UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y COMUNICACIONES

TEMA:

"CONTROL DE FLUJO DE UNA RED DEFINIDA POR SOFTWARE USANDO SENSORES TÉRMICOS"

Trabajo de Graduación. Modalidad: Proyecto de Investigación, presentado previo la obtención del título de Ingeniero en Electrónica y Comunicaciones.

SUBLÍNEAS DE INVESTIGACIÓN:Teoría, Diseño e Interconexión de Redes.

AUTOR: Jaime Andrés Escobar Ordoñez
TUTOR: Ing. Víctor Santiago Manzano Villafuerte, Mg

Ambato - Ecuador
Octubre, 2015

APROBACIÓN DEL TUTOR

En mi calidad de Tutor del Trabajo de Investigación sobre el Tema:

"CONTROL DE FLUJO DE UNA RED DEFINIDA POR SOFTWARE USANDO SENSORES TÉRMICOS", del señor, Jaime Andrés Escobar Ordoñez estudiante de la Carrera de Ingeniería en Electrónica y Comunicaciones, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial de la Universidad Técnica de Ambato, considero que el informe investigativo reúne los requisitos suficientes para que continúe con los trámites y consiguiente aprobación de conformidad con el numeral 7.2 de los Lineamientos Generales para la aplicación de Instructivos de las Modalidades de Titulación de las Facultades de la Universidad Técnica de Ambato.

Ambato, Octubre de 2015

EL TUTOR

Ing. Víctor Santiago Manzano Villafuerte, Mg

AUTORÍA

El presente Proyecto de Investigación titulado: CONTROL DE FLUJO DE UNA RED DEFINIDA POR SOFTWARE USANDO SENSORES TÉRMICOS. Es absolutamente original, auténtico y personal, en tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, Octubre de 2015

Jaime Andrés Escobar Ordoñez

CC: 1803032356

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato, para que haga uso de este Trabajo de Titulación como un documento disponible para la lectura, consulta y procesos de investigación.
Cedo los derechos de mi Trabajo de Titulación, con fines de difusión pública, además autorizo su reproducción dentro de las regulaciones de la Universidad.

Ambato, Octubre de 2015

Jaime Andrés Escobar Ordoñez

CC: 1803032356

APROBACIÓN COMISIÓN CALIFICADORES

La Comisión Calificadora del presente trabajo conformada por los señores docentes Ingenieros: Patricio Córdova y Juan Pablo Pallo, revisó y aprobó el Informe Final del trabajo de graduación titulado CONTROL DE FLUJO DE UNA RED DEFINIDA POR SOFTWARE USANDO SENSORES TÉRMICOS, presentado por el señor Jaime Andrés Escobar Ordoñez de acuerdo al numeral 9.1 de los Lineamientos Generales para la aplicación de Instructivos de las Modalidades de Titulación de las Facultades de la Universidad Técnica de Ambato.

Ing. José Vicente Morales Lozada, Mg.

PRESIDENTE DEL TRIBUNAL

Ing. Patricio Córdova, Mg.
DOCENTE CALIFICADOR

Ing. Juan Pablo Pallo, Mg.
DOCENTE CALIFICADOR

DEDICATORIA

Este proyecto se lo dedico a mi madre que es el pilar fundamental de mi vida y quién supo guiarme por el buen camino, darme fuerzas para seguir adelante y no permitirme desmayar en cada uno de los problemas que se presentaban, enseñándome a encarar las adversidades sin perder nunca la dignidad ni desfallecer en el intento.

A mi familia por siempre estar presentes y nunca dejar de alentarme en este largo camino, a mis amigos y compañeros que supimos apoyarnos en los buenos y malos momentos sabiendo que siempre podríamos contar unos con otros.

Andrés Escobar

AGRADECIMIENTO

El presente proyecto de graduación es el esfuerzo y trabajo en el que diversas personas han aportado directa e indirectamente siempre opinado, corrigiendo, criticando y apoyando el desarrollo del mismo. Este proyecto me a permitido aprovechar el conocimiento y experiencia de muchas personas a las que me permito agradecer.

