DESARROLLO DE UNA INTERFAZ WEB PARA EL ENVÍO DE TRABAJOS A SLURM USANDO CGI C++

FABIÁN ANDRÉS LEÓN PÉREZ

UNIVERSIDAD INDUSTRIAL DE SANTANDER

FACULTAD DE INGENIERÍAS FISICOMECÁNICAS

ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

BUCARAMANGA

2019

DESARROLLO DE UNA INTERFAZ WEB PARA EL ENVÍO DE TRABAJOS A SLURM USANDO CGI C++

FABIÁN ANDRÉS LEÓN PÉREZ

Una tesis presentada en cumplimiento de los requisitos para el grado de Ingeniero de Sistemas e Informática

Director:

Ms. Gilberto Javier Diaz Toro

Profesor Escuela de Ingeniería de Sistemas e informática

UNIVERSIDAD INDUSTRIAL DE SANTANDER

FACULTAD DE INGENIERÍAS FISICOMECÁNICAS

ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

BUCARAMANGA

2019

AGRADECIMIENTOS

El autor expresa su agradecimiento:

Al grupo de Investigación de Cómputo Avanzado y a Gran Escala (CAGE) y a mi director por su guía, paciencia y orientación, durante el desarrollo de este libro.

Gracias a mis Padres Oliva Pérez Pérez y Henry León Lizcano por su amor incondicional y esfuerzo que dedicaron en mí durante todos estos años, por sus consejos, su compañía y todos los valores y principios que me enseñaron a lo largo de mi vida.

Gracias a mis hermanos Jhon Henry León Pérez y Diana Marcela Rodríguez Pérez por su compañía y apoyo.

Gracias a Tatiana Carolina Gélvez Barrera por su compañía, apoyo, orientación y paciencia durante estos últimos semestres de mi carrera universitaria.

Gracias a Virginia Duarte Quintero por todo sus cuidados, consejos, valores, principios y ejemplo de ser humano que me acompañará a lo largo de mi vida.

A todos mis compañeros de universidad que de una u otra manera me han permitido construir el ser humano que soy, por su fiel compañía y sincera amistad.

CONTENIDO

LLISTA DE FIGURAS

LISTA DE TABLAS

RESUMEN

TÍTULO: DESARROLLO DE UNA INTERFAZ WEB PARA EL ENVÍO DE TRABAJOS A SLURM USANDO CGI C++*

AUTOR: FABIÁN ANDRÉS LEÓN PÉREZ**

PALABRAS CLAVE: SLURM, Cluster Linux, CGI, C++

DESCRIPCIÓN:

Protocolos como Shell seguro han sido utilizados comúnmente por los clusters de Linux para permitir a los usuarios enviar trabajos a SLURM. Sin embargo, implica el uso de un emulador de cónsola para establecer la comunicación remota que, en algunos casos, no está disponible. Además de esto implica también que los usuarios usen comandos y editores de texto basados en cónsola que para áreas no relacionadas con computación puede ser algo desafiante y tedioso. Por lo tanto, este documento presenta el desarrollo de la API Web Submit SLURM, que ofrece una interfaz web rápida y segura para enviar trabajos a SLURM usando un formulario web sencillo o el envío de trabajos usando un scrip shell. La API también cuenta con un diseño adaptativo que permite que sea compatible con la mayoría de los navegadores y dispositivos independientemente del tamaño de la pantalla. La implementación se realizó haciendo que la API sea escalable tanto en el tamaño del cluster y versiones y tipo del software necesario haciendo una instalación simple y altamente compatible. Los tiempos de ejecución usando el protocolo SSH y la API propuesta fueron calculados en el envío de 10 segundos trabajos mostrando una diferencia de 0.013 lo cual no degrada la experiencia final del usuario y mostrando la eficacia de la API propuesta.

* Trabajo de grado.

** Facultad de Ingenierías Fisicomecánicas. Escuela de Ingeniería de Sistemas e Informática. Director: Gilberto Javier Diaz Toro.

ABSTRACT

TITLE: DEVELOPMENT OF A WEB INTERFACE FOR SUBMITTING JOBS TO SLURM USING CGI C++*

AUTHOR: FABIÁN ANDRÉS LEÓN PÉREZ**

KEYWORDS: SLURM, Linux Cluster,CGI, C++.

DESCRIPTION:

Protocols such as Secure Shell have been commonly used by Linux clusters to allow users sending the jobs to SLURM. However, it implies the use of a console emulator to establish the remote communication which, in some cases, is not available.

in the same way implies that users use commands and text editors based on console that for areas not related to computing can be challenging and tedious. Therefore, this document presents the development of the API (Application programming interface.) Web Submit SLURM (Simple Linux Utility for Resource Management), which offers a fast and secure web interface for submitting jobs to SLURM using a simple web form or sending jobs using a scrip shell. The API also has a responsive design that allows it to be compatible with most browsers and devices regardless of screen size. The implementation is done by making the API scalable in cluster size and versions and type of software needed by making a simple and highly compatible installation. The execution times using the SSH protocol and the proposed API were calculated in the submission of 10 jobs showing a difference of 0.013 seconds which does not degrade the final user experience and showing the effectiveness of the proposed API.

* Bachelor tesis.

** Faculty of Physicomechanical Engineering. School of Systems and Computer Engineering. Director: Gilberto Javier Diaz Toro.

INTRODUCCIÓN

Computación de alto Rendimiento (HPC en inglés) hace referencia a un campo de la computación que da solución a problemas de cómputo muy complejos que normalmente involucran un gran número de datos y costo computacional. Para ello, se suele apoyar en tecnologías como supercomputadores, clusters, computación paralela, entre otras que juntas constituyen una plataforma confiable que permite la ejecución e integración de las diferentes herramientas necesarias para el HPC.

En la actualidad, la arquitectura más usada para el HPC, según el TOP 500 de junio del 2019 [1] con un 88.4%, son los clusters de cómputo y con un 11.6% MPP, como se observa en la figura 1 mostrando una gran predominancia por este primero que se puede definir como un conjunto de equipos de procesamiento matemático conectados a través de una red de alta velocidad de tal manera que comparten procesamiento, almacenamiento y memoria, que trabajan como si fuera uno solo. Típicamente estos clusters de cómputo son compartidos por varios usuarios que ejecutan programas independientes sin interferir en la ejecución de cada uno, esto se logra usando gestores de recursos que permiten compartir memoria, almacenamiento y procesamiento con varios usuarios.

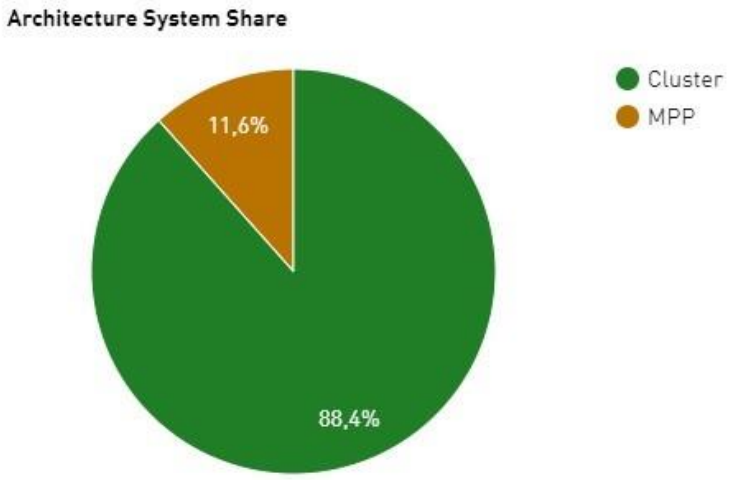
Simple Linux Utility for Resource Managment (abreviado SLURM) es un sistema para la programación de tareas y gestión de clusters de código abierto, el cual, ofrece tres funciones principales [2]. Primero, asigna acceso exclusivo y/o no

[1] STROHMAIER, Erich; DONGARRA, Jack; SIMON, Horst, et al. Top 500 OPERATING SYSTEM FAMILY. [en línea]. 2019. Disponible en https://www.top500.org/statistics/details/osfam/1

[2] YOO, A; JETTE, Morris; GRONDONA, Mark Simple linux utility for resource management. In *Workshop on Job Scheduling Strategies for Parallel Processing* Springer, 2003. pp. 44–60.

exclusivo a los recursos a los usuarios en el conjunto de nodos designados. En segundo lugar, proporciona un marco para iniciar, ejecutar y supervisar los trabajos enviados por los usuarios. Finalmente, modera la conexión a los recursos disponibles al administrar una cola de trabajos pendientes. El sistema SLURM ha sido ampliamente utilizado como programador y administrador de trabajos por varios clusteres Linux. Para nombrar algunos: el supercomputador tianhe-2, quien fue el segundo computador más rápido del mundo según el top500 de noviembre de 2017[3] ubicado en National Supercom-puter Center en Guangzhou, China; BlueGene/L y Purple del Lawrence Livermore National Laboratory se ubicaron en el puesto número 1 y 3, respectivamente en el top500 de noviembre de 2005[4]; Guane, el supercomputador de la Universidad Industrial de Santander a cargo del grupo de Súper Com-

Figura 1. Supercomputadores top 500 donde el 88.4% usan arquitectura de cluster



[3] STROHMAIER, Erich; DONGARRA, Jack; SIMON, Horst, et al. Top 500 november 2017. [en línea]. 2017. Disponible en https://www.top500.org/lists/2017/11/.

[4] STROHMAIER, Erich; DONGARRA, Jack; SIMON, Horst, et al. Top 500 november 2005. [en línea]. 2005. Disponible en https://www.top500.org/lists/2005/11/.

putación y Cálculo Científico UIS (SC3) de Bucaramanga, Colombia [5]. La función principal de estos clusteres es compartir sus recursos con diferentes usuarios para ejecutar sus trabajos. Por lo tanto, estos usuarios necesitan acceder a la línea de comandos en el nodo de administración que ejecuta el demonio de administración para enviar sus trabajos y visualizar la carga computacional del cluster y poder determinar que recursos están libres para ser utilizados. Para acceder a esta línea de comandos de forma remota los usuarios deben usar protocolos de acceso remoto como el protocolo Shell seguro (SSH, del inglés secure shell protocol)[6], el cual proporciona una forma segura de acceder e iniciar sesión en el nodo de administración. Sin embargo, esto implica que el usuario necesite un emulador de consola y algunas veces el uso de una computadora para acceder al nodo de administración y trabajar en él.

