado para intervenir. La documentación interna y los procedimientos estandarizados (como el checklist de reinstalación de nodos documentado en la sección 5.1) reducen el riesgo, pero no lo eliminan.

La experiencia operativa acumulada durante los meses de operación del clúster generó un conjunto de lecciones que condicionaron decisiones de diseño y configuración. El conflicto de UID entre slurm y dnsmasq en Ubuntu 24.04 (sección 5.1) obligó a estandarizar un orden estricto de instalación. Los IOPS incorrectos de Ceph (sección 5.5) provocaron latencias de 400 ms y reportes falsos de inconsistencia hasta que se ajustó manualmente el límite a 100 IOPS para los HDD de 20 TB. La habilitación del performance mode en BIOS corrompió el sistema operativo de un nodo, lo que llevó a adoptar una configuración conservadora con XMP deshabilitado y frecuencia de memoria fijada a 4000 MHz. La expiración inadvertida del certificado TLS del servidor de correo interrumpió el servicio de correo durante días hasta que se implementó la automatización con systemd timers (sección 6.4).

El incidente más complejo fue la aparición de errores CRC32 en CephFS durante la construcción del panel de referencia MCPS10k. El pipeline de concatenación y normalización de variantes (bcftools concat + norm) fallaba intermitentemente al procesar archivos BCF de entre 63 y 374 GB, reportando errores de checksum en offsets diferentes cada ejecución. El patrón errático, el mismo archivo fallaba en posiciones distintas cada vez, descartaba corrupción en disco y apuntaba a corrupción en tránsito.

El análisis de las estadísticas TCP reveló que un nodo (MCPS-HPC04) acumulaba 640,000 retransmisiones frente a las 35,000-45,000 de los demás. Los logs del kernel de ese nodo confirmaron el diagnóstico, libceph: bad crc/signature indicaba que los datos llegaban corruptos por la red, y ceph_con_workfn hogged CPU for >10000us revelaba que las funciones de red del kernel no obtenían tiempo de CPU cuando bcftools saturaba los 24 cores.

La causa raíz fue que MCPS-HPC04 reinstalado previamente tras una falla de sistema operativo, no tenía instalado irqbalance. Sin este demonio, las 24 colas de interrupción de la NIC de 10 GbE caían en un solo core. Cuando ese core se saturaba con cómputo, los paquetes de red no se atendían, el ring buffer de la NIC se desbordaba y Ceph recibía bloques incompletos. La instalación de irqbalance y la expansión de los ring buffers de 512 a 8,192 entradas resolvieron el problema. Los cuatro nodos restantes nunca presentaron el error porque irqbalance ya estaba activo en ellos.

El mismo incidente reveló que los NVMe tenían habilitada la transición autónoma de estados de energía (APST), que les permitía entrar en modo de bajo consumo entre operaciones. Al despertar, la primera operación sufría latencias de hasta 100 ms, suficientes para que BlueStore abortara transacciones y provocara la caída simultánea de dos OSDs que compartían el mismo NVMe como WAL/DB. La desactivación del APST a nivel de GRUB eliminó estos picos de latencia.

Tres lecciones se derivan de este incidente. La primera: un solo paquete faltante en un solo nodo causó días de fallos intermitentes. irqbalance no aparece en los tutoriales de Ceph ni es un paquete llamativo, pero sin él un enlace de 10 Gbps bajo carga HPC pierde paquetes en cuanto un core se satura. La segunda: los errores transitorios de CephFS aparecen como errores de aplicación (bcftools CRC32 mismatch) y es fácil culpar a la herramienta o al archivo. La clave para el diagnóstico fue que el error aparecía en offsets diferentes en cada ejecución. La tercera: en un clúster hiperconvergente donde los mismos nodos corren cómputo y almacenamiento, todo tiene que cooperar, no basta con que Ceph esté bien configurado si el kernel descarta paquetes, y no basta con la red si el NVMe decide dormirse entre operaciones.

El monitoreo del clúster se realiza con Prometheus y Grafana, desplegados como contenedores Podman. Prometheus recolecta métricas de los nodos (uso de CPU, memoria, disco, red), de Slurm (trabajos activos, recursos asignados, eficiencia de CPU) y de Ceph (estado del clúster,

latencia de OSD, uso de pools). Grafana visualiza estas métricas en dashboards accesibles desde grafana.mcps-epcm.org. La retención de datos en Prometheus está configurada a 90 días. Los dashboards de monitoreo de hardware y rendimiento de Slurm están en desarrollo al momento de esta documentación.

9. Conclusiones y trabajo futuro

9.1 Conclusiones

El objetivo general de este trabajo fue diseñar e implementar un clúster HPC académico que funcionara como plataforma computacional multiusuario para la ejecución de análisis genómicos y epidemiológicos, integrando servicios de soporte y validando su operación con cargas de trabajo representativas. El objetivo se cumplió.

El clúster opera con 5 nodos, 152 cores, 1 TB de RAM agregada, 5 GPUs NVIDIA RTX 3060, y 232 TB de almacenamiento distribuido CephFS. Slurm gestiona la asignación de recursos con 6 niveles de QoS, aislamiento por cgroups y contabilidad en base de datos. Los contenedores Apptainer encapsulan las herramientas bioinformáticas sin contaminar el sistema operativo.

Los cuatro objetivos específicos se cumplieron. La arquitectura lógica del clúster se definió en el capítulo 4, estableciendo los roles de los nodos, la infraestructura de red, el modelo de acceso multiusuario y las consideraciones de seguridad. La implementación se documentó en el capítulo 5, cubriendo la instalación del sistema operativo, Slurm, Apptainer, la configuración de red y el despliegue de Ceph con arquitectura híbrida HDD+NVMe. La integración de servicios se describió en el capítulo 6, incluyendo JupyterHub, RStudio Server, Nextcloud, Zammad y el correo institucional. La validación operativa del capítulo 7 demostró que el clúster ejecuta cargas genómicas reales, la conversión de 23 cromosomas del Freeze 150k del MCPS de VCF a Zarr, y el cálculo de puntajes de riesgo poligénico con Spark y Dask distribuidos en múltiples nodos, cumpliendo con los requerimientos funcionales y no funcionales definidos.

El trabajo demuestra que un clúster HPC funcional para investigación genómica puede construirse con hardware de consumo, software de código abierto y un solo administrador. Resuelve un problema concreto: darle a un grupo de investigación académico la capacidad de procesar sus datos localmente, con control total sobre el entorno, sin depender de presupuestos mensuales de plataformas en la nube ni de la disponibilidad de recursos compartidos a nivel nacional.

9.2 Líneas de trabajo futuro

La operación del clúster reveló áreas que requieren desarrollo posterior. La implementación de respaldos externos automatizados, ya sea hacia un servidor remoto o hacia almacenamiento en la nube, es necesaria para cubrir el requerimiento de protección contra pérdida de datos que actualmente se cumple solo parcialmente.

En el ámbito de la experiencia de usuario, la evaluación de Open OnDemand como portal web para el clúster podría reducir la barrera de adopción para investigadores que no están familiarizados con la línea de comandos. La integración de JupyterHub con un spawner de Slurm permitiría asignar recursos controlados a cada sesión de notebook, evitando la competencia por recursos en el nodo maestro. La plataforma Galaxy, cuya configuración parcial ya existe en el clúster, podría ofrecer una interfaz gráfica para la ejecución de pipelines bioinformáticos estándar sin requerir conocimientos de Slurm.

Los dashboards de Grafana destinados al monitoreo del rendimiento del hardware y la evaluación de la eficiencia de Slurm se encuentran actualmente en fase de desarrollo. La Figura 15 presenta únicamente una vista general del estado del clúster, ya que aún se sigue trabajando en las métricas necesarias para analizar la eficiencia de ejecución de cada job. La finalización de estos dashboards permitirá contar con una supervisión en tiempo real de los recursos y del desempeño del clúster, además de facilitar la identificación temprana de posibles cuellos de botella y problemas de rendimiento. La automatización del procedimiento de alta de usuarios, actualmente manual en cuatro fases, con un script unificado reduciría el tiempo de aprovisionamiento y minimizaría errores.



Figura 15. Dashboard general del clúster

10. Referencias

- Tapia-Conyer, R., Kuri-Morales, P., Alegre-Díaz, J., Whitlock, G., Emberson, J., Clark, S., Peto, R. y Collins, R. (2006). Cohort profile: The Mexico City Prospective Study. *International Journal of Epidemiology*, 35(2), 243–249. <https://doi.org/10.1093/ije/dyl042>
- Ziyatdinov, A., Torres, J., Alegre-Díaz, J., Backman, J., Mbatchou, J., Turner, M., Gaynor, S. M., Kessler, M. D., Majority, L. O., Regeneron Genetics Center, Collins, R., Emberson, J. y Baras, A. (2023). Genotyping, sequencing and analysis of 140,000 adults from Mexico City. *Nature*, 622(7984), 784–793. <https://doi.org/10.1038/s41586-023-06595-3>
- Eijkhout, V. (2010). *Introduction to High Performance Scientific Computing* (2.^a ed.). Lulu. ISBN: 978-1-257-99254-6
- Sterling, T., Anderson, M. y Brodowicz, M. (2018). *High Performance Computing: Modern Systems and Practices*. Morgan Kaufmann.
- Yoo, A. B., Jette, M. A. y Grondona, M. (2003). SLURM: Simple Linux Utility for Resource Management. En *Job Scheduling Strategies for Parallel Processing* (pp. 44–60). Springer. https://doi.org/10.1007/10968987_3
- Buffalo, V. (2015). *Bioinformatics Data Skills: Reproducible and Robust Research with Open Source Tools*. O'Reilly Media.
- Kurtzer, G. M., Sochat, V. y Bauer, M. W. (2017). Singularity: Scientific containers for mobility of compute. *PLOS ONE*, 12(5), e0177459. <https://doi.org/10.1371/journal.pone.0177459>
- Merkel, D. (2014). Docker: Lightweight Linux containers for consistent development and deployment. *Linux Journal*, 2014(239), 2.
- Johnson, R. (2025). *Ceph architecture and administration: Definitive reference for developers and engineers*. HiTeX Press.
- Flores Salgado L. Y., Quintanar Sierra J. J. C. & Universidad Nacional Autónoma de México. (2018). Configuración y puesta en operación de un Clúster HPC para la Unidad MOFABI de la Facultad de Ingeniería de la UNAM.
- Ceph Foundation. (s.f.). mClock Config Reference. *Ceph Documentation*. <https://docs.ceph.com/en/latest/rados/configuration/mclock-config-ref/>