A mi tutor de proyecto Ing. Santiago Manzano mi más sincero agradecimiento por haber confiado en el desarrollo de este proyecto, por su inmensa paciencia ante mis equivocaciones, por todas las valiosas aportaciones y apoyo para el desarrollo y culminación de este gran proyecto.

A todos mis maestros, compañeros y amigos a lo largo de la carrera que siempre han sabido aportar a mi desarrollo como profesional pero ante todo a mi desarrollo como persona a todos y cada uno con los que pude compartir el aula de clases y también compartir grandiosos momentos que jamas podre olvidar .

Andrés Escobar

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

Resumen

En el presente proyecto presenta una innovadora propuesta de un sistema para la gestión inteligente del flujo de información apoyado en el uso de las redes definidas por software, SDN. El empleo de sensores, específicamente de temperatura, proporcionará un elevado nivel de eficiencia siendo una de las variables más comunes que se puede incorporar a las redes de datos.

El proyecto plantea un sistema que permitirá el control y monitoreo a distancia de una red de datos y su comportamiento basado en su totalidad en variables externas. El empleo de las redes definidas por software permite la integración de las redes de datos y sistemas electrónicos debido a su amplia apertura para el desarrollo de nuevos módulos y/o topologías de red. El pilar fundamental del proyecto es el uso sistemas tanto de Hardware como Software libre convirtiéndolo así en un sistema escalable al igual que adaptable.

La implementación de una arquitectura de red usando equipos habilitados para trabajar con el protocolo openflow permitió la aplicación de un módulo de balanceo de carga, el cual fue desarrollado para el controlador Floodlight y modificado para trabajar en dependencia a información entregada por la plataforma Arduino. Al ser una arquitectura de red para monitoreo distante, el uso de la herramienta de Tunneling permitió establecer una comunicación segura entre el controlador y la arquitectura de red a través de Internet.

Abstract

This project presents an innovative proposal about a system for the intelligent management of the information's flow supported by the use of software defined networks, SDN. The use of sensors; temperature sensors specifically, will provide a high level of efficiency; being one of the most common variables that can be incorporated into the data networks.

The project proposes a system that will allow the control and remote monitoring of a data network and its behavior based on external variables in a whole. The use of software defined networks allows the integration of data networks and electronic systems due to its wide opening to the development of new modules and/or network topologies. The mainstay of the project is the use of both, free Hardware and Software systems; turning them into a scalable and adaptable system.

The implementation of an architecture network using enabled equipment to work with the "openflow" protocol, allowed the implementation of a charge balancing module; which was developed for the Floodlight controller and modified to work in dependence on the information provided by the Arduino platform. As an architecture network for remote monitoring, the use of the Tunneling tool allowed to establish a secure communication between the controller and the architecture network through Internet.

Glosario de Términos y Acrónimos

Términos

- Acceso remoto. Conectarse a una red desde una ubicación distante.
- Protocolo. Es un conjunto de reglas que definen cómo interactúan las entidades de comunicación. Para que una computadora se pueda comunicar con otra se requieren de varios protocolos los cuales van a definir las reglas de la comunicación.
- Plano de Control. Uno de los 3 elementos básicos de una arquitectura de red que lleva el tráfico de señalización y es responsable de enrutamiento, configuración del sistema y gestión de la red.
- Plano de Información. Uno de 3 elementos básicos de una arquitectura de red que transporta el tráfico de usuario.
- Tunneling. El tunneling es un método utilizado para encapsular paquetes (conocidos como datos de usuario) dentro de otros paquetes los cuales son enviados utilizando la tecnología de la red por la que viaja.
- Virtualización. Recurso que significa crear una versión virtual que no es el propio dispositivo físico, pero ambos se comporta y aparece al usuario como el dispositivo real.
- VPN. Es una red privada que utiliza la infraestructura de una red pública para poder transmitir información.