Para superar este problema este proyecto desarrolló una aplicación Web para el envío y visualización de trabajos a SLURM, la cual, provee al usuario una herramienta para enviar, crear scripts y visualizar la cola de trabajos a través de una interfaz Web simple. Para esto, se deberá establecer una comunicación vía Web con el servidor de autenticación. Particularmente, en este proyecto se plantea usar el protocolo ligero de acceso a directorios (LDAP, del inglés Lightweight Directory Access Protocol) como servidor de autenticación [7] para validar los datos del usuario y conectarse con SLURM para el envío de trabajos. De esta manera, el usuario solo necesitará autenticarse, llenar un formulario o enviar un script Shell para solicitar recursos y enviar su trabajo, o simplemente iniciar sesión para ver la cola de

[5] BARRIOS, Carlos. Cluster guane. [en línea]. 2019. Disponible en http://wiki.sc3.uis.edu.co/index.php/Cluster%2BGuane.

[6] SCHONWALDER, Jurgen, et al. Session resumption for the secure shell protocol. In *2009 IFIP/IEEE International Symposium on Integrated Network Management* IEEE, 2009, pp. 157–163.

[7] ANDJARWIRAWAN, Justinus; PALIT, Henry Novianus; SALIM, Julio Christian. Linux pam to ldap authentication migration. In *2017 International Conference on Soft Computing, Intelligent System and Information Technology (ICSIIT)* IEEE, 2017, pp. 155–159.

trabajos. El único requerimiento que se plantea es que el usuario tenga una cuenta para autenticarse y un directorio en el nodo de administración para guardar los script Shell e identificarse de manera única.

Diferentes APIs proveen ayuda tanto para administradores y usuarios en el uso de SLURM. Estas APIs han sido implementadas en diferentes lenguajes de programación como pyslurm con Python [8], Heron con java [9] u Open On Demand con módulos Node.js [10], Slurm-web[11] el cual hace uso de pyslurm junto con HTML y Java script. Este proyecto uso PHP, HTML y una interfaz simple para ejecutar programas externos, software o gateways bajo un servidor de información de una manera independiente de la plataforma, llamada interfaz de entrada común (CGI, del inglés Common Gateway Interface) [12], junto con la librería Cgicc [13], que simplifica la creación de aplicación CGI en C++.

[8] ROBERTS, Mark; TORRES, Giovanni. Pyslurm: Slurm interface to python. [en línea]. febrero 2019. Disponible en https://pyslurm.github.io/.

[9] Fischer, Josh. Heron. [en línea]. Agosto 2019. Disponible en https://github.com/apache/incubator-heron.

[10] OSC. Open on demand. [en línea]. Agosto 2019. Disponible en https://openondemand.org/.

[11] SLURMWEB. Slurm-web. [en línea]. Agosto 2019. Disponible en https://edf-hpc.github.io/slurm-web/introduction.html.

[12] ROBINSON, David. The common gateway interface (cgi) version 1.1. [en línea]. 2004. Disponible en https://tools.ietf.org/html/rfc3875.

[13] Diaz, Sebastian. Cgicc - gnu project - free software foundation. [en línea]. 2016. Disponible en https://www.gnu.org/software/cgicc/index.html.

1. PLANTEAMIENTO Y JUSTIFICACIÓN DEL PROBLEMA

SLURM [14] es un planificador de tareas de uso libre usado por diversos súper computadores a nivel mundial con arquitectura de cluster como GUANE [15], el cluster de la universidad Industrial de Santander. Los ordenadores de un cluster comparten sus recursos con diferentes usuarios para el envío y ejecución de trabajos usado scripts Shell. Típicamente, estos usuarios hacen parte de diferentes áreas de conocimiento y no exclusivamente relacionadas con la computación [16], haciendo que la creación de scripts Shell o la interacción con cónsolas UNIX sea algo desafiante.

El flujo de trabajo para la creación y envío de trabajos usando SLURM es el siguiente [17]: un usuario se conecta a la consola del cluster usando algún protocolo de conexión remota y un emulador de consola, normalmente estos clusteres usan como sistema operativo alguna distribución de UNIX, una vez en la consola del cluster, el usuario deberá usar un editor de texto basado en comandos para la creación del script para solicitar los recursos usando las directivas de SLURM. Finalmente, deberá ejecutar un último comando para la solicitud de recursos. El anterior flujo se puede resumir en 3 pasos, el primero es el uso e instalación de un emulador de consola para conectarse con el cluster, el segundo es el uso de los editores de consola , los cuales requieren el uso de comandos para realizar una interacción, el tercero es el uso de las directivas propias de SLURM para solicitar recursos, estos pasos se pueden hacer tediosos para usuarios que requieren el uso de los clusters para solicitar y enviar trabajos a SLURM, por esto se hace necesario la creación de

[14] Ibít., p. 12
[15] Ibít., p. 14
[16] SC3. Proyectos – sc3 uis. [en línea]. 2019. Disponible en http://www.sc3.uis.edu.co/lista-de-proyectos/.

[17] DIAZ, Gilberto. ¿cómo acceder a los recursos? [en línea]. 2019. Disponible en http://wiki.sc3.uis.edu.co.

una interfaz web que facilite el envío de trabajos a SLURM por medio de un formulario web o subiendo un script shell para solicitudes más específicas.

2. OBJETIVOS

2.1 OBJETIVO GENERAL

Diseñar una interfaz web que facilite: la visualización de la cola de trabajos y el envío de trabajos al manejador de tareas SLURM a través de un formulario web.

2.2 OBJETIVOS ESPECÍFICOS

- Determinar los datos necesarios a obtener del servidor de autenticación para la ejecución de los trabajos en SLURM.
- Diseñar una interfaz web adaptativa (también responsive) para el uso de la API desde cualquier dispositivo.
- Determinar los datos y comandos necesarios para obtener la carga computacional y la cola de trabajos en el cluster.
- Evaluar el rendimiento del envío de trabajos a SLURM usando la Web y usando el protocolo SSH.
- Elaborar un artículo publicable basado en el trabajo desarrollado.

3. MARCO TEÓRICO

Un cluster es una colección de computadores tradicionales interconectados por una red, y un conjunto de bibliotecas y aplicaciones que lo hacen ver como un único computador con mucho poder de cálculo [18]. También se suele definir un cluster como un conjunto de computadoras de uso común dedicadas a una red para capturar su poder de procesamiento acumulativo para ejecutar aplicaciones de procesamiento paralelo [19]. Estos clusteres de computadoras han sido ampliamente usados a lo largo de la historia desde 1977 con el primer cluster ARCnet diseñado por Datapoint [20], hasta la actualidad donde el 88.4% de los supercomputadores usan arquitectura de clusteres. [21].

Típicamente, la arquitectura que siguen estos clusters de cómputo se compone por nodos conectados en una red, donde cada nodo puede tener uno o muchos procesadores, memoria RAM, interfaces de red, dispositivos de entrada y salida y su propio sistema operativo. Estos nodos de cómputo suelen dividirse en dos: nodos maestros, los cuales gestionan los recursos a través de un middleware especializado que maneja la reservación de recursos para la ejecución de los programas, el balanceo de la carga de trabajo, etc. Y nodos de cómputo los cuales se encargan de ejecutar todos los trabajos como se muestra en la Figura 2.

---

[18] EADLINE, Douglas. *High Performance Computing for Dummies*. Indianápolis: Wiley Publishing, Inc., 2009.p. 52.

[19] BAKER, Mark; BUYYA, Rajkuma. Cluster computing: the commodity supercomputer. *Software: Practice and Experience* 1999. *vol 29*, p 551–576.

[20] SHARMA, Priti; KUMAR, Brijesh; GUPTA, Pooja An introduction to cluster computing using mobile nodes. In *2009 Second International Conference on Emerging Trends in Engineering & Technology* IEEE, 2009, pp. 670–673.

[21] Ibít., p. 12.

En la antigüedad, los clusteres de cómputo solían usar sistemas operativos específicos para cada máquina. Por ejemplo, el sistema operativo VAX/VMS [22]diseñado para el cluster VAX-11. En la actualidad la familia de sistemas operativos más usada es LINUX, donde el 100% de las supercomputadoras del mundo usan esta familia de sistema operativo según el top 500 [23].

3.1 GESTOR DE RECURSOS

A lo largo del tiempo las computadoras, desde la Z3 conocida como la primera máquina calculadora programable del mundo[24] hasta Summit el cluster de OAK RIDGE National Laboratory, que es actualmente la computadora más rápida del mundo según el Top 500 de noviembre del 2018[25], buscan satisfacer una o muchas necesidades de los usuarios, además de aplicaciones

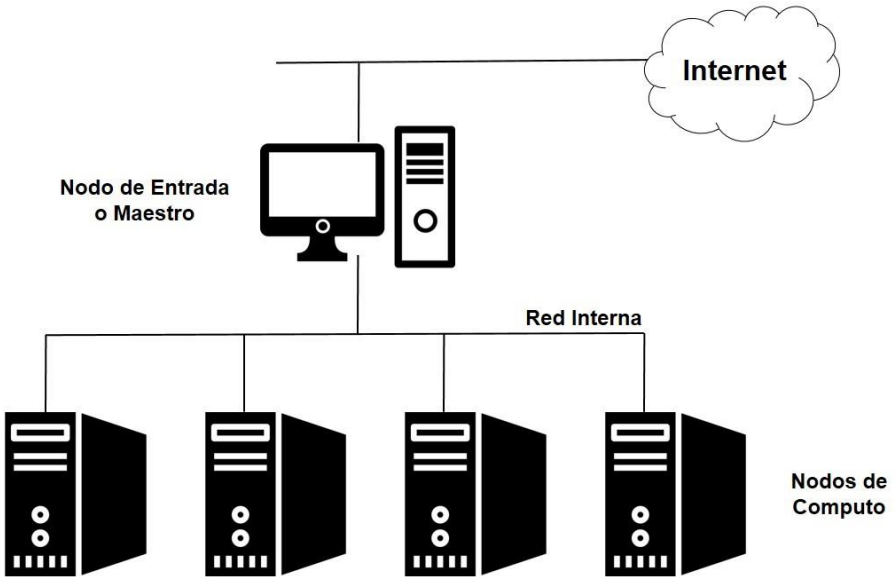
[22] OFFSET, Byte. Virtual memory management in the vax/vms operating system. *Computer. Vol. 50*, 1982 p. 36.