Anexos

Anexo A

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
Unidad de Medicina Experimental de la Facultad de Medicina UNAM

POLÍTICAS DE ACCESO Y USO DEL CLÚSTER DE ALTO RENDIMIENTO MCPS-HPC

Clúster de Cómputo de Alto Rendimiento del Estudio Prospectivo de la Ciudad de México (MCPS-HPC, por sus siglas en inglés: Mexico City Prospective Study - High Performance Computing)

Documento alineado con la norma
ISO/IEC 27001:2022
Sistema de Gestión de Seguridad de la Información

Versión 2.0
Fecha de emisión: Febrero 2026
Fecha de revisión: Enero 2026

CONTROL DEL DOCUMENTO

Nombre del documento	Políticas de Acceso y Uso del Clúster MCPS-HPC
Código	POL-MCPS-HPC-001
Versión	2.1
Fecha de emisión	Febrero 2026
Elaborado por	Administración del Clúster MCPS-HPC
Aprobado por	Comité Técnico del MCPS
Marco normativo	ISO/IEC 27001:2022 - Controles A.5, A.8, A.9, A.12, A.18

Historial de Revisiones

Versión	Fecha	Descripción	Responsable
1.0	Dic 2025	Versión inicial	Admin. MCPS-HPC
2.0	Ene 2026	Aclaración de sanciones, QoS y asignación de recursos, autenticación PAM, Definición de Encargado del MCPS. Definición de Administrador del Cluster MCPS-HPC	Admin. MCPS-HPC
2.1	Febrero 2026	Definición de Jefe de Bioinformática, Modificación de QoS, Creación de Glosario Técnico, Procedimiento de sanciones con derecho de réplica (24h), Subsección de contenedores (Apptainer), Artículo de citación en publicaciones	Admin. MCPS-HPC

CAPÍTULO I. DISPOSICIONES GENERALES

1.1 Objetivo

Artículo 1. El presente documento establece las políticas, normas y lineamientos para el acceso y uso adecuado del Clúster MCPS-HPC de la Unidad de Medicina Experimental de la Facultad de Medicina UNAM, con el fin de garantizar la seguridad, integridad, confidencialidad y disponibilidad de los recursos computacionales y la información procesada, conforme a los principios establecidos en la norma ISO/IEC 27001:2022.

1.2 Alcance

Artículo 2. Estas políticas aplican a todos los usuarios del Clúster MCPS-HPC, incluyendo investigadores, técnicos académicos, estudiantes de posgrado y colaboradores externos autorizados que participen en proyectos de investigación del Estudio Prospectivo de la Ciudad de México (Mexico City Prospective Study, MCPS).

1.3 Marco Normativo (ISO 27001:2022)

Artículo 3. Las presentes políticas se encuentran alineadas con los siguientes controles de la norma ISO/IEC 27001:2022:

- **A.5 Políticas de Seguridad de la Información:** Directrices de gestión para la seguridad.
- **A.8 Gestión de Activos:** Responsabilidad sobre los activos de información.
- **A.9 Control de Acceso:** Gestión de acceso de usuarios y responsabilidades.
- **A.12 Seguridad de las Operaciones:** Protección contra software malicioso y respaldo de información.
- **A.18 Cumplimiento:** Cumplimiento de requisitos legales y contractuales.

1.4 Definiciones

Artículo 4. Para efectos de este documento, se establecen las siguientes definiciones:

Término	Definición
MCPS-HPC	Clúster de Cómputo de Alto Rendimiento del Estudio Prospectivo de la Ciudad de México (MCPS-HPC, por sus siglas en inglés: Mexico City Prospective Study - High Performance Computing). Infraestructura computacional compuesta por múltiples nodos de cómputo interconectados para procesamiento paralelo.
Administrador del Clúster	Angel Gael Pacheco Saavedra. Responsable técnico responsable de la gestión, mantenimiento, seguridad y operación del Clúster MCPS-HPC. Autoridad técnica para la suspensión inmediata de cuentas ante incidentes de seguridad.
Jefe de Bioinformática	Fernando Javier Rivas Reyes. Consulta y dirección de los asuntos de bioinformática de la Organización
Encargado del MCPS	Mtro. Jesús Alegre Díaz, Encargado del Estudio Prospectivo de la Ciudad de México. Autoridad responsable de determinar acciones administrativas y legales derivadas de incumplimientos graves.
Usuario	Persona autorizada para utilizar los recursos del MCPS-HPC. Cada usuario tiene una cuenta individual e intransferible y es responsable total de todas las actividades realizadas con su cuenta.
Datos Genéticos de la Cohorte	Información genómica y de salud correspondiente al estudio MCPS, clasificada como información sensible y confidencial de conformidad con la Ley Federal de Protección de Datos Personales en Posesión de los Particulares.

Datos Epidemiológicos	Conjunto de información fenotípica, sociodemográfica, clínica y bioquímica recabada de los participantes del MCPS. Incluye variables de estilo de vida, historial médico, mediciones antropométricas y resultados de análisis de sangre (perfiles bioquímicos). Estos datos se clasifican como información confidencial y están protegidos bajo los mismos controles de seguridad que los datos genéticos. Los cuales deberán estar disponibles en el Cluster MCPS-HPC para uso de los investigadores.
QoS (Quality of Service)	Nivel de calidad de servicio asignado a un usuario que determina los límites de recursos computacionales (CPUs, memoria, GPUs, tiempo de ejecución) que puede utilizar.
Proyecto de Investigación	Actividad de investigación científica formalmente registrada para el uso del MCPS-HPC, con objetivos, vigencia, participantes y recursos definidos. Cada usuario es responsable de un proyecto debe proporcionar esta información al solicitar acceso.
Slurm	Sistema de gestión de colas y planificación de trabajos utilizado en el MCPS-HPC para distribuir recursos computacionales de manera equitativa entre usuarios.
Tiempo Máximo (QoS)	Tiempo máximo de ejecución permitido por trabajo individual. Una vez alcanzado este límite, el trabajo será terminado automáticamente por el sistema de colas (Slurm). Los tiempos máximos varían según la partición asignada.
Job/Trabajo	Tarea computacional enviada al sistema de colas para su ejecución. Cada trabajo solicita recursos específicos (CPUs, memoria, GPUs, tiempo) y se ejecuta según disponibilidad.
Comité Técnico del MCPS	Cuerpo colegiado integrado inicialmente por el Administrador del Clúster MCPS-HPC, el Jefe de Bioinformática y el Encargado del MCPS. La conformación de este comité podrá ser modificada o ampliada siempre que cuente con la autorización conjunta de los titulares mencionados. Requisito de ingreso: Cualquier nuevo integrante deberá acreditar formación profesional y experiencia comprobable en ambientes de Tecnologías de la Información (TI).
Datos de la Organización	Datos genéticos de Cohorte, Datos Epidemiológicos, Resultados de los análisis
Datos Complementarios	Bases de datos públicas, Referencias bibliográficas, Datos auxiliares
Resultados de Análisis	Información derivada del procesamiento de los datos de la organización dentro del Clúster MCPS-HPC, que no contiene datos genéticos crudos, identificables o sensibles, y que corresponde exclusivamente a salidas analíticas agregadas, estadísticas o anonimizadas, generadas en el marco de un proyecto de investigación aprobado.
Medios de contacto establecidos	Correo electrónico: soporte@mcps-epcm.org Web: https://soporte.mcps-epcm.org/

CAPÍTULO II. USUARIOS Y CONTROL DE ACCESO

(Control A.9 ISO 27001 - Control de Acceso)

2.1 Principios de Acceso

Artículo 1. El acceso al Clúster MCPS-HPC se rige por el principio de mínimo privilegio: cada usuario recibirá únicamente los permisos estrictamente necesarios para realizar las tareas asociadas a su proyecto, durante el tiempo que sea requerido.

Artículo 2. El acceso es personal, confidencial e intransferible. Queda estrictamente prohibido compartir credenciales de acceso con terceros, incluyendo compañeros de proyecto.

2.2 Solicitud de Acceso

Artículo 3. El acceso a los recursos del Clúster MCPS-HPC deberá solicitarse formalmente a través de los medios de contacto publicados, indicando obligatoriamente:

- Nombre completo del solicitante.
- Institución de adscripción y cargo.
- Correo electrónico.
- Proyecto de investigación y justificación de uso.
- Firmar la presente política manifestando su conformidad y compromiso de cumplimiento.

Artículo 4. El interesado deberá tener conocimientos básicos en el uso de sistemas Linux y entornos de cómputo de alto rendimiento

2.3 Gestión de Credenciales

Artículo 5. Una vez aprobada la solicitud, el Administrador del Clúster asignará las credenciales de acceso correspondientes conforme a los requisitos del proyecto.

Artículo 6. El acceso al clúster se realiza mediante autenticación por llave SSH. Las credenciales deberán cumplir con los siguientes requisitos:

- La llave privada es personal e intransferible.
- No almacenar la llave en servicios en la nube.
- No enviar la llave por medios de mensajería; correo, WhatsApp, etc.
- En caso de pérdida de la llave, notificar inmediatamente al Administrador.

2.4 Autenticación y Seguridad de Credenciales

Artículo 7. El Clúster MCPS-HPC utiliza dos mecanismos de autenticación según el tipo de acceso:

2.4.1 Acceso por Terminal (SSH)

Para conexiones por línea de comandos (SSH), el acceso se realiza **exclusivamente mediante llave ssh proporcionadas por el administrador del Cluster**:

- El usuario es responsable de la seguridad y custodia de su llave privada.
- No se permite el acceso SSH mediante contraseña.