Acrónimos

- ARP. Address Resolution Protocol (Protocolo de Resolución de Direcciones).
- GRE. Generic Routing Encapsulation (Encapsulación Genérica para Ruteo).
- IDC. International Data Corporation (Corporación Internacional de Información).
- IP. Internet Protocol (Protocolo de Internet).
- IPSec. Internet Protocol Security (Seguridad del Protocolo de Internet).
- MAC. Media Access Control (Control de Acceso al Medio).
- PPP. Point to Point Protocol (Protocolo Punto a Punto).
- SDN. Software Defined Networl (Red Definida por Software).
- WSN. Wireless Sensor Network (Red de Sensores Inalambricos).

INTRODUCCIÓN

En la Actualidad la posibilidad de integrar redes de datos con dispositivos electrónicos no existe, el presente proyecto presenta la posibilidad de dicha integración apoyado en la implementación de redes definidas por software y sistemas electrónicas basados en su totalidad en hardware libre. El trabajo se organiza de la siguiente forma.

Capítulo I. Se analiza la problemática actual de las redes de datos teniendo un especial énfasis en problemas que las altas temperaturas pueden llegar a ocasionar, así como la debida justificación para el desarrollo del proyecto y se establecen los objetivos a cumplir en este proyecto.

Capítulo II. Se realiza un análisis sobre el estado actual de las redes SDN y los avances que estas han tenido a lo largo de los últimos años, se abordan los distintos conceptos necesarios para el completo entendimiento del proyecto y se presenta una propuesta de solución al problema descrito anteriormente.

Capítulo III. Se describe los distintos aspectos de la metodología con la que se elaborara el proyecto y las diversas etapas por las que a travesara.

Capítulo IV. Se presenta el desarrollo completo del proyecto con una detallada descripción de las partes importantes que tiene el proyecto.

Capítulo V. Se presentan las conclusiones y recomendaciones que se obtuvieron al finalizar el proyecto.

CAPÍTULO 1

El problema

1.1. Tema de Investigación

Control de flujo de una red definida por software usando sensores térmicos.

1.2. Planteamiento del problema

Las redes actuales se concibieron de modo que el control, enrutamiento y señalización de la red fueran parte de una misma arquitectura, con una gestión jerárquica, estática y así mismo dependiente de la infraestructura que posea la red. Actualmente el principal problema que presentan las redes de datos es no poder realizar cambios de una manera dinámica, ya sea por requerimientos de tráfico o por requerimiento de servicios dentro de la red, esto implicaría intervenir en cada uno de los dispositivos para programar rutinas, cambiar enrutamientos o agregar servicios[5].

Las redes tradicionales en su mayoría ofrecen servicios de uso cotidiano como el envío de correo electrónico, tecnologías de búsqueda, redes sociales, comercio electrónico y recientemente servicios de Cloud Computing mismos que han impulsado al análisis y re-estructuración dentro de la industria del networking. Las exigencias actuales como el alto número de peticiones, replicación o virtualización de servicios generan problemas para la mayoría de dispositivos de red tradicionales en aspectos como capacidad, cambios en los patrones de tráfico que circula por la red y el evidente desarrollo de aplicaciones de alto rendimiento basadas en virtualización, siendo estas las principales limitaciones que presentan los equipos actuales causado por un crecimiento acelerado de las redes de alto rendimiento[6].

Actualmente no existe la posibilidad de integrar redes de datos con dispositivos electrónicos (sensores o transductores) y que en dependencia del estado de estos dispositivos electrónicos sea posible implementar cambios en el comportamiento de los equipos de red.

1.3. Delimitación

Área académica: Programación y Redes.

Línea de investigación: Programación y Redes.

Sublíneas de investigación: Teoría, diseño e Interconexión de Redes.

Delimitación espacial: La investigación se realizará en la Ciudad de Ambato.

Delimitación temporal: El proyecto de investigación se desarrollará en un plazo de seis meses a partir de la aprobación del Honorable Consejo Directivo de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial.

1.4. Justificación

Según datos de la International Data Corporation (IDC) el mercado de las redes definidas por software tendrá un gran avance dentro de los próximos años con un crecimiento anual de un 89 %. Para el 2018 el negocio de las redes SDN tendrá un incremento de los 960 millones de dólares previstos para 2014 a 8000 millones de dólares, esta cantidad que pronostica el IDC abarca la infraestructura física, el software de control de las redes virtualizadas y los servicios de seguridad añadidos así como las aplicaciones y otros servicios.