[23] SANKARANARAYAN, Mukund. et al. Resource manager architecture, Sept. 28 Google Patents, 2004. US Patent 6, 799, 208.

[24] ROJAS, Raul. Konrad zuse's legacy: the architecture of the z1 and z3. *IEEE Annals of the History of Computing. Vol. 19*, 1997, p. 5–16.

[25] Strohmaier, Erich. et al. Top 500 noviembre 2018. [en línea]. 2018. Disponible en https://www.top500.org/list/2018/11/.

Figura 2. Arquitectura común en un cluster de computación, compuesto por un nodo maestro que gestiona los recursos a través de un middleware y nodos de cómputo los cuales ejecutan los trabajos.



de escritorio como editores de texto y correos electrónicos, es común que reproduzcan música, visualicen imágenes, entre otras tareas. A medida que se solicitan estas tareas, es frecuente que los usuarios esperen que las computadoras respondan a todas ellas y sea la computadora quien las gestione, creando una mayor demanda de recursos y provocando una mayor probabilidad de que la computadora no tenga los recursos suficientes en un momento solicitado para realizar todas las tareas simultáneamente [26].

La gestión de recursos limitados es cada vez más importante a medida que más de un usuario hace uso de una computadora como en el caso de clusteres de cómputo donde más de un usuario accede para ejecutar sus trabajos. En este entorno la

[26] SANKARANARAYAN, Mukund. et al. Resource manager architecture, Sept. 28 Google Patents, 2004. US Patent 6, 799, 208.

computadora debe decidir qué recursos destina a cada usuario, que prioridades debe tener cada usuario, como gestionar los recursos libres, las colas de trabajo, entre otras tareas. Sin el uso los gestores de recursos, el manejo de los clusteres se volvería intratable en proporción a la cantidad de usuarios.

3.2 SLURM

SLURM es un gestor de pequeños y grandes clusteres (middleware) Linux de código abierto desarrollado por Lawrence Livermore National Laboratory, el cual no requiere modificaciones al kernel para su funcionamiento y es relativamente autónomo. SLURM inició debido al crecimiento de los clusteres de cómputo y por la no existencia de gestores de recursos de código abierto que satisficieran las necesidades de estos clusteres, necesidades como tolerancia a fallos y escalabilidad.

Por esta razón, SLURM fue diseñado con los siguientes objetivos:

- Simplicidad: su diseño es lo suficientemente simple para permitir que usuarios finales puedan entender su código y agregar funcionalidades.
- Código abierto: SLURM esta disponible para cualquier persona y de uso libre bajo la GNU General Public License[27].
- Portable: está escrito en el lenguaje C lo que lo hace fácil de importar a otros sistemas UNIX y permite una variedad de diferentes infraestructuras.
- Escalabilidad: SLURM es diseñado para escalarse a sistemas con miles de nodos. Los usuarios pueden especificar una gran variedad de

[27] TSAI, Jonh. For better or worse: Introducing the gnu general public license version 3. *Berkeley Tech* 2008*. LJ vol. 23*, p 547.

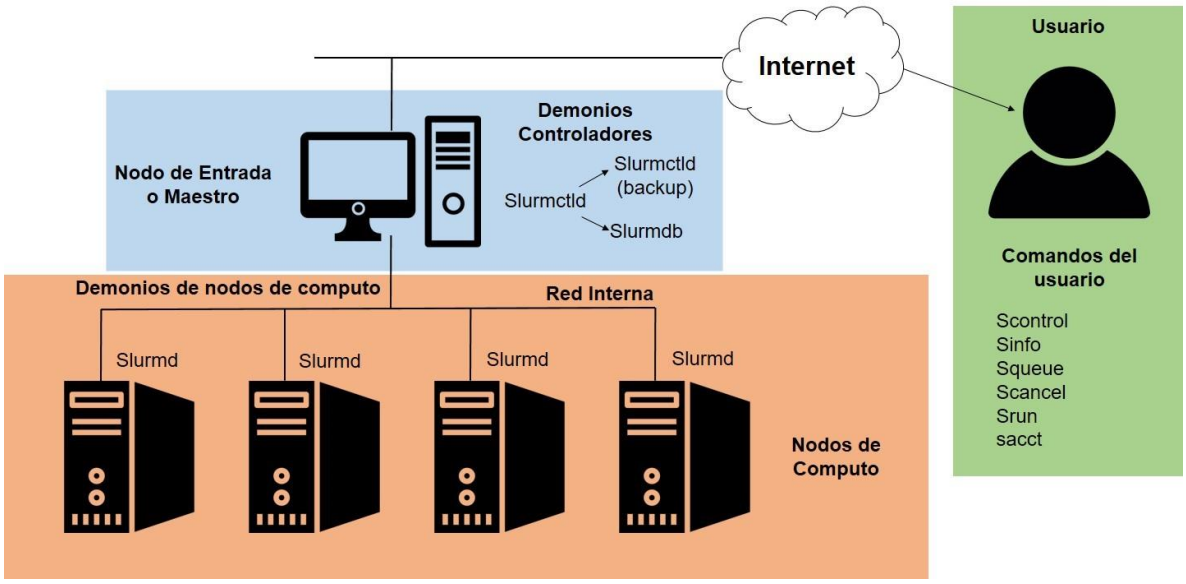
requisitos de recursos permitiendo potencialmente una iniciación más rápida.

- Robustez: SLURM puede manejar una variedad de fallos sin terminar los trabajos, incluyendo fallos en el nodo de ejecución de SLURM. Los trabajos de los usuarios pueden ser configurados para continuar en ejecución a pesar del fallo de uno o más nodos en los que se estuvieran ejecutando.
- Seguridad: SLURM emplea tecnología de encriptación para autenticar usuarios entre servicios y servicios entre sí. SLURM no asume que sus redes son físicamente seguras, pero sí que todo el cluster está dentro de un único dominio administrativo con una base de usuarios común en todo el cluster.
- Administrador de sistema amigable: Se diseñó para ser configurado desde un archivo de texto simple. Esta configuración puede ser cambiada a cualquier tiempo sin afectar los trabajos en ejecución.

3.2.1 Arquitectura de SLURM. La arquitectura de un cluster con SLURM como Guane se basa en un administrador centralizado, slutmctld, el cual monitorea los recursos y los trabajos. También cuenta con la opción de un administrador de respaldo en caso de que el administrador principal falle. Cada servidor de cómputo tiene un demonio llamado *slurmd*, que se puede comparar con un Shell remoto. Los demonios *slurmd* proporcionan comunicaciones jerárquicas tolerantes a fallos. Existe un demonio opcional, *slurmdb*, que se puede usar para registrar información de contabilidad para múltiples clusteres administrados por SLURM en una sola base de datos. Las herramientas de usuario incluyen *srun* para iniciar trabajos, *scancel* para terminar trabajos en cola o en ejecución sin reportar el estado del sistema, *squeue* para informar del estado de los trabajos, y *sacct* para obtener información acerca de los trabajos y los pasos de trabajo que se están ejecutando o que se hayan completado. Los comandos *smap* y *sview* informan gráficamente el estado del sistema y del trabajo, incluida la topología de red[28]. También existe una herramienta administrativa llamada *scontrol* disponible para monitorear o modificar la configuración y estado del cluster. La organización y comunicación entre los diferentes demonios es mostrada en la Figura 3.

[28] YOO, Andy; JETTE, Morris; GRONDONA, Mark. Slurm workload manager - overview. [en línea]. 2019. Disponible en https://slurm.schedmd.com/overview.html.

Figura 3. Organización y comunicación entre los diferentes componentes en un cluster con SLURM.

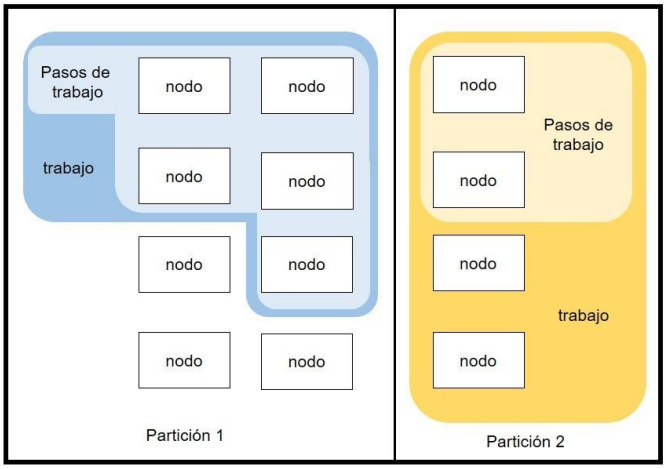


la Figura 4 muestra las entidades manejadas por los demonios de SLURM, incluyendo nodos, particiones, las cuales son grupos lógicos de nodos, trabajos o asignaciones de recursos asignados a un usuario por un periodo de tiempo y pasos de trabajo, los cuales son conjuntos de tareas dentro de un trabajo. Las particiones se pueden considerar con colas de trabajos, donde cada partición tiene una variedad de restricciones tales como, límite de tamaño de trabajo, límite de tiempo de trabajo, usuarios autorizados para cada partición, entre otras.