2.3.2 Acceso a Interfaces Gráficas (RStudio Server, JupyterHub)

Para el acceso a las interfaces web (RStudio Server, JupyterHub), se requiere **autenticación mediante contraseña**:

- **Razón técnica:** Estas interfaces utilizan autenticación PAM (Pluggable Authentication Modules) del sistema operativo, que requiere validación de contraseña del usuario.
- **Requisitos de contraseña:** Mínimo recomendable 12 caracteres, combinando mayúsculas, minúsculas, números y caracteres especiales.
- **Cambio periódico:** Se recomienda cambiar la contraseña cada 90 días.
- La contraseña es personal e intransferible, no debe compartirse ni almacenarse en texto plano.

Artículo 8. IMPORTANTE: Cada usuario debe mantener tanto su llave SSH como su contraseña de forma segura. La llave SSH es para acceso por terminal; la contraseña es necesaria para RStudio Server y JupyterHub.

CAPÍTULO III. USO DE DATOS DE LA ORGANIZACIÓN

(Control A.8 ISO 27001 - Gestión de Activos de Información)

3.1 Clasificación de Datos

Artículo 1. Los datos genéticos del MCPS se clasifican como **INFORMACIÓN CONFIDENCIAL Y SENSIBLE**. Esta clasificación implica el máximo nivel de protección y restricciones de uso.

3.2 Restricciones de Uso de Datos de la Organización

Artículo 2. Los datos genéticos y epidemiológicos del MCPS sólo podrán ser utilizados, procesados y almacenados dentro del Clúster MCPS-HPC.

Queda estrictamente prohibida la descarga, copia, extracción o transferencia de los datos de la organización, total o parcialmente, fuera del clúster por iniciativa del usuario, hacia dispositivos personales, servicios en la nube o cualquier sistema externo.

Bajo ninguna circunstancia los usuarios podrán descargar datos genéticos crudos, bases de datos originales o información sensible identificable.

Artículo 3. La violación a estas restricciones constituye una falta crítica sujeta al procedimiento de sanciones descrito en el Capítulo XI de este documento.

3.2.1 Excepción controlada - Resultados de análisis

Artículo 4. Los usuarios podrán descargar únicamente los resultados de los análisis generados dentro del Clúster MCPS-HPC, siempre que dichos resultados:

- No contengan datos genéticos crudos ni información sensible identificable.
- Correspondan exclusivamente al proyecto de investigación aprobado.
- Cumplan con los principios de confidencialidad y minimización de datos.

El usuario será responsable de verificar que los archivos descargados correspondan únicamente a resultados y no a datos de la organización.

3.3 Uso de Datos Complementarios

Artículo 5. Los usuarios podrán cargar, almacenar y descargar datos complementarios en su directorio asignado, siempre que:

- Sean necesarios para el desarrollo del proyecto de investigación aprobado.
- Cumplan con las licencias de uso correspondientes.
- No comprometan la seguridad o integridad del sistema.
- Los datos cumplan con los criterios de uso establecidos en estas políticas
- No excedan la cuota de almacenamiento asignada
- Se respeten las normativas de protección de datos aplicables

CAPÍTULO IV. RESPONSABILIDADES DEL USUARIO

(Control A.8 ISO 27001 - Gestión de Activos)

4.1 Uso Responsable

Artículo 1. El usuario se compromete a utilizar los recursos del Clúster MCPS-HPC exclusivamente para los fines académicos y de investigación especificados en su solicitud de acceso.

Artículo 2. Cada usuario debe respetar la privacidad, confidencialidad y derechos individuales de los demás usuarios. Será sancionado quien realice alguna actividad que afecte el uso de los recursos de los demás usuarios. De acuerdo con lo establecido en el capítulo XI

Artículo 3. El usuario es responsable de todas las actividades realizadas desde su cuenta.

Cualquier uso indebido realizado desde una cuenta personal será responsabilidad exclusiva del titular de la misma.

Artículo 4. El usuario se compromete a otorgar el crédito correspondiente al uso de la infraestructura del MCPS-HPC en cualquier tesis, artículo científico, reporte técnico o presentación derivada de los resultados obtenidos en el clúster.

Artículo 5. Los usuarios que publiquen resultados utilizando datos genéticos, fenotípicos o de seguimiento de la cohorte MCPS deberán citar las siguientes referencias según corresponda.

Para datos genómicos y epidemiológicos:

Ziyatdinov A, et al. Genotyping, sequencing and analysis of 140,000 adults from Mexico City. *Nature*. 2023;622:784-793. DOI: [10.1038/s41586-023-06595-3](https://doi.org/10.1038/s41586-023-06595-3)

Para descripción de la cohorte:

Tapia-Conyer R, et al. Cohort profile: the Mexico City Prospective Study. *Int J Epidemiol*. 2006;35:243-249. DOI: [10.1093/ije/dyl042](https://doi.org/10.1093/ije/dyl042)

4.2 Confidencialidad de Datos

Artículo 6. Queda estrictamente prohibida la extracción de datos de la organización fuera del Clúster MCPS-HPC, con excepción de lo establecido en el Artículo 4, Capítulo 3.2.1 respecto a los resultados de análisis propios del usuario. El usuario se compromete a guardar absoluta confidencialidad sobre cualquier información sensible a la que tenga acceso.

4.3 Gestión del Almacenamiento

Artículo 7. El usuario es responsable de gestionar eficientemente su espacio de almacenamiento asignado, eliminando archivos temporales y datos obsoletos de manera regular.

4.4 Instalación de Software

Artículo 8. El usuario podrá solicitar la instalación de software adicional siempre que:

- Sean necesarios para el desarrollo del proyecto de investigación aprobado.
- Cumplan con las licencias de uso correspondientes.
- No comprometan la seguridad o integridad del sistema.

CAPÍTULO V. USO DE SOFTWARE

(Control A.12 ISO 27001 - Seguridad de las Operaciones)

5.1 Software Disponible

Artículo 1. El Clúster MCPS-HPC cuenta con software preinstalado y validado para su uso en proyectos de investigación, incluyendo herramientas de bioinformática, análisis estadístico, aprendizaje automático y procesamiento de datos genómicos.

Artículo 2. La lista detallada con versiones específicas estará disponible para usuarios registrados en <https://docs.mcps-epcm.org/>

5.2 Solicitud de Software Adicional

Artículo 3. Cualquier software adicional que requiera el usuario deberá ser solicitado formalmente a través de los medios de contacto de soporte del Clúster MCPS-HPC, indicando obligatoriamente:

- i. **Sitio oficial de descarga** del software (URL verificable).
- ii. **Licencia de uso:** Especificar si es software libre (GPL, MIT, Apache, etc.), comercial con licencia adquirida, o software académico.
- iii. **Justificación técnica:** Motivo por el cual se requiere y validación documentada de que ningún otro software instalado cumple con los requerimientos.
- iv. **Requisitos de sistema:** Dependencias, recursos computacionales estimados y compatibilidad.

Artículo 4. La instalación de software queda sujeta a la evaluación técnica del Administrador del Clúster, quien verificará la compatibilidad, seguridad y cumplimiento con las políticas de licenciamiento. El tiempo de respuesta estimado es de 5 días hábiles.

5.3 Prohibiciones de Software

Artículo 5. Queda estrictamente prohibido:

- Instalar software que viole la Ley Federal del Derecho de Autor o cualquier legislación aplicable.
- Instalar o ejecutar software que atente contra la seguridad del clúster (malware, herramientas de hacking, criptomonera).
- Modificar o alterar el software del sistema sin autorización expresa.

5.4 Uso de Contenedores

Artículo 6. El MCPS-HPC permite el uso de contenedores mediante Apptainer (anteriormente Singularity) para garantizar reproducibilidad y portabilidad de entornos de análisis.

Lineamientos:

- Los usuarios pueden ejecutar contenedores sin requerir privilegios de administrador
- Se recomienda utilizar imágenes de fuentes confiables o construir sus propias imágenes desde un archivo .def que es lo más recomendable
- Los contenedores deben cumplir con las políticas de seguridad del clúster
- El almacenamiento de imágenes cuenta contra la cuota del usuario

CAPÍTULO VI. USO DE RECURSOS COMPUTACIONALES

6.1 Responsabilidades del Usuario sobre Proyectos

Artículo 1. Cada proyecto deberá contar con un usuario responsable quien asumirá las siguientes responsabilidades:

- Proporcionar la información completa del proyecto (objetivos, vigencia, recursos estimados).
- Supervisar el uso adecuado de los recursos asignados al proyecto.
- Notificar cambios en la composición del equipo o la conclusión del proyecto.

6.2 Vigencia de Proyectos

Artículo 2. Los proyectos tendrán una vigencia definida al momento de su registro. Al término de la vigencia, el usuario responsable podrá solicitar una extensión con la debida justificación. Los proyectos sin actividad por más de 6 meses podrán ser dados de baja.

Artículo 3. Para efectos de esta política, se considerará que un proyecto está "inactivo" si no se registra ejecución de trabajos (jobs) en el sistema de colas o modificación de archivos en el almacenamiento asignado durante el periodo de 6 meses.

Nota Importante: El simple inicio de sesión (login) al sistema o la consulta de archivos (lectura) no constituyen actividad suficiente para reiniciar el contador de vigencia ni para evitar la baja del proyecto.

6.3 Principio de Asignación de Recursos

Artículo 4. **IMPORTANTE:** La asignación de recursos se realiza **por usuario individual**, no por proyecto ni por cantidad de proyectos. **Tener más proyectos NO otorga más recursos.** Cada usuario tiene límites individuales según su nivel de QoS asignado, independientemente de cuántos proyectos participe.

6.4 Sistema de Calidad de Servicio (QoS)

Artículo 5. El Clúster MCPs-HPC implementa un sistema de Calidad de Servicio (QoS) que define los recursos disponibles para cada usuario. Los niveles de QoS determinan los límites máximos de CPUs, memoria, tiempo de ejecución y trabajos simultáneos. Todos los usuarios podrán disponer de dos QoS por defecto small y normal

6.4.1 Tabla de Niveles de QoS

QoS	CPUs Máx.	Memoria Máx.	Tiempo Máx.	Jobs Simultáneos
small	8	32 GB	Sin límite	20
normal	24	150 GB	2 días	10
interactive	16	64 GB	12 horas	2 (+1 GPU)
gpu	48	128 GB	3 días	5 (+2 GPUs)
large	96	256 GB	7 días	Sin límite
vip	96	512 GB	Sin límite*	Sin límite (+5 GPUs)

Nota: La columna "CPUs Máx." indica el número máximo de hilos de procesamiento (threads) que puede solicitar un trabajo, no procesadores físico

6.4.2 Descripción de cada nivel de QoS

- **small:** Para tareas pequeñas y pruebas. Permite muchos trabajos simultáneos con recursos limitados por trabajo.
- **normal:** Nivel estándar para análisis regulares. Equilibrio entre recursos y concurrencia.
- **interactive:** Esta QoS se asignará la primera vez que soliciten acceso a GPU
- **gpu:** Para cargas de trabajo que requieren procesamiento en GPU intenso.
- **large:** Para análisis que requieren grandes cantidades de memoria y CPUs.
- **vip:** Reservado para proyectos prioritarios aprobados por el Comité Técnico del MCPs. Otorga máximos recursos y prioridad.