Las redes definidas por software también plantean un reto que involucra a las tecnologías de la tercera plataforma como son el cloud, movilidad, Big Data, Internet de las cosas, las cuales irán consolidando el sector de las redes SDN. El IDC plantea que el periodo que va desde 2014 a 2016 será solo el inicio para el mercado de las redes definidas por software en las empresas [7].

En nuestro país las redes definidas son una tecnología parcialmente nueva por lo cual distintas instituciones han adoptado a las redes SDN como tema de investigación y desarrollo. Aunque la implementación de redes definidas por software más allá de los laboratorios son muy pocas, Google es una de las empresas pioneras en el tema. En el Open Networking Summit de 2013 Google presento detalles sobre su red SDN, la cual estaría dedicada a la interconexión de sus Data Centers así como la interconexión dentro de la misma empresa[8].

El desarrollo del presente proyecto tendrá un alto impacto para todos los usuarios de redes de comunicaciones ya que SDN puede ser implementado en una infinidad de redes y equipos gracias a todos los beneficios que ofrece. El aspecto principal del proyecto es la integración de dos tecnologías como son los dispositivos electrónicos como sensores con las redes de datos, con el fin de tener una red mucho más dinámica y a la vez autónoma. El proyecto presenta distintos niveles de dificultad destacando la complejidad de integrar ambas tecnologías, sin embargo utilizando los recursos adecuados es posible obtener resultados favorables abriendo así nuevas áreas de trabajo, investigación y desarrollo.

El proyecto involucrará tanto software y hardware libre otorgándole al proyecto una mayor aplicabilidad y escalabilidad, convirtiéndose en uno de los aspectos relevantes del proyecto. El desarrollo de aplicaciones en redes definidas por software permitirá ofrecer servicios tan diferentes de los actuales, entre ellos el Internet que ha seguido siendo el mismo de hace 25 años, permitiéndole a la industria innovar a corto plazo.

1.5. Objetivos

1.5.1. General

Desarrollar un control de flujo para una red definida por software utilizando sensores térmicos

1.5.2. Específicos

- Determinar los niveles máximos y mínimos de temperatura dentro de los equipos de la red SDN.
- Determinar un conjunto de nuevos flujos basados en los niveles térmicos de los equipos de red SDN.
- Diseñar una aplicación para la visualización e inserción de reglas de direccionamiento dentro del controlador.

CAPÍTULO 2

Marco Teórico

2.1. Antecedentes Investigativos

En el año 2014 Alejandro de Gante en su artículo universitario denominado "Smart Wireless Sensor Network Management Based on Software Defined Networking" sostiene que la gestión inteligente de redes utilizando SDN promete una solución para algunos de los problemas inherentes a la gestión de redes de sensores inalámbricos (WSN).Además, propone una arquitectura genérica para una estación base en una red de sensores inalámbricos definida por software. También propone un marco general para una red inalámbrica de sensores definida por software donde el controlador se implementa en la estación base [9].

En el año 2012 Ana Milena Rojas Calero en su artículo universitaria bajo el título "Software Defined Networks" concluye que el futuro de las redes se basará cada vez más en software. SDN promete transformar las redes estáticas de hoy en plataformas flexibles y programables con la inteligencia para asignar recursos de forma dinámica, virtualizar servicios y optimizar las capacidades de cómputo IT y de infraestructura de red. Está en camino de convertirse en la nueva arquitectura para las redes [5].

Diana Gabriela Morillo Fueltala en el año 2014 dentro de su tesis universitaria denominada "Implementación de un prototipo de una red definida por software empleando una solución basada en software", concluye que el futuro de las redes dependerá cada vez más del software, el cual acelera el ritmo de innovación de las mismas, por ello se considera a SDN como la solución capaz de transformar las redes estáticas actuales y se alienta a seguir con el desarrollo de aplicaciones que cubran las necesidades del usuario en cuanto a la movilidad, manejo de servicios en la nube y seguridad [10].