3.3 Arquitectura Guane

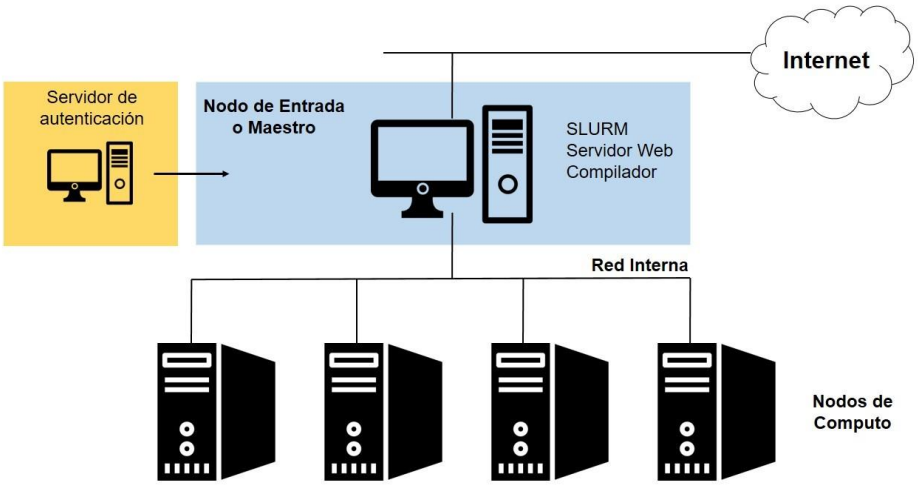
La arquitectura de Guane está compuesta por 16 nodos que usan SLURM como gestor y planificador de tareas, cuenta en su nodo maestro con un servidor web, un compilador y los demonios controladores de SLURM, también cuenta con un servidor de autenticación para el ingreso a la

Figura 4. Organización de las diferentes entidades en SLURM



consola del cluster como lo muestra la Figura 5. El funcionamiento, utilidad y especificación de cada uno de estos servidores y software se explican a continuación:

Figura 5. Arquitectura Guane



3.3.1 Servidor de Autenticación. El servidor de autenticación es el encargado de verificar que las credenciales de cada usuario (nombre de usuario y contraseña) sean los correctos para ingresar al cluster, en este Guane cuenta con el Protocolo ligero de acceso a directorios (LDAP del inglés Lightweight Directory Access Protocol, LDAP) que es un protocolo de Internet para acceder a servicios de directorio distribuidos que actúan de acuerdo con los modelos de datos y servicios de X.500. [29]. Este protocolo se utiliza a nivel de aplicación para acceder a los servicios de directorio remoto [30]. Al estar basado en el protocolo X.500 LDAP organiza la información de forma jerárquica usando directorios, proporcionando una estructura intuitiva desde el punto de gestión para los administradores.

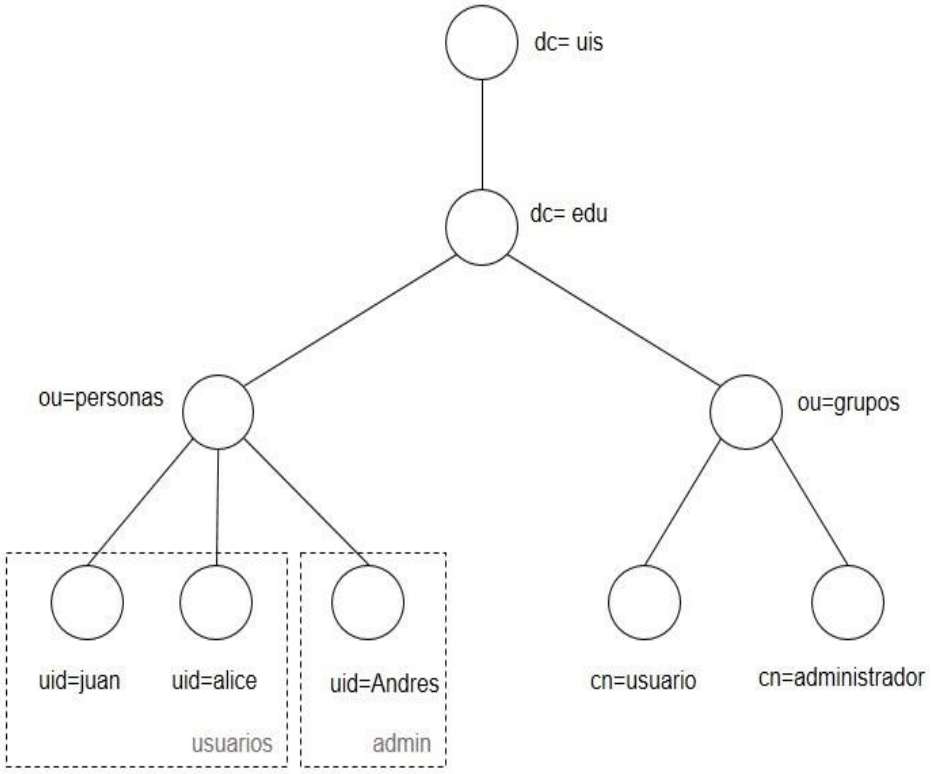
El modelo de información de LDAP está basado en entradas. Una entrada es una colección de atributos que tienen un Nombre Distinguido (DN) único y global. Este nombre se utiliza para dar un identificador único e irrepetible a una entrada del directorio. Cada atributo posee un tipo identificador y su correspondiente valor. Los tipos se utilizan para identificar los nombres de los atributos por ejemplo: 'cn' para *common name* o *mail* para una dirección de correo entre otros. Algunos de los atributos que pertenecen a una entrada deben ser obligatorios y otros opcionales. También se utiliza LDIF que es el formato de intercambio de datos de LDAP que es la representación en texto ASCII de las entradas LDAP, este debe ser el formato de los archivos que se utilicen para importar información a un directorio LDAP. Cuando se escribe una línea en blanco significará que es el fin de una entrada. Estas entradas están organizadas en una estructura jerárquica en árbol que permite una mejor administración y visualización. La Figura 6 muestra un árbol de directorios

[29] ZEILENGA, Kurt. Lightweight directory access protocol (ldap): Technical specification road map. 2006. p.p

[30] HOWES, Timothy; SMITH, Mark; GOOD, Gordon S. *Understanding and deploying LDAP directory services*. Addison-Wesley Longman Publishing Co., Inc., 2003.p.p

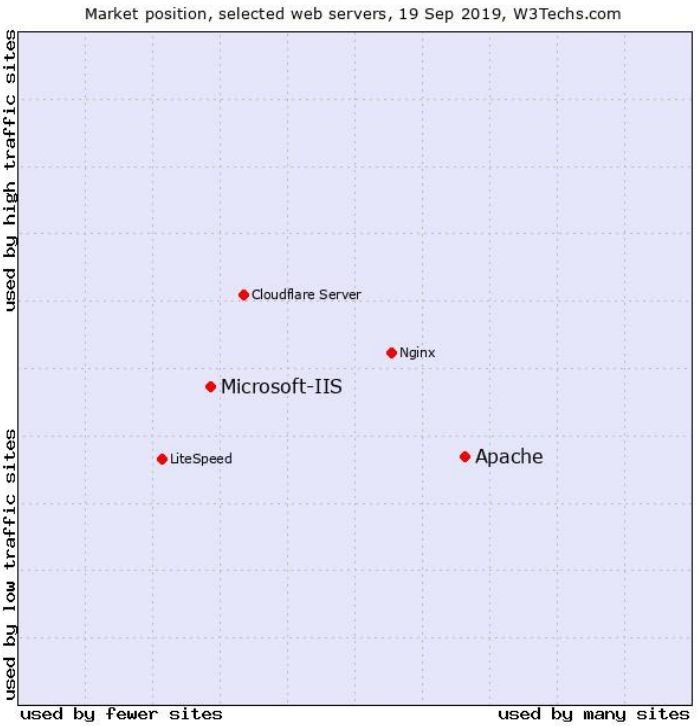
típico de LDAP donde el DN está compuesto por dc = uis, dc = edu, el cual tiene dos unidades organizacionales (OU) llamadas usuario y administrador, dos usuarios llamados juan y alice son miembros de usuario y Andrés es miembro del grupo administrador.

Figura 6. Árbol jerárquico en un servidor LDAP



3.3.2 Servidor Web. Un servidor web, explicado de una forma simple es el encargado de aceptar solicitudes de clientes, por ejemplo el navegador web de un visitante y luego enviar respuesta a esa solicitud. Apache es uno de los servidores web multiplataforma de fuente abierta más populares, antiguo y confiable, lanzó su primera versión en 1995. Algunas empresas de alto perfil que utilizan Apache incluyen a Cisco, IBM, Salesforce, General Electric, Adobe, VMware, Xerox, LinkedIn, Facebook, Hewlett-Packard, AT & T, Siemens, eBay y muchas más[31]. Actualmente es usado por el 44% de sitios web de todo el mundo a pesar de sus problemas a la hora de usarse en sitios web con mucho tráfico. Como lo muestra la figura 7 actualmente es mantenido por la Apache Software Foundation.

Figura 7. Servidores web más usados



[31] MAVRIDIS, Ilias; KARATZA, Helen. Performance evaluation of cloud-based log file analysis with apache hadoop and apache spark. *Journal of Systems and Software* 2017, *vol. 125. p.* 133–151.

El servidor apache es altamente personalizable ya que cuenta con muchos módulos que agregan funciones a su software como MPM para el manejo de modos de procesamiento múltiple, ssl para habilitar SSL y TLS o CGI que habilita el uso de archivos CGI en el servidor web otros módulos de apache sirven para almacenamiento en caché, reescritura de URL, autenticación de contraseña y más.

3.3.2.1 Diseño Web El diseño web es la actividad que consiste en la planificación, diseño e implementación de sitios web. Engloba una gran variedad de aspectos como puede ser el diseño de la interfaz, el material gráfico, la visualización entre otros.