* Nota sobre Tiempo Sin Límite: Aunque no existe un límite duro de tiempo de ejecución (Walltime), los trabajos bajo esta QoS están sujetos a monitoreo de actividad. El Administrador

se reserva el derecho de terminar procesos que permanezcan en estado de inactividad (idle), sin consumo de CPU/GPU o bloqueados (zombis) por un periodo continuo superior a 24 horas, previa verificación técnica.

6.4.3 Asignación de QoS a Usuarios

Artículo 6. El nivel de QoS se asigna a cada usuario individualmente según:

- a) La naturaleza y requerimientos del proyecto de investigación.
- b) La justificación técnica presentada.
- c) La disponibilidad de recursos del clúster.
- d) El historial de uso responsable del usuario.

6.5 Restricciones de Almacenamiento

Artículo 7. **No se permite almacenar información que no corresponda con el proyecto** en los espacios de almacenamiento asignados. El Administrador se reserva el derecho de eliminar archivos que no cumplan con esta política, previa notificación al usuario.

Artículo 8. Todos los usuarios tendrán 500GB de almacenamiento disponible.

6.6 Solicitud de Cambio de QoS y almacenamiento

Artículo 9. Los usuarios podrán solicitar un cambio de nivel de QoS y aumento de capacidad de almacenamiento mediante comunicación formal al Administrador del Clúster MCPS-HPC, indicando:

- Justificación técnica del requerimiento adicional de recursos.
- Periodo estimado de necesidad del QoS solicitado.
- Descripción del análisis o proceso que requiere recursos adicionales.

6.7 Notificación de Irregularidad de Cancelación Programada

Artículo 10. El Administrador del Clúster MCPS-HPC monitoreará continuamente la ejecución de trabajos. En caso de detectar procesos que incumplan las políticas, afecten el rendimiento o excedan recursos, se procederá bajo el siguiente esquema de Aviso y Cancelación:

- Se enviará un aviso inmediato al usuario (Medios de contacto establecidos) informando que su trabajo (Job ID) ha sido marcado para cancelación por incumplimiento de políticas.
- El trabajo permanecerá en ejecución condicional por un periodo de 60 minutos a partir de la notificación.
- Si al término del periodo el usuario no ha presentado una justificación válida o no ha tomado acciones correctivas, el trabajo será cancelado definitivamente por el sistema.

6.8 Apelación Inmediata y Resolución

Artículo 11. El usuario podrá detener el proceso de cancelación programada siguiendo estos pasos durante el periodo después de ser notificado:

- Deberá responder al aviso de notificación indicando el script en ejecución.
- Deberá presentar una justificación técnica breve que sustente por qué el comportamiento del trabajo es necesario y legítimo para su investigación.
- El Administrador evaluará la apelación de inmediato. Si la justificación es procedente, se retirará la marca de cancelación y el trabajo continuará su ejecución normal. En caso contrario, se confirmará la cancelación.

CAPÍTULO VII. PROTECCIÓN DE DATOS

(Control A.18 ISO 27001 - Cumplimiento)

7.1 Datos de la Organización

Artículo 1. Los datos genéticos y epidemiológicos almacenados en el Clúster MCPS-HPC son propiedad del proyecto MCPS y están sujetos a la legislación mexicana en materia de protección de datos personales y datos sensibles.

7.2 Prohibición de Extracción

Artículo 2. **Prohibición de extracción de datos de la organización:** Queda prohibida la extracción de cualquier dato genético, epidemiológico o información sensible perteneciente a la Organización MCPS fuera del Clúster MCPS-HPC, por cualquier medio o formato.

Esta prohibición no aplica a los resultados de análisis, siempre que dichos resultados cumplan con lo establecido en el Capítulo III y no contengan datos crudos ni información identificable.

La extracción indebida de datos de la organización será considerada una **falta crítica** y sancionada conforme al Capítulo XI

CAPÍTULO VIII. RESPALDO DE INFORMACIÓN

8.1 Alcance y Responsabilidad del Respaldo

Artículo 1. El usuario es el único responsable de realizar copias de seguridad (respaldos) periódicas de sus archivos de trabajo propios, tales como: códigos fuente, scripts, documentación generada y los resultados de análisis que cumplan con los criterios de descarga establecidos en la sección 3.2.

Artículo 2. Se recomienda que el usuario respalde periódicamente su información según la importancia y frecuencia de modificación de la misma.

Artículo 3. En estricto cumplimiento con el Capítulo VII el usuario queda exento y tiene prohibido realizar respaldos o copias de los Datos de la Organización. La integridad y persistencia de estos datos sensibles son gestionadas exclusivamente por la infraestructura de almacenamiento del Clúster MCPS-HPC y sus políticas internas de redundancia.

8.2 Garantías del Sistema

Artículo 4. El Clúster MCPS-HPC implementa redundancia a nivel de almacenamiento distribuido (Ceph) para protección contra fallos de hardware. Sin embargo, esto no constituye un sistema de respaldo completo y no protege contra eliminación accidental por parte del usuario.

Artículo 5. El Administrador del Clúster MCPS-HPC no se hace responsable por la pérdida de información derivada de:

- Acciones del usuario: eliminación accidental, sobreescritura.
- Fallas catastróficas no previstas.
- Interrupción de procesos por violaciones a las políticas de uso.

8.3 Notificación de Límites

Artículo 6. El Administrador informará oportunamente a los usuarios cuando se esté alcanzando el límite de almacenamiento asignado al proyecto o cuando el espacio común esté próximo a su capacidad máxima.

CAPÍTULO IX. MANTENIMIENTO DEL MCPS-HPC

9.1 Mantenimiento Programado

Artículo 1. El Administrador realizará mantenimientos preventivos periódicos del Cluster MCPS-HPC. Los usuarios serán notificados con un mínimo de 72 horas de anticipación sobre los periodos de mantenimiento programado.

9.2 Mantenimiento de Emergencia

Artículo 2. En caso de mantenimiento de emergencia por fallas críticas o incidentes de seguridad, el Administrador procederá de inmediato y notificará a los usuarios tan pronto como sea posible.

9.3 Responsabilidad del Administrador

Artículo 3. El Administrador del Clúster no se hace responsable por la interrupción de procesos o daño a la información por fallas no atribuibles al sistema o por mantenimientos de emergencia.

CAPÍTULO X. GESTIÓN DE PROYECTOS

10.1 Registro de Proyectos

Artículo 1. El usuario deberá proporcionar:

- Nombre y descripción del proyecto.
- Objetivos de investigación.
- Vigencia estimada.
- Recursos computacionales estimados.
- Lista de colaboradores que requerirán acceso.
- Fuentes de financiamiento (si aplica).

Artículo 2. La inclusión de colaboradores en esta lista tiene fines únicamente de registro y vinculación al proyecto. No implica la creación automática de cuentas.

Cada colaborador listado deberá tramitar su propia solicitud de acceso y obtener sus credenciales individuales conforme al Capítulo II. Queda estrictamente prohibido el uso de "cuentas de proyecto" compartidas entre varios miembros del equipo; el acceso es siempre personal e intransferible.

10.2 Cierre de Proyectos

Artículo 3. Una vez finalizado el periodo de vigencia de un proyecto:

- Los recursos del Clúster MCPS-HPC asignados serán suspendidos.
- Los participantes dispondrán de 30 días naturales para respaldar su información.
- Transcurrido este periodo, las cuentas serán inhabilitadas.
- Después de 30 días adicionales, se dispondrá del espacio de almacenamiento.

CAPÍTULO XI. PROHIBICIONES Y SANCIONES

11.1 Conductas Prohibidas

Artículo 1. Queda estrictamente prohibido:

- Inspeccionar, copiar y almacenar programas de cómputo, software y demás fuentes que violen la Ley Federal del Derecho de Autor.
- Instalar cualquier aplicación de software que atente contra la seguridad del MCPS-HPC.
- Usar los recursos de cómputo con fines comerciales, lucrativos, promoción, propaganda, venta o difusión no autorizados.
- Usar los recursos para propósitos ilegales, criminales o no éticos.
- Ejecutar actividades de criptominería o cualquier proceso no relacionado con investigación.
- Intentar acceder a cuentas o datos de otros usuarios sin autorización.
- Dejar sesiones abiertas sin control alguno.
- Realizar actividades que afecten el rendimiento del sistema o perjudiquen a otros usuarios.
- Extraer datos genéticos de la organización MCPS fuera del clúster.

11.2 Clasificación de Faltas

Artículo 2. Las faltas se clasifican en tres niveles según su gravedad:

Nivel	Ejemplos	Consecuencia Inicial
LEVE	Sesiones abandonadas, uso excesivo de recursos sin justificación, almacenamiento de archivos no relacionados	Amonestación escrita
GRAVE	Compartir credenciales, intentar acceder a cuentas ajenas, instalación de software no autorizado, reincidencia en faltas leves	Suspensión temporal (1-30 días)
CRÍTICA	Extracción de datos genéticos, criptominería, actividades ilegales, daño intencional al sistema, reincidencia en faltas graves	Suspensión inmediata + Reporte al Encargado del MCPS

11.3 Procedimiento de Sanciones

Artículo 3. **El procedimiento de sanciones involucra dos autoridades con funciones claramente definidas:**

11.3.1 Atribuciones del Administrador del Clúster

El Administrador del Clúster (autoridad técnica) tiene las siguientes atribuciones:

- a) **Detección:** Monitorear el uso del sistema e identificar actividades sospechosas o violaciones a las políticas.
- b) **Suspensión inmediata:** Suspender de forma inmediata cualquier cuenta que represente un riesgo para la seguridad del sistema o los datos.
- c) **Reporte:** Reportar al Encargado del MCPS las actividades que considere necesario revisar, especialmente faltas graves y críticas.
- d) **Reactivación:** Reactivar cuentas suspendidas temporalmente una vez concluido el periodo de sanción.