Pablo Andrés Guamán Guachichullca en su tesis universitaria en el año 2013 concluye que SDN rompe barreras en el modo de trabaja haciendo en un 100 % fiable y flexible el manejo de las redes que por mucho tiempo han sufrido de inconvenientes ya sean estos por inseguridad, por problemas en velocidad en redes inalámbricas. Esta tecnología permite que miles de dispositivos estén conectados entre sí sin importar que los equipos sean físicos o virtuales, estén o no en la nube [11].

Juan Carlos Chico Jiménez en su tesis bajo el título "Implementación de un prototipo de una red definida por software empleando una solución basada en hardware", concluye que SDN permite disociar la infraestructura de la red de aplicaciones y servicios presentes en ella, los cuales ven a la red como una entidad lógica permitiendo así tener redes más escalables y flexibles que se adaptan fácilmente [12].

Carlos Arteche Gonzáles en el año 2014 dentro de su tesis universitaria titulada "Despliegue de una maqueta de red basada en openflow" concluye que el concepto de Software Defined Networking (SDN) está ganando cada vez mayor popularidad. El crecimiento de los data centers y la necesidad de disponer de tecnologías de red que aseguren un comportamiento más adecuado por parte de las redes es ya una realidad y los protocolos de encaminamiento y conmutación que se han venido usando hasta el momento no parecen los más apropiados para dar respuesta a los retos que están apareciendo [13].

En el año 2015, Alex Núñez Ramírez en su tesis de ingeniería bajo el título "Red definida por software (SDN) en base a una infraestructura de software de libre distribución" concluye que el surgimiento de las redes definidas por software se debe a la incapacidad de las redes convencionales de permitir cambios en los patrones de tráfico de forma dinámica, mediante la adición, eliminación o modificación de reglas de flujo en los dispositivos de interworking que soporte OpenFlow[3].

2.2. Fundamentación Teórica

2.2.1. Redes Definidas por Software

Introducción

Las empresas y los proveedores de servicios actualmente se encuentran buscando soluciones para los retos a los que se enfrentan en el terreno de las redes de datos. Necesitan que las redes se ajusten y respondan de forma dinámica en función de su política de negocio. Asimismo, necesitan que estas políticas estén actualizadas, con el fin de poder reducir el trabajo manual y los costes asociados con la administración de sus redes.

Hoy en día uno de los más grandes retos es implementar y ejecutar con rapidez nuevas aplicaciones dentro de sus redes de datos, con el fin de poder ofrecer resultados de negocio. A la vez intentan introducir estas nuevas funciones sin que afecte al negocio. Sin embargo, las redes SDN parecen ser la respuesta a estos retos [8].

Las redes SDN surgen como una alternativa viable de las redes actuales, todo ocasionado por la evidente necesidad de tener redes con una mayor escalabilidad y un tráfico de datos más personalizado y eficiente, ya que la capacidad y eficiencia de las redes actuales se ha visto altamente afectado por la cantidad y tipo de tráfico que manejan [12].

Definición

Las redes definidas por Software son una arquitectura de red dinámica, manejable, rentable y adaptable. Esta arquitectura desacopla el control de la red y las funciones de reenvío permitiendo que el control de la red pase a ser directamente programable y empleado la infraestructura específicamente para las aplicaciones y servicios de red [14].

El protocolo OpenFlow es un elemento fundamental para la construcción de soluciones SDN. La arquitectura SDN presenta diversas características como:

- Directamente programable: control de la red se puede programar directamente ya que está desacoplada de las funciones de reenvío.
- Ágil: Permite a los administradores ajustar dinámicamente el flujo de tráfico de toda la red para satisfacer las necesidades cambiantes.
- Gestiona de forma centralizada: la inteligencia de la red es (lógicamente) centralizado en software basado en controladores SDN que mantienen una

visión global de la red, que aparece a aplicaciones y motores de políticas como una sola, switch lógico.

- Programación configurada: SDN permite a los administradores de red configurar, administrar, proteger y optimizar los recursos de red muy rápidamente a través de los programas de SDN dinámicos, automatizados, que pueden escribir ellos mismos porque los programas no dependen de software propietario.