Bootstrap es una biblioteca multiplataforma de código abierto para el diseño de sitios y aplicaciones web, facilita la maquetación de sitios web ofreciendo herramientas para que los sitios web se vean de manera apropiada en todos los dispositivos, permitiendo la creación de sitios web adaptable a los diferentes tamaños de pantallas de una manera fácil. Este framework cuenta con plantillas de diseño con tipografía, formularios, botones, cuadros, menus de navegación y otros elementos de diseño basado en HTML y CSS, así como extensiones de JavaScript adicionales, además es compatible con la mayoría de sitios web.

Bootstrap es modular y consiste esencialmente en una serie de hojas de estilo LESS que implementan la variedad de componentes de la herramienta. Una hoja de estilo llamada bootstrap.less incluye los componentes de las hojas de estilo. Los desarrolladores pueden adaptar el mismo archivo de Bootstrap, seleccionando los componentes que deseen usar en su proyecto.

Actualmente Bootstrap añadió un sistema de GRID que permite diseñar la página web usando una GRID de 12 columnas donde se plasma el contenido permitiendo diseñar las páginas de una forma mucho más fácil e intuitiva

3.3.3 Compilador. La interfaz de entrada común (CGI, del inglés Common Gateway Interface) es un método para la transmisión de información hacia un compilador o intérprete instalado en el servidor. En una aplicación CGI, el servidor web pasa las solicitudes del cliente a un programa externo evitando tener que instalar software adicional por lo que es dependiente del servidor y no del usuario. Este programa puede estar escrito en cualquier lenguaje que soporte el servidor, aunque por razones de portabilidad se suelen usar lenguajes de script. La salida de dicho programa es enviada al cliente en lugar del archivo estático tradicional.

1. En primera instancia, el servidor recibe una petición (el cliente ha activado un URL que contiene el CGI ya sea por un formulario o de manera involuntaria) y comprueba si se trata de una invocación de un CGI.

2. Posteriormente, el servidor prepara el entorno para ejecutar la aplicación. Esta información proviene mayoritariamente del cliente.

3. Seguidamente, el servidor ejecuta la aplicación, capturando su salida estándar.

4. A continuación, la aplicación realiza su función: como consecuencia de su actividad se va generando un objeto MIME (text, multipart, message, image, audio, video o application) que la aplicación escribe en su salida estándar.

5. Finalmente, cuando la aplicación finaliza, el servidor envía la información producida, junto con información propia, al cliente, que se encontraba en estado de espera. Es responsabilidad de la aplicación anunciar el tipo de objeto MIME que se genera en el campo CONTENT_TYPE.

4.METODOLOGÍA

En este trabajo se presenta una interfaz web para el envío de trabajo a SLURM usando CGI con C++ facilitando el envío y uso de clusters de cómputo para las áreas que no están familiarizadas con el uso de consola de sistemas UNIX.

La API propuesta en este trabajo se le dio el nombre de Web Submit SLURM (WebSS) y se desarrolló teniendo en cuenta cómo viaja la información a través del sistema y cómo se comunican los diferentes actores que participan en este mismo, también se usó el modelo de cascada para tener una estructura y control durante el proceso de desarrollo e implementación del software.

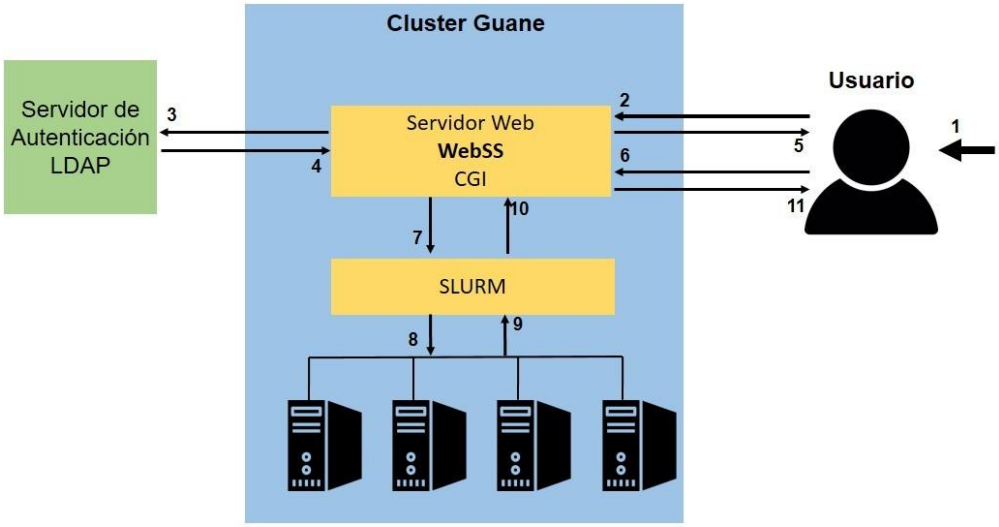
4.1 FLUJO DE INFORMACIÓN.

El modo en que viaja la información a través de un cluster que usa SLURM se usó para la implementación de este trabajo, éste ayudó a tener un modelo preliminar permitiendo tener en cuenta todos los actores y como se comunican estos entre si facilitando el desarrollo de la API propuesta. El flujo de información se muestra en la figura 8 donde los actores representan los servidores, usuarios, aplicaciones entre otros, estos se representaron por un cuadro, las flechas representan como viaja la información entre los diferentes actores y como se observa cada flecha tiene una entrada y una salida esto se debe a que cada actor recibe y responde a los mensajes o consultas de otros actores, cada número sobre cada una de las flechas representa el orden de los mensajes entre los diferentes actores a lo largo de un envío de trabajo usando WebSS. El cuadro más grande de color azul representa el cluster Guane donde se encuentran otros actores como el servidor web, la API propuesta WebSS, los CGI y SLURM, ésta arquitectura es necesaria para que la API funcione. El servidor web y los CGI serán representados y nombrados como uno solo WebSS

para facilitar la demostración del flujo, una explicación más detallada de la comunicación e interacción con estos se dará en una sección posterior.

La interpretación de cada una de las flechas, actores y números y la forma en que se implementó WebSS es como sigue: El inicio del flujo de información comienza con el usuario al usar

Figura 8. Estructura básica, autores y flujo de información en WebSS



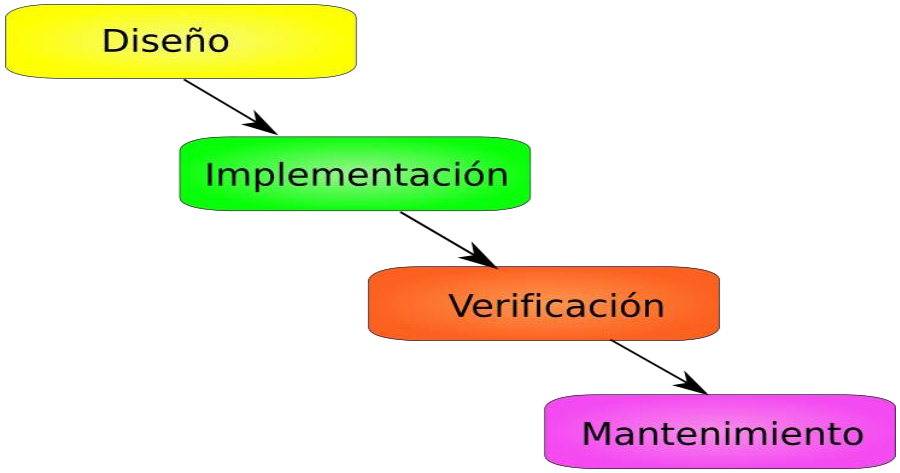
WebSS por medio de un navegador web(1), WebSS le muestra al usuario un formulario HTML donde pedirá el nombre del usuario y la contraseña, el usuario deberá introducir esta información y enviarla(2), WebSS recibe esta información y la valida con el servidor LDAP(3), que en este caso es usado como servidor de autenticación, LDAP valida el usuario y hace algunas consultas que devuelve a WebSS(4), WebSS recibe la información y le muestra al usuario la interfaz web para el envió y consulta de trabajos a SLURM(5), en esta parte el usuario puede simplemente consultar la cola de trabajos o enviar su propio script con los recursos y trabajos que desea correr o usando un formulario PHP para la solicitud de recursos y envío de trabajos, una vez el usuario llene el formulario o use su propio script lo

envía a WebSS(6), WebSS con el uso de llamados al sistema a través de un CGI con C++ pide los recursos a SLURM(7) este consulta con el cluster de cómputo si cuenta con los recursos(8) quien a su vez responde a SLURM la disponibilidad de los mismos(8). SLURM procesa la información enviada por el cluster y envía la salida del proceso a WebSS(10) quien a su vez la muestra al usuario(11) la salida enviada de SLURM junto con la cola de trabajos para que pueda observar el estado y datos de su trabajo, terminando así el flujo de la información para el envío de trabajos a SLURM usando CGI con C++.

4.2 MODELO DE CASCADA

El desarrollo de WebSS se lleva a cabo usando el modelo de cascada este un enfoque de ingeniería de software metodológico que concibe el desarrollo del software como un conjunto de etapas rigurosas que se ejecutan una detrás de la otra, esto quiere decir que el inicio de una nueva etapa es el final de una etapa anterior, se denomina cascada por que las diferentes etapas que componen el proyecto se colocan una encima de la otra y su ejecución es de arriba hacia abajo como una cascada. La figura 9 muestra el diagrama del modelo en cascada que se usó en este trabajo.