11.3.2 Atribuciones del Encargado del MCPS

Artículo 4. El Mtro. Jesús Alegre Díaz, Encargado del Estudio Prospectivo de la Ciudad de México (autoridad administrativa), tiene las siguientes atribuciones:

- a) **Revisión:** Revisar los reportes de incidentes presentados por el Administrador del Clúster.
- b) **Determinación:** Determinar la gravedad final de la falta y la sanción correspondiente.
- c) **Acciones adicionales:** Decidir sobre acciones administrativas adicionales, incluyendo notificación a instituciones de origen del usuario.

- d) **Acciones legales:** Determinar si procede iniciar acciones legales conforme a la legislación aplicable en materia de protección de datos personales, propiedad intelectual y delitos informáticos.

11.3.3 Flujo del Procedimiento

Artículo 5. El procedimiento de sanciones sigue el siguiente flujo:

1	DETECCIÓN: El Administrador detecta la falta. Solo en casos de riesgo crítico de seguridad podrá suspender el acceso de inmediato. En cualquier otro caso, deberá notificar primero.
2	NOTIFICACIÓN Y RÉPLICA El usuario recibirá un aviso por correo electrónico detallando la falta y la evidencia. Contará con 24 horas hábiles para presentar su justificación o defensa antes de que se aplique la sanción
3	RESOLUCIÓN: Para Faltas Leves y Graves: Tras evaluar la respuesta del usuario, el Administrador podrá aplicar directamente la amonestación o la suspensión breve. Para Faltas Críticas: El Administrador remitirá la evidencia y la respuesta del usuario al Comité Técnico, quien tomará la decisión final para evitar conflictos de interés.
4	NOTIFICACIÓN DE RESOLUCIÓN: La resolución definitiva se notificará al usuario por correo electrónico institucional en un plazo máximo de 48 horas.
5	ACCIONES ADICIONALES: El Administrador recopila evidencia técnica del incidente (logs, registros, capturas).
4	NOTIFICACIÓN: Para faltas críticas, el Encargado del MCPS evaluará si proceden acciones administrativas o legales conforme a la legislación aplicable.

11.4 Sanciones Aplicables

Artículo 6. Las sanciones aplicables según la gravedad de la falta son:

- **Amonestación escrita:** Primera falta leve. Se registra en el expediente del usuario.
- **Suspensión temporal (7-30 días):** Faltas graves o reincidencia en faltas leves.
- **Inhabilitación definitiva:** Faltas críticas.
- **Acciones legales:** Para faltas críticas, el Encargado del MCPS podrá determinar el inicio de acciones legales conforme a la legislación aplicable.

CAPÍTULO XII. DISPOSICIONES FINALES

12.1 Modificación de Políticas

Artículo 1. El Administrador del Clúster MCPS-HPC notificará a los usuarios las modificaciones de estas políticas mediante los medios de contacto.

Artículo 2. Si un usuario no está conforme con estas políticas y considera necesario modificarlas, podrá ejercer su derecho a solicitar cambios, proporcionando razones válidas y debidamente justificadas. Dichas solicitudes serán analizadas y discutidas por Comité Técnico del Clúster MCPS-HPC. Se enviará una respuesta en un periodo de 7 días hábiles

12.2 Casos No Previstos

Artículo 3. Los casos no previstos en estas políticas serán resueltos por el Administrador del Clúster en primera instancia y, de ser necesario, por el Encargado del MCPS, conforme al espíritu de estas disposiciones y a la normatividad aplicable.

12.3 Aceptación de Políticas

Artículo 4. El uso del Clúster MCPS-HPC implica la aceptación total de estas políticas. El usuario declara haber leído, comprendido y aceptado las condiciones aquí establecidas.

12.4 Vigencia

Artículo 5. Estas políticas entrarán en vigor a partir de su fecha de emisión y permanecerán vigentes hasta su actualización o derogación.

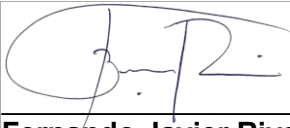
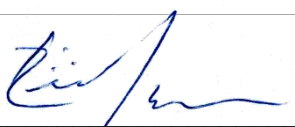
12.5 Medios de Contacto Establecidos

Artículo 6. En el caso del contacto por medio de la página web de soporte tendrá que solicitar su usuario y contraseña al administrador por medio del correo electrónico de soporte, antes de usarlo.

ANEXO A DE LAS POLÍTICAS DE ACCESO Y USO DEL CLÚSTER DE ALTO RENDIMIENTO: CARTA COMPROMISO

Yo, _____, en mi calidad de usuario del Clúster MCPS-HPC de la Unidad de Medicina Experimental de la Facultad de Medicina de la UNAM:

- Declaro haber leído y comprendido las Políticas de Acceso y Uso del MCPS-HPC.
- Me comprometo a cumplir con todas las disposiciones establecidas en el presente documento.
- Entiendo que el incumplimiento de estas políticas puede resultar en la suspensión temporal o definitiva de mi acceso al clúster, así como en acciones legales si el Encargado del MCPS lo determina procedente.
- Me comprometo a no extraer, copiar o transmitir datos de la organización fuera del entorno del clúster.
- Me comprometo a reportar cualquier incidente de seguridad o violación a las políticas que detecte.
- Acepto la responsabilidad sobre todas las actividades realizadas desde mi cuenta de usuario.
- Entiendo que para activar mi cuenta, debo enviar un correo electrónico a soporte@mcps-epcm.org, con el asunto "Activación de cuenta" adjuntando esta carta compromiso y el registro del proyecto debidamente llenados y firmados.

	
Nombre y Firma del Usuario del Clúster	Fernando Javier Rivas Reyes Jefe de Bioinformática
	
Mtro. Jesús Alegre Díaz Encargado del Estudio Prospectivo de la Ciudad de México	
	Angel Gael Pacheco Saavedra Administrador del Clúster
Fecha	

Documento alineado con ISO/IEC 27001:2022
Unidad de Medicina Experimental Facultad de Medicina - UNAM

ANEXO B DE LAS POLÍTICAS DE ACCESO Y USO DEL CLÚSTER DE ALTO RENDIMIENTO: GLOSARIO TÉCNICO

Término	Término
CPU (en contexto HPC)	En el contexto del MCPS-HPC, cuando se indica "CPUs" en las asignaciones de QoS, se refiere a hilos de procesamiento (threads), no a procesadores físicos completos. Un procesador físico moderno típicamente contiene múltiples núcleos (cores), y cada núcleo puede ejecutar uno o más hilos simultáneamente. Por ejemplo, "8 CPUs" significa 8 hilos de ejecución disponibles para el trabajo.
GPU	Unidad de Procesamiento Gráfico. Procesador especializado para cálculos paralelos masivos.
RAM	Memoria de Acceso Aleatorio. Memoria volátil para almacenamiento temporal durante la ejecución.
QoS	Quality of Service. Políticas que definen límites de recursos por usuario o trabajo.
Job/Trabajo	Tarea computacional enviada al sistema de colas para su ejecución.
Slurm	Sistema de gestión de colas y recursos del clúster.
Partición	Agrupación lógica de nodos con características o políticas específicas.
SSH	Protocolo de conexión segura para acceso remoto.
Apptainer	Sistema de contenedores para HPC (anteriormente Singularity).

Anexo B

FORMULARIO DE REGISTRO DE PROYECTO

Clúster de Alto Rendimiento MCPS-HPC

No. de Registro:		Fecha de Solicitud:	
-------------------------	--	----------------------------	--

1. DATOS DEL PROYECTO

Nombre del Proyecto:	
Acrónimo (si aplica):	
Área de Investigación:	☐ Genómica ☐ Bioinformática ☐ Epidemiología ☐ Machine Learning ☐ Otro: _____

Descripción del Proyecto:

Objetivos del Proyecto:

Fecha de Inicio:		Fecha de Término:	
-------------------------	--	--------------------------	--

2. USUARIO DEL CLÚSTER MCPS-HPC

Nombre Completo:	
Institución:	
Departamento/Área:	
Cargo/Puesto:	
Correo Electrónico:	

3. COLABORADORES DEL PROYECTO

Liste a todos los colaboradores que requieren acceso al proyecto. Cada colaborador deberá firmar la Carta Compromiso de Políticas de Uso.

No.	Nombre Completo	Correo Electrónico	Institución	Rol en Proyecto
1				
2				
3				

* Si requiere más espacios, adjunte una hoja adicional con el mismo formato.

4. RECURSOS COMPUTACIONALES ESTIMADOS

Nivel de Recursos Solicitado:	☐ small ☐ normal ☐ interactive ☐ gpu ☐ large
-------------------------------	---

Referencia de niveles de recursos:

QoS	CPUs Máx.	Memoria Máx.	Tiempo Máx.	Jobs Simultáneos
small	8	32 GB	Sin límite	20
normal	24	150 GB	2 días	10
interactive	16	64 GB	12 horas	2 (+1 GPU)
gpu	48	128 GB	3 días	5 (+2 GPUs)
large	96	256 GB	7 días	Sin límite

Número de Jobs Estimados (mensual):	
¿Requiere acceso a GPUs?	☐ Sí ☐ No

5. DATOS Y SOFTWARE

¿Utilizará datos genéticos del MCPS?	☐ Sí ☐ No
--------------------------------------	---

Tipo de Datos a Utilizar:

☐ Genómicos ☐ Metabolómicos ☐ Secuenciación ☐ CNVs ☐ Datos fenotípicos ☐ Datos de Cuestionario / Sociodemográficos ☐ Otro: _____

Software Requerido (especifique si necesita software adicional al preinstalado):

--

6. DECLARACIONES Y COMPROMISOS

El Responsable del Proyecto declara y se compromete a:

- ☐ He leído y acepto las Políticas de Acceso y Uso del Clúster MCPS-HPC (POL-MCPS-HPC-001 v2.1)
- ☐ Notificar al Administrador sobre cambios en el equipo de colaboradores
- ☐ Garantizar que los datos genéticos y Epidemiológicos del MCPS NO serán extraídos del clúster
- ☐ Reportar inmediatamente cualquier incidente de seguridad detectado

7. FIRMAS

USUARIO DEL CLUSTER MCPS-HPC	
Firma	Nombre: _____ Fecha: _____
ADMINISTRADOR DEL CLÚSTER	
Firma	Proyecto Aprobado: ☐ Sí ☐ No Nivel Asignado: _____ Fecha: _____

Anexo C

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
Unidad de Medicina Experimental de la Facultad de Medicina UNAM

Manual de Usuario

Configuración de Llave SSH y VPN

MCPS

Mexico City Prospective Study

Administrador del cluster MCPS-HPC

Angel Gael Pacheco Saavedra

Jefe de bioinformática

Fernando Javier Rivas Reyes

11 de febrero de 2026

Configuración de Llave SSH y VPN

Guía para usuarios del Clúster MCPS-HPC

Este documento describe el procedimiento para configurar WireGuard, llave SSH privada y conectarse al clúster MCPS-HPC. Siga las instrucciones correspondientes a su sistema operativo.