Figura 9. Modelo de cascada usado en el desarrollo de WebSS



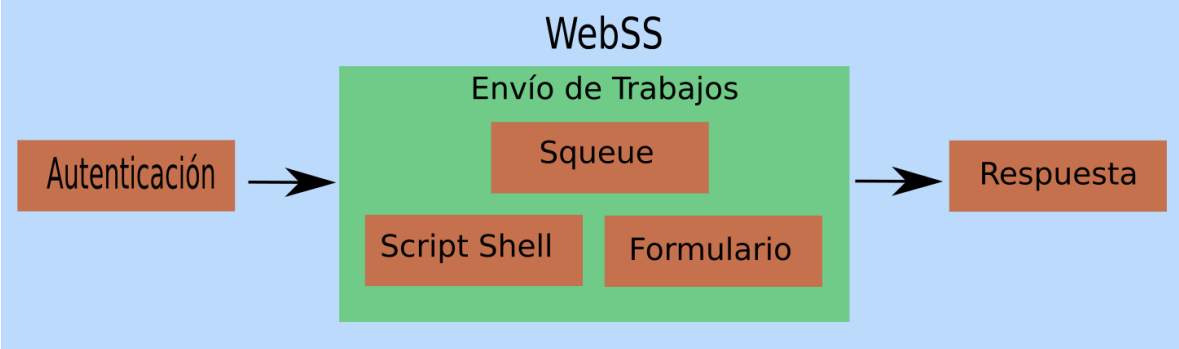
4.2.1 Diseño. El diseño de WebSS se planteó teniendo en cuenta experiencias de los usuarios, en una primera etapa se pensó usar solo un formulario HTML donde los usuarios podían de una forma fácil introducir los recursos que necesitaban y los trabajos que deseaban correr, pero debido a la gran cantidad de directivas que SLURM nos permite seleccionar para solicitar recursos daba como resultado un formulario extenso y con algunas directivas que solo algunos usuarios con experiencia usarían para trabajos más especializados, se decidió acortar esta lista y usar solo directivas de uso más común, pero se necesitaba una forma de que usuarios más experimentados pudieran solicitar recursos de una forma más especializada, para esto se implementó también el envío de trabajos usando Scripts shell con los recursos y trabajos que se querían correr en caso de que el formulario proporcionado no tuviera las directivas que los usuarios requieran, esto permitiría tanto a usuarios nuevos hacer uso del cluster sin tener que profundizar mucho en las directivas básicas de SLURM y a usuarios más experimentados poder hacer el uso de scripts para el envío de trabajos más complejos.

En una segunda etapa se observó también la necesidad de ver la cola de trabajos para ver el estado de los trabajos enviados sin tener que acceder al cluster de forma remota por consola, para esto se desarrollo una tabla que permitiría ver la cola de trabajos de forma rápida y sencilla a través de la API propuesta.

En una tercera etapa se pensó en el uso de WebSS no solamente desde un computador de escritorio o portátil si no también hacer la API aún más portable permitiendo el envío de trabajos a través de dispositivos móviles, este daría aun una mayor portabilidad y facilidad en el uso de clusters de cómputo.

4.2.2 Implementación. La implementación de WebSS se desarrolló teniendo en cuenta como viajaba la información al enviar un trabajo a SLURM permitiendo saber con que actores se debía tener comunicación directa y la etapa anterior de diseño permitiendo satisfacer las necesidades de los usuarios. Se hizo un diseño de tres módulos para la implementación de WebSS como lo muestra la figura 10. Cada uno de estos módulos se presenta al usuario como una interfaz web diferente.

Figura 10. Módulos en la implementación de WebSS



El primer módulo corresponde a la autenticación del usuario, es la primer interfaz de la API, donde se le solicita al usuario usando un formulario HTML el nombre de usuario y contraseña para ser verificada con el servidor LDAP por medio de una validación PHP usando el DN del usuario conformado por el identificador del usuario o UID que en este caso es el nombre y las unidades organizativas uis.edu.co. Una vez validado el usuario se hace una consulta al servidor LDAP para conocer el UID y GID para enviar esta información al siguiente módulo. En caso de fallar la autenticación se le notifica al usuario un error de autenticación y este podrá volver a intentar el ingreso a la API.

El segundo módulo corresponde al envío de trabajos, este módulo esta conformado por 3 submódulos y corresponde a la segunda interfaz mostrada al usuario después de una autenticación exitosa. El primer sub-módulo corresponde a la cola de trabajos y a los formularios correspondientes tanto al envío de scripts shell o de la

solicitud de recursos por medio de un formulario HTML. La cola de trabajos se diseño usando llamados al sistema utilizando PHP para obtener la salida del comando squeue de SLURM y por medio de una tubería y de la aplicación sed de UNIX para dar el formato apropiado aprovechando los saltos de línea y los espacios entre palabras para obtener una tabla HTML en un solo llamado al sistema. El segundo sub-módulo corresponde al envío de trabajos a SLURM usando un script shell en caso de que el usuario hubiera deseado esto, en este sub-módulo se hace uso de un CGI con C++ para ejecutar llamados al sistema y enviar los trabajos, en un primer paso se retorna el script shell enviado por el usuario por el método POST junto con el UID y el GID del usuario consultado del servidor de autenticación, se hace uso de la librería CGICC para recibir estos datos, una vez retornado estos datos se crea en el home del usuario en el cluster un archivo que contendrá el contenido del Script enviado por el usuario, seguidamente se comienza a formatear el comando que se ejecutará en el cluster para enviar el trabajo a SLURM con un llamado al sistema, la salida que retorna SLURM del éxito o fallo del envío es almacenada en un archivo de texto. El último sub-módulo corresponde al envío de trabajos a SLURM usando un formulario en caso de que el usuario hubiera deseado esto, en una primera instancia se retorna del servidor de autenticación el GID y el UID del usuario, luego se crea un archivo de texto vacío en el home del usuario en el cluster y se le agrega como primera línea el llamado al intérprete encargado de dar significado a los comandos, seguidamente con ayuda de la librería CGICC se comienza a recibir los datos enviados en el formulario por el usuario y línea por línea se comienza añadir las directivas de SLURM con los recursos que solicitó el usuario para así crear el script con el cual se solicitaran los recursos, después, se comienza a dar formato al comando con el que se solicitarán los recursos a SLURM y al igual que el anterior sub-módulo guardar su salida para manejo de posibles errores.

El tercer módulo corresponde a la respuesta de WebSS al usuario del envío de su trabajo, en una primera versión este módulo solo mostraba la salida que generaba SLURM al enviar el trabajo, pero también se vio la necesidad de poder ver la cola

de trabajos para ver el estado del trabajo que se había enviado, para esto se implementó la cola de trabajos de la misma forma en la que se hizo en el módulo dos.

Todos los módulos se diseñaron haciendo uso del framework Bootstrap para permitir el uso de la API en cualquier dispositivo.

4.2.3 Verificación. La implementación y diseño de WebSS se llevó por cuestiones prácticas para realizar pruebas sin afectar otros sistemas ni usuarios en un computador de escritorio con un procesador Intel Core i5 7200U con sistema operativo Linux Mint 18.3 Sylvia, OpenLDAP 2.4.42 y SLURM 15.08.7, mostrando resultados positivos, libres de errores y fallos de seguridad pero en un entorno controlado sin agentes externos y con usuarios artificiales. Para comprobar el funcionamiento correcto de la API en un entorno industrial se implementó y verificó el correcto funcionamiento de WebSS en Guane el cluster a cargo del grupo de súper computación y calculo científico (SC3) ubicado en el parque tecnológico Guatiguara, Santander, Colombia, esta compuesto por 16 nodos ProLiant SL390s G7, sistema operativo CentOS 7.6.1810 (Core), OpenLDAP 2.4.48 y SLURM 18.08.7. En este entorno la API será sometida a usuarios y pruebas reales con usuarios que no tendrán conocimiento de la codificación de la API.

4.2.4 Mantenimiento. WebSS se diseñó con la premisa de ser compatible con cualquier sistema operativo basado en UNIX, de igual forma que fuera independiente de la versión de SLURM y del tipo de cluster en el cual se ejecutará, en este caso, se usó LDAP como servidor de autenticación pero el uso de otro servicio de autenticación no debería afectar el funcionamiento de la API, el único requisito es que el servidor guarde la información del UID y del GID de cada usuario y como se pudo observar en la sub-sección anterior se implementó WebSS en dos sistemas diferentes tanto en poder de cómputo, versiones y sistema operativo sin problemas de compatibilidad.

4.3 INSTALACIÓN

La instalación y preparación de los requisitos y del sistema para el uso de WebSS se realizó como sigue:

En una primera instancia se instalaron las librerías para compilar C++ en este caso se optó por G++ como compilador, este se usó para compilar los CGI. Como segundo paso se instaló el servidor web, este debe ser instalado en el nodo maestro de SLURM debido a que los llamados al sistema simulan como si fuera el usuario quien envía el trabajo. Seguidamente, se instala PHP y se procede a configurar el servidor web, en este caso se usó apache como servidor web y debido a su estructura de módulos se tuvieron que activar e instalar los módulos para poder hacer la autenticación con LDAP, los módulos de PHP y los módulos de CGI, estos permitirán al servidor web ejecutar de forma correcta la API. En un cuarto paso se instaló la librería libre CGICC que permite tener un mejor manejo de los elementos enviados por POST a un CGI con C++ facilitando la implementación, y como último paso se instaló Bootstrap para hacer la API adaptativa al tamaño.

Una vez instalado y configurado todos los requisitos solo es necesario descargar los archivos HTML y PHP y ponerlos en el path indicado por el servidor web, también se debe modificar las líneas de autenticación para el servicio con el que cuente el cluster por defecto es LDAP, de ser el mismo, solo seria necesario cambiar el DN de la búsqueda y la dirección del servidor LDAP. Seguidamente se deben descargar y compilar los archivos .cpp que son los correspondientes a los cgi estos se deben compilar como súper usuario para poder ejecutar los llamados al sistema y otorgarles el permiso setuid esto para que apache pueda ejecutar estos archivos como si fuera un usuario root.