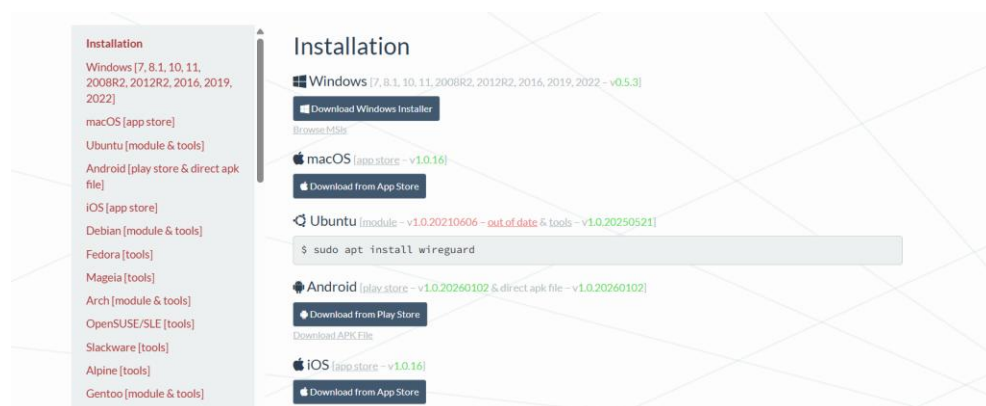
Configuración WireGuard

Abre el siguiente enlace oficial:

<https://www.wireguard.com/install/>

Selecciona la versión correspondiente a tu sistema operativo (Windows, macOS o Linux).

Una vez descargada, instala la aplicación siguiendo las instrucciones del instalador.

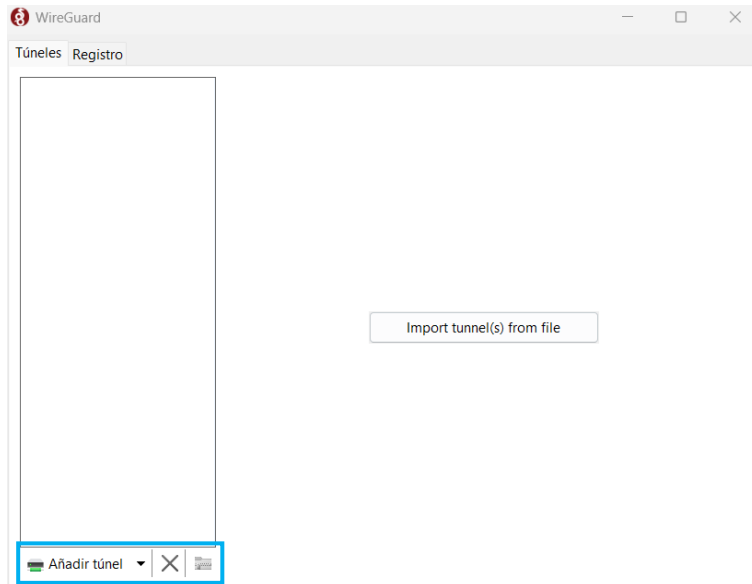


Importar el archivo de configuración (.conf)

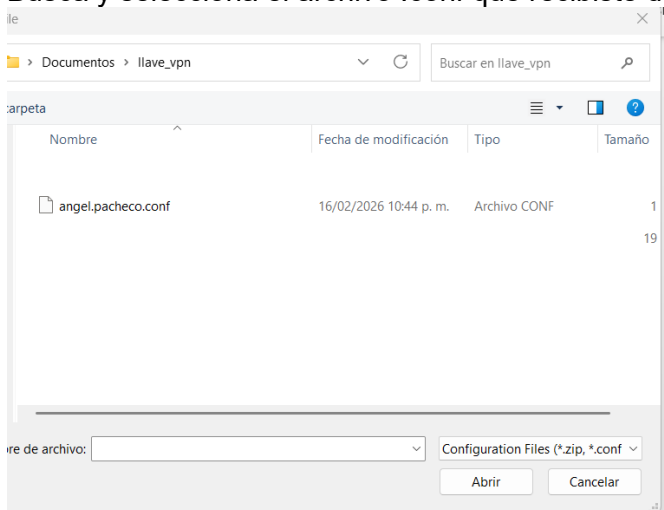
Abre la aplicación WireGuard.

Haz clic en “Add tunnel” / “Añadir túnel”.

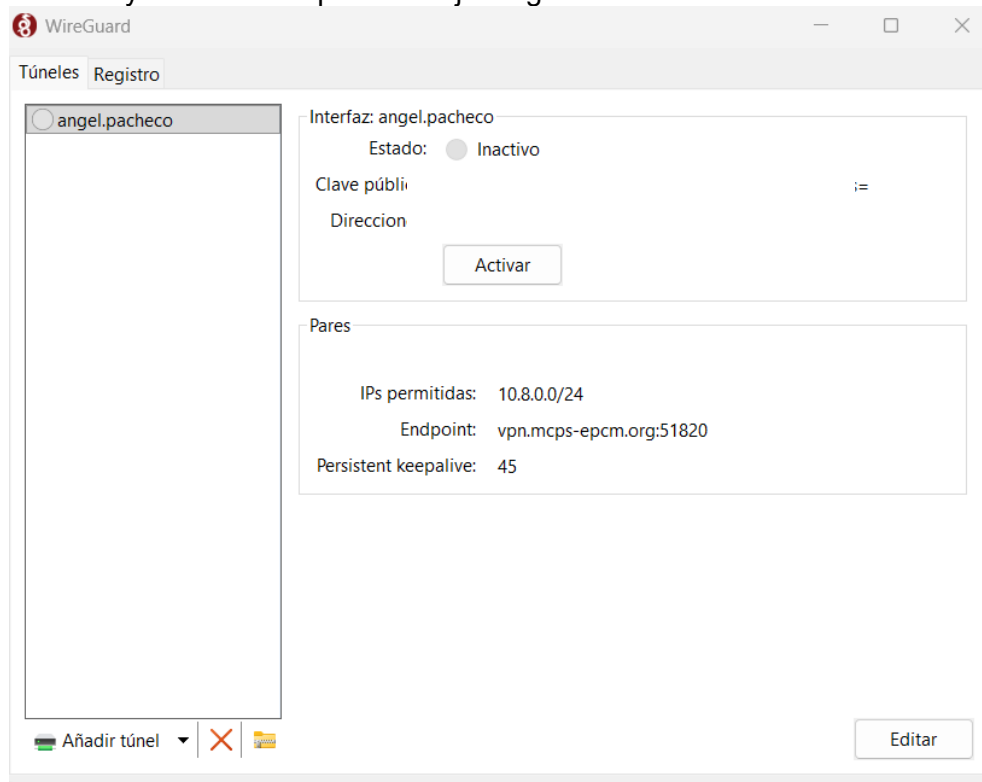
Esto abrirá el explorador de archivos.



Busca y selecciona el archivo .conf que recibiste del soporte técnico.



Una vez cargado, la aplicación mostrará los parámetros de la VPN. Esta información es sensible y no debe compartirse bajo ninguna circunstancia.



En la ventana principal de WireGuard, activa el switch de encendido del túnel. Una vez encendido, tu equipo estará conectado a la red interna del clúster MCPS-HPC y podrás continuar con los pasos de acceso por SSH.

Inicio con contraseña temporal por ssh

Deberá de iniciar sesión por primera vez y cambiar la contraseña con las características establecidas en las Políticas de acceso y uso del clúster de alto rendimiento.

```
ssh USUARIO@10.8.0.18
```

Contraseña archivo .zip para llave ssh y vpn

La contraseña del archivo ZIP se obtiene usando tu nombre y apellido correspondientes a tu usuario (nombre.apellido). Siguiendo la expresión !NNn#aAAxy:

1. Primeras 2 letras de tu nombre en MAYÚSCULAS
2. Última letra de tu nombre en minúscula
3. Símbolo ! al inicio
4. Símbolo # como separador
5. Primera letra de tu apellido en minúscula
6. Últimas 2 letras de tu apellido en MAYÚSCULAS
7. Número de letras del nombre y del apellido al final

Windows (PowerShell)

1. **Guardar el archivo recibido como:**

```
C:\Users\SU_USUARIO\.ssh\id_ed25519
```

⚠ *Si la carpeta .ssh no existe, créela manualmente.*

2. **Ajustar permisos (MUY IMPORTANTE):**

```
icacls $env:USERPROFILE\.ssh\id_ed25519 /inheritance:r
```

```
icacls $env:USERPROFILE\.ssh\id_ed25519 /grant:r "$env:USERNAME:F"
```

3. **Conectarse al clúster:**

```
ssh USUARIO@10.8.0.18
```

(Reemplace USUARIO con su nombre de usuario asignado)

macOS / Linux

1. **Guardar el archivo como:**

```
~/.ssh/id_ed25519
```

2. **Configurar permisos:**

```
chmod 600 ~/.ssh/id_ed25519
```

3. **Conectarse al clúster:**

```
ssh USUARIO@132.248.169.96
```

(Reemplace USUARIO con su nombre de usuario asignado)

Reglas para su Llave SSH

Su llave SSH es como la llave de su casa. Protégjala con el mismo cuidado.

No compartir la llave con nadie
No renombrar el archivo
No subirla a la nube (Google Drive, Dropbox, OneDrive, etc.)
No enviarla por WhatsApp, Telegram, correo u otra mensajería
Guardarla como si fuera su contraseña más importante
Al terminal el proceso con éxito enviar un correo de confirmación a soporte

Solución de Problemas Comunes

Error: "Permission denied (publickey)"

- Verifique que el archivo esté en la ubicación correcta
- Verifique que los permisos estén configurados correctamente
- Asegúrese de usar el nombre de usuario correcto
-

Error: "Permissions are too open"

Este error indica que los permisos del archivo son demasiado permisivos. Ejecute nuevamente los comandos de permisos según su sistema operativo.

La carpeta .ssh no existe

Windows:

```
mkdir %env:USERPROFILE%.ssh
```

macOS / Linux:

```
mkdir -p ~/.ssh
```

Soporte Técnico

Si tiene problemas para conectarse después de seguir estas instrucciones, contacte al administrador del clúster:

Correo: sosporte@mcps-epcm.org

Incluya en su mensaje: su sistema operativo y el error exacto que recibe.

Anexo D

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
Unidad de Medicina Experimental de la Facultad de Medicina UNAM

Manual de Usuario

Configuración del correo institucional @mcps-epcm.org

MCPS

Mexico City Prospective Study

Administrador del cluster MCPS-HPC

Angel Gael Pacheco Saavedra

Jefe de bioinformática

Fernando Javier Rivas Reyes

11 de febrero de 2026

Objetivo

Este manual describe los pasos necesarios para **agregar y sincronizar un correo institucional con dominio @mcps-epcm.org dentro de una cuenta de Gmail**, utilizando el protocolo POP3.

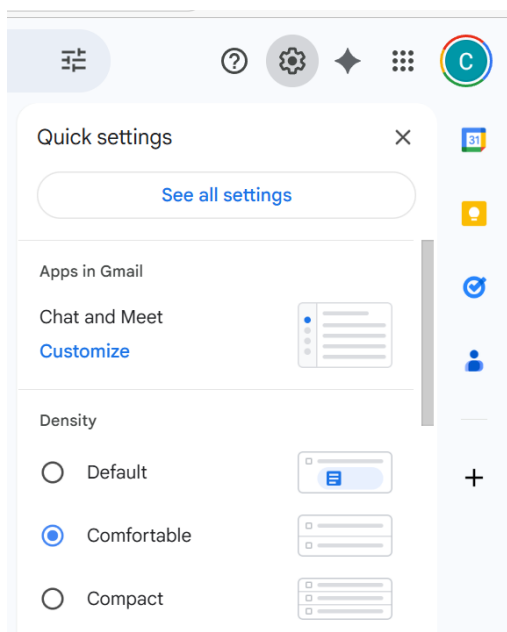
Requisitos

Antes de iniciar, asegúrate de contar con:

- Acceso a una cuenta de Gmail.
- Usuario y contraseña del correo con dominio **@mcps-epcm.org**.
- Conectividad a Internet.
- Configuración POP3 habilitada en el servidor institucional (administración MCPS-HPC).

Pasos para enlazar el correo en Gmail

- 1 Inicia sesión en tu cuenta de Gmail.
- 2 En la parte superior derecha, haz clic en **el ícono de engrane** (⚙).
- 3 Elige **"Ver toda la configuración"** / **See all settings**.



Dentro del menú superior, selecciona **"Accounts and import"**.
En el apartado **"Check mail from other accounts"**, haz clic en:
Add a mail account

Settings

General

Labels

Inbox

Accounts and Import

Filters and Blocked Addresses

Forwarding and POP/IMAP

Add-ons

Chat and Meet

Advanced

Offline

Themes

no cost

Import mail and contacts:

Learn more

Import from Yahoo!, Hotmail, AOL, or other webmail or POP3 accounts.
[Import mail and contacts](#)

Send mail as:

(Use Gmail to send from your other email addresses)
[Learn more](#)

Carlos Lira <clira5097@gmail.com>
[Add another email address](#)

[edit info](#)

Check mail from other accounts:

[Learn more](#)

[Add a mail account](#)

Grant access to your account:

(Allow others to read and send mail on your behalf)
[Learn more](#)

[Add another account](#)

Mark as read

☒ Mark conversation as read when opened by others

☐ Leave conversation unread when opened by others

En la ventana emergente, ingresa tu dirección:
username@mcps-epcm.org
Haz clic en **Next**.

Gmail - Add a mail account - Personal: Microsoft Edge

https://mail.google.com/mail/u/0/?ui=2&ik=002661aa57&jsver=BFOqm4uv...

Add a mail account

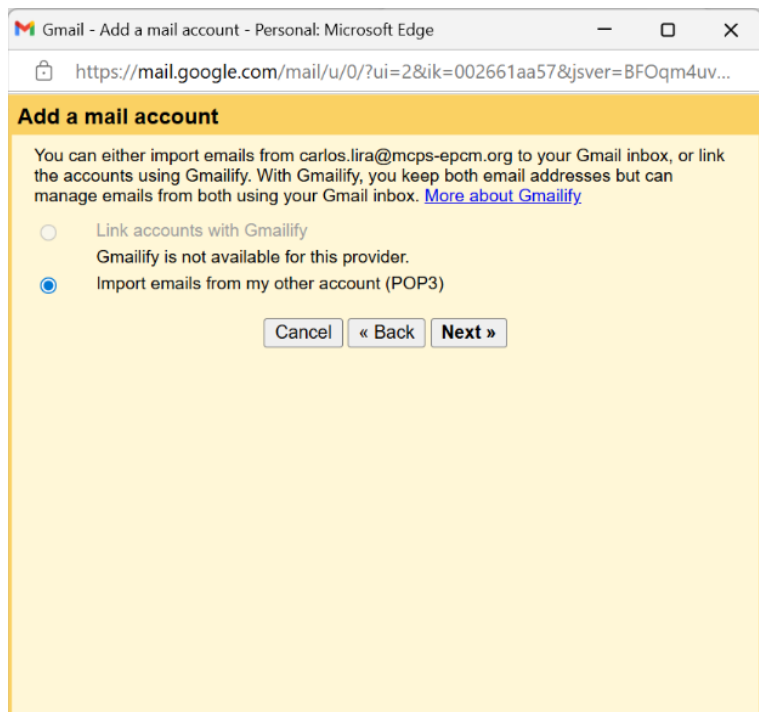
Enter the email address you would like to add.

Email address:

Cancel

Next »

Import emails from my other account (POP3)
Luego haz clic en Next.



En el formulario, llena los campos:

Campo	Valor
Email address	username@mcps-epcm.org
Username	username@mcps-epcm.org
Password	(tu contraseña institucional)
POP Server	mail.mcps-epcm.org
Port	995 (normal para POP3 con SSL)
Always use a secure connection (SSL)	Activado
Leave a copy of retrieved messages on the server	Opcional
Label incoming messages	Opcional
Archive incoming messages	Desactivado

Mail - Add a mail account - Personal, Microsoft Edge

https://mail.google.com/mail/u/0/?ui=2&ik=002661aa57&jsver=BFOq...

Add a mail account

Enter the mail settings for **carlos.lira@mcps-epcm.org**. [Learn more](#)

Email address: **carlos.lira@mcps-epcm.org**

Username:

Password:

POP Server: Port:

☒ Leave a copy of retrieved message on the server. [Learn more](#)

☒ Always use a secure connection (SSL) when retrieving mail. [Learn more](#)

☐ Label incoming messages:

☐ Archive incoming messages (Skip the Inbox)

Después de agregar, se te preguntará:
¿Deseas enviar correos como username@mcps-epcm.org?
Selecciona Yes y presiona Next.

https://mail.google.com/mail/u/0/?ui=2&ik=002661aa57&jsver=BFOq...

✓ Your mail account has been added.

You can now retrieve mail from this account.
Would you also like to be able to send mail as **carlos.lira@mcps-epcm.org**?

☒ Yes, I want to be able to send mail as **carlos.lira@mcps-epcm.org**.

☐ No (you can change this later)

Add another email address you own

Enter information about your other email address.

(your name and email address will be shown on mail you send)

Name:

Email address: **carlos.lira@mcps-epcm.org**

☒ Treat as an alias. [Learn more](#)

[Specify a different "reply-to" address](#) (optional)

Completa:

- Nombre a mostrar (Nombre y Apellido).
- Servidor SMTP: mail.mcps-epcm.org.
- Usuario: **username@mcps-epcm.org**
- Contraseña institucional.
- Conexión segura TLS/SSL.

Gmail enviará un código de confirmación al correo institucional; debes copiarlo para validar.