4.4 SEGURIDAD.

La seguridad en la implementación de la API también se tuvo en cuenta a la hora de la implementación, se tomaron medidas de seguridad tanto para guardar la integridad de los datos enviada por el usuario y para la seguridad del cluster mismo, esta se dividió como sigue:

- Seguridad del cluster: La primera medida de seguridad que se tomo fue asegurar que todas la sesiones abiertas fueran cerradas para evitar la inyección de información en estos campos, también se tuvo en cuenta cerrar los permisos setuid dentro del CGI.
- Seguridad de interfaz: La segunda medida de seguridad corresponde al cliente donde se uso Javascript para evitar una inyección de información en la consulta con el servidor LDAP que pudiera poner en riesgo la información y la integridad del cluster.
- Seguridad de WebSS: La tercera medida de seguridad fue asegurar que en ninguna parte del formulario o del script enviado por el usuario se pudieran insertar caracteres o palabras que pusieran en riesgo el cluster.
- Seguridad del servidor web: la ultima medida de seguridad es el uso de TLS para una comunicación segura entre el cliente y el servidor web pero esta depende de la administración del mismo.

La razón de usar CGIs en la implementación de WebSS se produjo debido a que se necesitaba una forma de ejecutar llamadas al sistema como administrador para poder enviar los trabajos. En un principio se ejecutaron los llamados al sistema usando php pero este ejecutaría los llamados con las credenciales del servidor web que en este caso es apache, por lo cual se tendría que darle credenciales de root a apache para la ejecución de los llamados al sistema lo que permite graves problemas de seguridad, por esta razón los CGIs ofrecen una alternativa de ejecutar comandos como root sin tener que poner en riesgo el cluster.

5. EVALUACIÓN Y RESULTADOS

WebSS es una API diseña para clusters de cómputo que usen SLURM como gestor de tareas, se diseño usando HTML, PHP, CGI con C++ y bootstrap, actualmente es usado por el supercomputador guane para el envío de trabajos de forma web, el cual usa LDAP como servidor de autenticación.

Su diseño se basó en una interfaz simple que no fuera complicada para los usuarios y que de igual forma demostrara el objetivo de la API hacer más fácil y simple el envío de trabajos. Consta de 3 interfaces una de inicio de sesión, una para el envío de trabajos y otra donde se muestra el resultado del envío del trabajo, se usaron colores verdes para la interfaz haciendo alusión a los colores de la Universidad Industrial de Santander.

WebSS al igual que SLURM se diseño para que sea un sistema escalable para cualquier tipo de cluster pequeño o grande y portable para cualquier sistema ya que no depende ni de la versión de SLURM, ni del sistema operativo y de igual forma la API no requiere ninguna instalación solo se debe tener los requisitos necesarios para el correcto funcionamiento de WebSS.

El diseño de la interfaz web de Webss al igual que la implementación se hizo pensando en que fuera portable por lo cual se uso bootstrap que ofrece una forma rápida y sencilla de implementar páginas web responsive usando una GRID proporcionada por el framework, esto permitió que WebSS se pueda usar en diversos navegadores web y desde dispositivos grandes como computadores de escritorio como se muestra en las figuras 11, 12 y 13, dispositivos medianos como tablets como se muestra en las figuras 14, 15 y 16, hasta dispositivos más pequeños

como teléfonos celulares como se muestra en las figuras 17, 18 y 19, donde la interfaz se adapta al tipo de pantalla de una forma agradable para el usuario final.

La contribución de este proyecto se encuentra en los archivos CGI ya que estos proporcionaron una forma segura de poder ejecutar los llamados al sistema para enviar los trabajos a SLURM. La lógica básica de los CGI tanto para el envío de trabajos por medio de un script o del formulario es como sigue: Se toma la información enviada tanto por el servidor de autenticación y del usuario ya se un script shell o el formulario con los recursos que desea, con esta

Figura 11. Interfaz de inicio de sesión en dispositivos de escritorio

Figura 12. Interfaz de envío de trabajos en dispositivos de escritorio

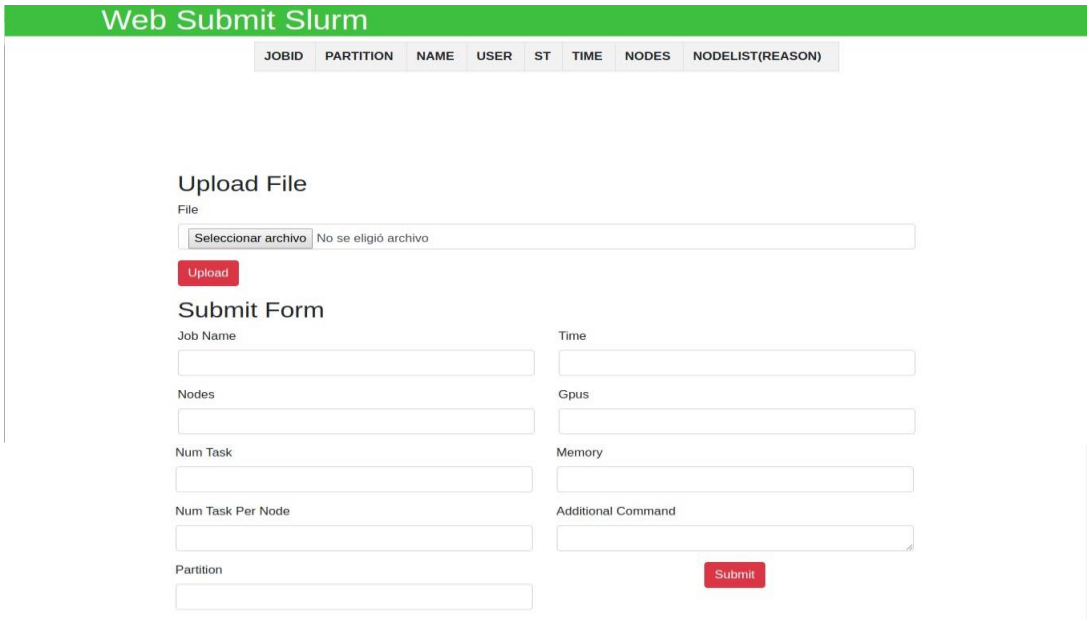


Figura 13. Interfaz de salida en dispositivos de escritorio

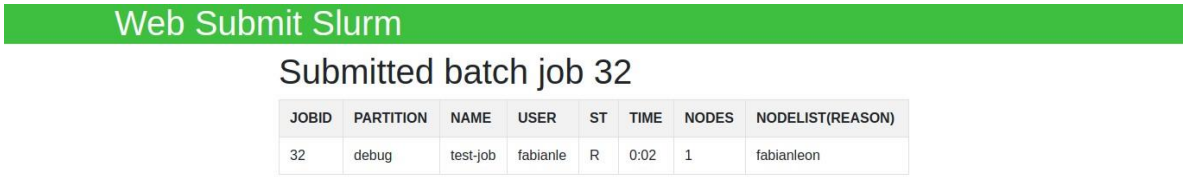


Figura 14. Interfaz de inicio de sesión en tablets

Figura 15. Interfaz de envío de envío de trabajos en tablets

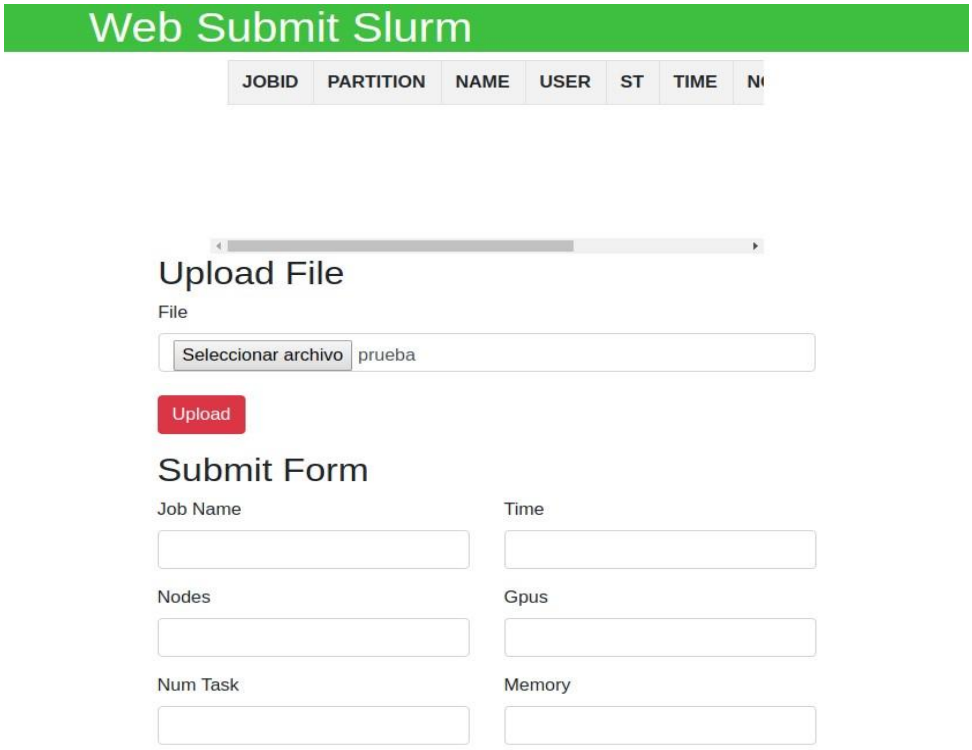


Figura 16. Interfaz de salida en tablets

Figura 17. Interfaz de inicio de sesión en dispositivos celulares

Figura 18. Interfaz de envío de trabajos en dispositivos celulares

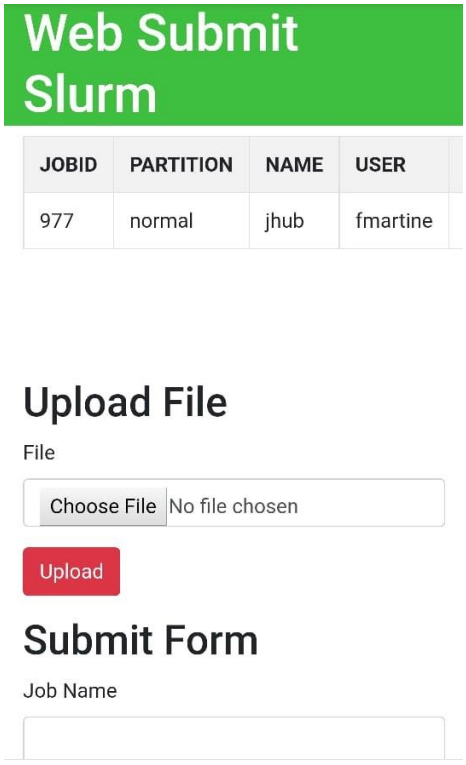
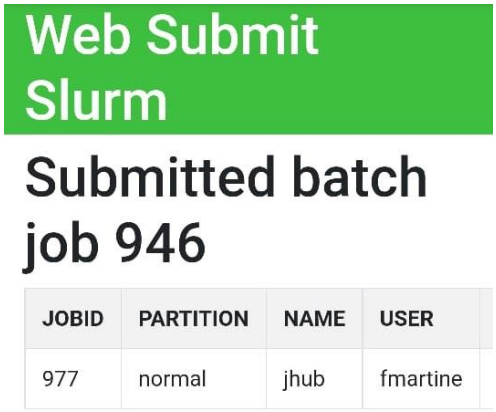


Figura 19. Interfaz de salida en dispositivos celulares



información se comienza a formatear el comando para hacer la llamada al sistema y enviar el trabajo el formato que se le da al comando inicia con sbatch para enviar los trabajos, luego se le agrega las opciones –uid y –gid acompañados de el uid y gid consultado del servidor de autenticación esto sirve para decirle a SLURM que el trabajo sera enviado por el usuario al cual le pertenece ese uid y gid seguidamente se toma ya sea el script o el formulario del usuario y se almacena en el directorio home del usuario en el cluster tomando como nombre la hora a la que se envió el trabajo y el sufijo .in seguidamente se agrega al comando el nombre de este archivo y se agrega la opción –output seguido con la hora que se envía el trabajo y el sufijo .out que es donde se almacenara los resultados del trabajo y de esta forma se daría por terminado el comando para realizar el llamado al sistema. Las líneas de código correspondientes al llamado del sistema son las que tienen permiso de ejecutarse como root puesto que el usuario root es quien puede hacer la asignación de trabajos a los usuarios por medio de las opciones –uid y –gid una vez terminado el llamado al sistema se quitan las credenciales root por razones de seguridad.

Para comprobar el rendimiento y el tiempo que tarda en enviarse un trabajo usando WebSS y usando la consola por medio de ssh, se uso la librería chrono y el comando time() de Linux respectivamente, se enviaron 10 trabajos y se capturo el tiempo de ejecución, el resultado obtenido es el mostrado en la tabla 1. Donde en promedio para el envió de 10 trabajos usando SSH fue de 0.011 segundos y usando WebSS 0.024 segundos, obteniendo un aumento de 128% usando WebSS pero en ambos casos tiempos menores a 0.03 segundos lo que para la experiencia final del usuario es algo insignificante. Aunque los tiempos de ejecución sean parecidos la verdadera contribución y facilidad de la API propuesta es a la hora de cambiar el envío de trabajos usando la consola por SSH a usar una página web reduciendo y facilitando notablemente el envío de trabajos a un cluster de computo

Tabla 1. Tiempo de ejecución en segundos enviando un trabajo usando SSH y WebSS

WebSS	0.017	0.029	0.019	0.028	0.027	0.029	0.021	0.017	0.029	0.026
SSH	0.013	0.015	0.013	0.013	0.013	0.011	0.012	0.013	0.015	0.011

	Promedio
WebSS	0.024
SSH	0.011

6. CONCLUSIONES Y PERSPECTIVAS

En este trabajo se presenta el desarrollo de una interfaz web para el envío y consulta de colas en SLURM usando CGI con C++ de una forma rápida, fácil, segura, escalable a los diferentes tamaños de cluster y portable entre los diferentes sistemas, arquitecturas, navegadores y dispositivos. La implementación y desarrollo se realizaron en el cluster Guane donde esta actualmente funcionando. El tiempo de ejecución al momento de el envío de trabajos usando el protocolo SSH y usando la API propuesta fue parcialmente parecido, para el protocolo SSH fue de 0.011 segundos y para WebSS fue de 0.024 segundos mostrando un diferencia menor a 0.02 segundos sin mostrar un cambio significativo para el usuario final, en cuanto a tiempo de ejecución, pero un gran cambio a la hora de usar un navegador web para el envío de trabajos y consulta de la cola sin tener que acceder a la consola del cluster. Trabajos futuros incluyen la cancelación de trabajos, la consulta de la carga en los diferentes nodos y mas información acerca del estado del cluster.

CONTRIBUCIONES

Los resultados de este trabajo de investigación fueron plasmados en los siguientes productos cientifícos:

- León, F., & Diaz, G. Desarrollo de una interfaz web para el envio de trabajos a SLURM. U18- Ideas para transformar el mundo (2019).

- León, F.,& Diaz, G. Development of a web interface for submitting jobs to slurm using CGI C++.Latin America High Performance Computing Conference (CARLA) (2018).

- León, F., & Diaz, G. Development of a web interface for submitting jobs to SLURM. Revista UIS Ingenierías (2019), 18(4), 95-98.

BIBLIOGRAFÍA

ANDJARWIRAWAN, Justinus; PALIT, Henry Novianus; SALIM, Julio Christian. Linux pam to ldap authentication migration. In *2017 International Conference on Soft Computing, Intelligent System and Information Technology (ICSIIT)* IEEE, 2017, pp. 155–159.

BAKER, Mark; BUYYA, Rajkuma. Cluster computing: the commodity supercomputer. *Software: Practice and Experience* 1999. *vol 29*, p 551–576.

BARRIOS, Carlos. Cluster guane. [en línea]. 2019. Disponible en http://wiki.sc3.uis.edu.co/index.php/Cluster%2BGuane.

DIAZ, Gilberto. ¿cómo acceder a los recursos? [en línea]. 2019. Disponible en http://wiki.sc3.uis.edu.co.

Diaz, Sebastian. Cgicc - gnu project - free software foundation. [en línea]. 2016. Disponible en https://www.gnu.org/software/cgicc/index.html.

EADLINE, Douglas. *High Performance Computing for Dummies*. Indianápolis: Wiley Publishing, Inc., 2009.p. 52.

Fischer, Josh. Heron. [en línea]. Agosto 2019. Disponible en https://github.com/apache/incubator-heron.

HOWES, Timothy; SMITH, Mark; GOOD, Gordon S. *Understanding and deploying LDAP directory services*. Addison-Wesley Longman Publishing Co., Inc., 2003.p.p

MAVRIDIS, Ilias; KARATZA, Helen. Performance evaluation of cloud-based log file analysis with apache hadoop and apache spark. *Journal of Systems and Software* 2017, *vol. 125. p.* 133–151.

OFFSET, Byte. Virtual memory management in the vax/vms operating system. *Computer. Vol. 50*, 1982 p. 36.

OSC. Open on demand. [en línea]. Agosto 2019. Disponible en https://openondemand.org/.

ROBERTS, Mark; TORRES, Giovanni. Pyslurm: Slurm interface to python. [en línea]. febrero 2019. Disponible en https://pyslurm.github.io/.

ROBINSON, David. The common gateway interface (cgi) version 1.1. [en línea]. 2004. Disponible en https://tools.ietf.org/html/rfc3875.

ROJAS, Raul. Konrad zuse's legacy: the architecture of the z1 and z3. *IEEE Annals of the History of Computing. Vol. 19*, 1997, p. 5–16.

SANKARANARAYAN, Mukund. et al. Resource manager architecture, Sept. 28 Google Patents, 2004. US Patent 6, 799, 208.

SC3. Proyectos – sc3 uis. [en línea]. 2019. Disponible en http://www.sc3.uis.edu.co/lista-de-proyectos/.

SCHONWALDER, Jurgen, et al. Session resumption for the secure shell protocol. In *2009 IFIP/IEEE International Symposium on Integrated Network Management* IEEE, 2009, pp. 157–163.

SHARMA, Priti; KUMAR, Brijesh; GUPTA, Pooja An introduction to cluster computing using mobile nodes. In *2009 Second International Conference on Emerging Trends in Engineering & Technology* IEEE, 2009, pp. 670–673.

SLURMWEB. Slurm-web. [en línea]. Agosto 2019. Disponible en https://edf-hpc.github.io/slurm-web/introduction.html.

Strohmaier, Erich. et al. Top 500 noviembre 2018. [en línea]. 2018. Disponible en https://www.top500.org/list/2018/11/.

STROHMAIER, Erich; DONGARRA, Jack; SIMON, Horst, et al. Top 500 november 2017. [en línea]. 2017. Disponible en https://www.top500.org/lists/2017/11/.

STROHMAIER, Erich; DONGARRA, Jack; SIMON, Horst, et al. Top 500 november 2005. [en línea]. 2005. Disponible en https://www.top500.org/lists/2005/11/.

STROHMAIER, Erich; DONGARRA, Jack; SIMON, Horst, et al. Top 500 OPERATING SYSTEM FAMILY. [en línea]. 2019. Disponible en https://www.top500.org/statistics/details/osfam/1

STROHMAIER, Erich; et al. Top 500 architecture. [en línea]. 2019. Disponible en https://www.top500.org/statistics/details/architecture/4.

TSAI, Jonh. For better or worse: Introducing the gnu general public license version 3. *Berkeley Tech* 2008*. LJ vol. 23*, p 547.

YOO, A; JETTE, Morris; GRONDONA, Mark Simple linux utility for resource management. In *Workshop on Job Scheduling Strategies for Parallel Processing* Springer, 2003. pp. 44–60.

YOO, Andy; JETTE, Morris; GRONDONA, Mark. Slurm workload manager - overview. [en línea]. 2019. Disponible en https://slurm.schedmd.com/overview.html.

ZEILENGA, Kurt. Lightweight directory access protocol (ldap): Technical specification road map. 2006. p.p