Add another email address you own

Send mail through your SMTP server

Configure your mail to be sent through mcps-epcm.org SMTP servers [Learn more](#)

SMTP Server:

Port:

Username:

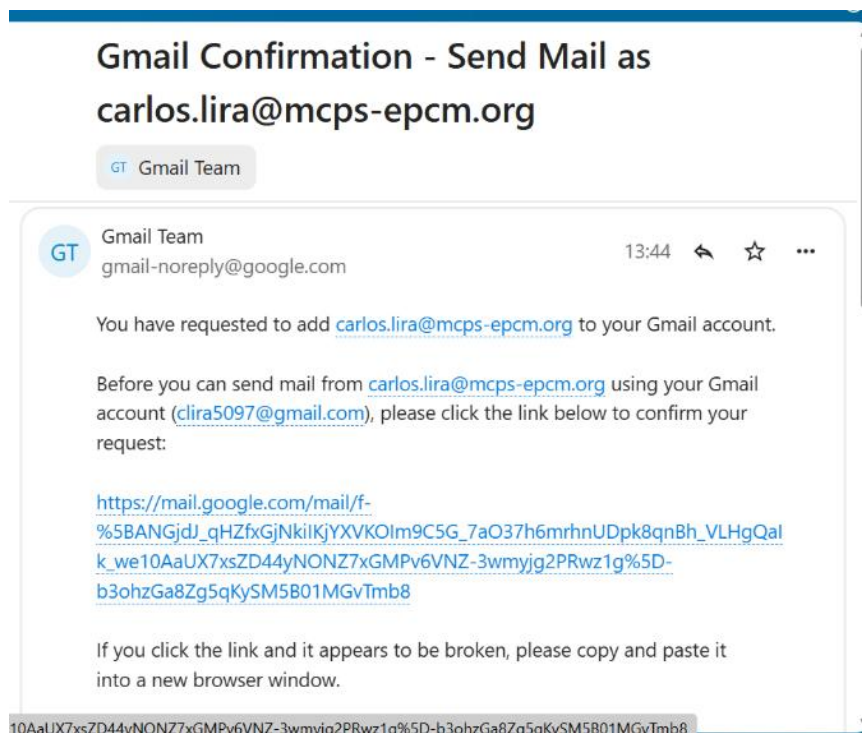
Password:

☒ Secured connection using [TLS](#) (recommended)

☐ Secured connection using [SSL](#)

Una vez concluido el proceso:

- Gmail deberá comenzar a **importar los correos POP3**.
- En la bandeja lateral, aparecerá una etiqueta con el nombre del correo vinculado.
- Podrás **enviar y recibir correos** directamente desde Gmail utilizando la identidad MCPS.

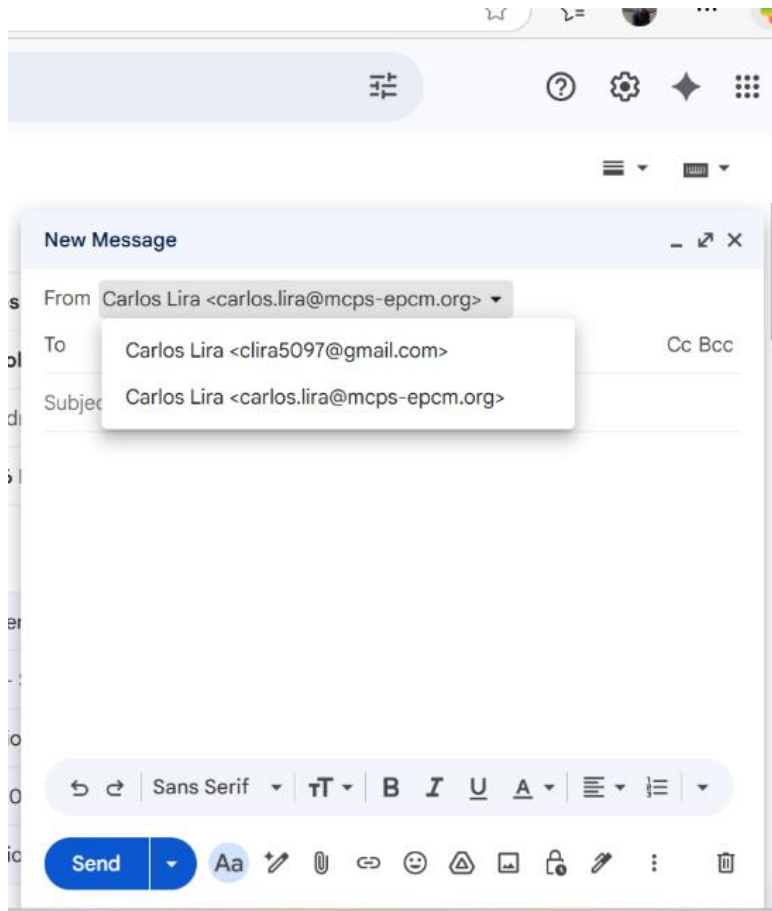


Confirmation

Please confirm sending mail as **carlos.lira@mcps-epcm.org**.

Click here to return to your Gmail account: <https://mail.google.com>.

Confirm

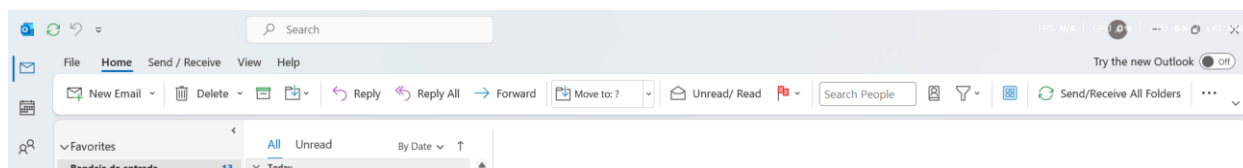


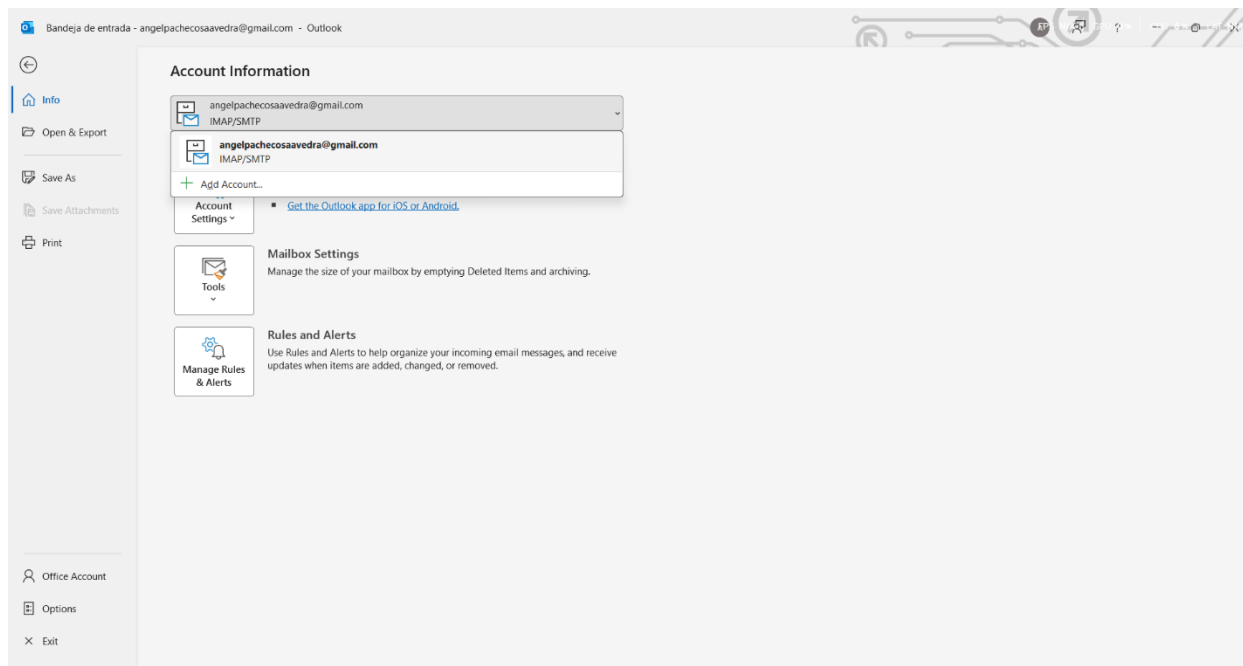
Pasos para enlazar el correo en Outlook

Abre **Outlook**.

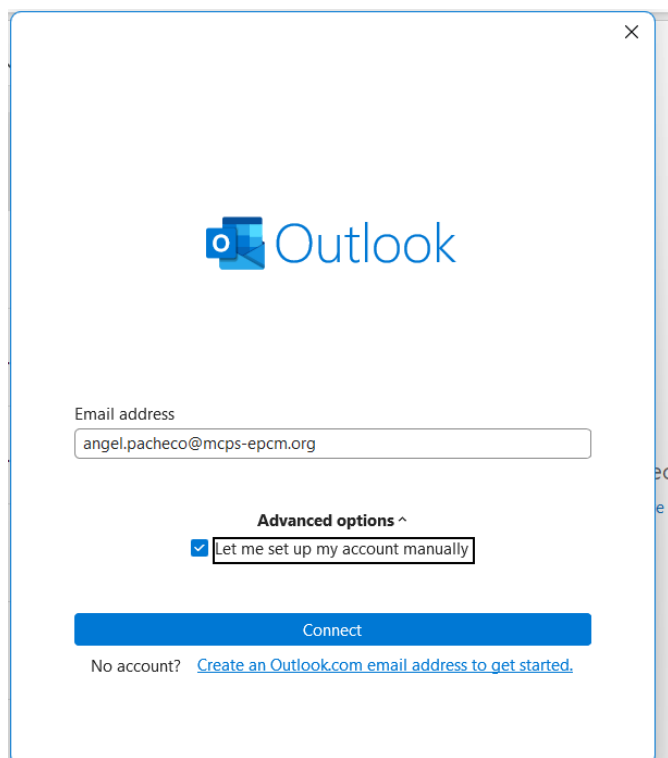
Ve al menú superior y selecciona **File / Archivo**.

Haz clic en **Add Account / Agregar cuenta**.





En la ventana que aparece, selecciona **Advanced options**.
Marca la casilla **Let me set up my account manually** / “Permitir configurar manualmente”.
Presiona **Connect / Conectar**.
Selecciona el tipo de cuenta: **IMAP**.



En los campos solicitados, ingresa lo siguiente:

Campo	Valor
Email address	username@mcps-epcm.org
Server	mail.mcps-epcm.org
Port Incoming	993
Encryption Incoming	SSL/TLS
Port Outgoing	587
Encryption Outgoing	STARTTLS
Username	username@mcps-epcm.org
Password	(tu contraseña institucional)

×

IMAP Account Settings

angel.pacheco@mcps-epcm.org

(Not you?)

Incoming mail

Server

mail.mcps-epcm.org

Port

993

Encryption method

SSL/TLS

☐ Require logon using Secure Password Authentication (SPA)

Outgoing mail

Server

mail.mcps-epcm.org

Port

587

Encryption method

STARTTLS

☐ Require logon using Secure Password Authentication (SPA)

[Go back](#)

Next

Haz clic en **Next / Siguiente**.

Outlook intentará iniciar sesión y enviar un correo de prueba.

Si la validación es correcta, presiona **Finish / Finaliza**