- Basada en estándares abiertos y el vendedor-neutrales: Cuando se implementa a través de estándares abiertos, SDN simplifica el diseño de la red y la operación porque las instrucciones son proporcionados por controladores SDN en lugar de múltiples dispositivos, específicos del fabricante y los protocolos [14].

Principios de SDN

Estos son los seis principios de las SDN y los correspondientes beneficios que ofrecen para los clientes [8]:

1. Separan de forma eficiente el software de red en cuatro niveles (planos): administración, servicios, control y reenvío. Esto ofrece el apoyo estructural necesario para optimizar cada plano dentro de la red.

2. Centralizan los aspectos necesarios de los planos de administración, servicios y control para simplificar el diseño de red y para reducir los costes operativos.

3. Utilizan la nube para una implementación flexible y con una escala adaptable, que permite una estructura de precios basada en el uso con el fin de reducir el tiempo de servicio y establecer una correlación entre costes y valor.

4. Crean una plataforma para las aplicaciones de red, los servicios y la integración en los sistemas de administración, lo que permite el desarrollo de nuevas soluciones de negocio.

5. Permiten una estandarización de los protocolos, lo que hace posible disponer de asistencia interoperativa y heterogénea entre proveedores, y que da lugar a la posibilidad de elección y de reducción del coste.

6. Aplican con una amplia perspectiva los principios de las SDN a todos los servicios de red, incluidos los de seguridad. Esto abarca desde el centro de datos y el campus empresarial hasta las redes móviles e inalámbricas utilizadas por los proveedores de servicio.

Los cuatro planos de redes

Dentro de cada dispositivo de red y de seguridad (conmutadores, enrutadores y firewall), es posible separar el software en cuatro niveles o planos. A medida que se avanza más hacia las SDN, es necesario diferenciarlos y separarlos con claridad. Se trata de algo esencial para el desarrollo de la siguiente generación de redes altamente escalables. Los planos de red están conformados por plano de Administración, Servicios, Control y Forwarding de los cuales se detalla a continuación:

- Forwarding: El plano inferior, el de forwarding, se encarga de trabajo duro en lo relativo al envío de paquetes de red. Está optimizado para mover los datos lo más rápido posible. El plano de forwarding puede implementarse en el software, aunque suele desarrollarse sobre circuitos integrados específicos para aplicaciones, que están diseñados para este propósito.

- Control: Si se contempla el plano de forwarding como los músculos de la red, el control sería el cerebro. El plano de control analiza la topología de red y toma decisiones sobre el destino del flujo del tráfico de red. El plano de control es el policía de tráfico que comprende y descifra los protocolos de red, asegurándose que el flujo de tráfico sea continuo.

- Servicios: En algunos casos, el tráfico de red requiere más procesamiento. En estos casos, el plano de servicios se encarga del trabajo. Se encarga de llevar a cabo las operaciones complejas con los datos de red que el hardware de reenvío no puede llevar a cabo.

- Administración: Al igual que todos los ordenadores, los dispositivos de red tienen que configurarse o gestionarse. El plano de gestión proporciona las instrucciones básicas sobre cómo debe interactuar el dispositivo de red con el resto de esta. Aunque el panel de control no puede aprender todo lo que necesita de la propia red, el plano de gestión necesita que se le indique qué tiene que hacer [8].

2.2.2. Protocolo OpenFlow

Una de las bases del concepto de SDN es el protocolo OpenFlow, propuesto por miembros de diversas universidades, como Stanford, Berkeley y MIT. OpenFlow se presenta como un lenguaje abierto y universal para la "comunicación" entre elementos de red, a partir de la creación de tablas de flujo dinámicas.

Actualmente, muchos equipos ya son compatibles con el protocolo OpenFlow, popularizando y viabilizando la expansión de las redes definidas por software.

Como alternativa al protocolo OpenFlow, diversos fabricantes crearon, en el 2013, una asociación llamada Open DayLight. Misma que recibe un gran apoyo por parte de Linux Foundation, se trata de un proyecto colaborativo cuyo objetivo principal es crear un controlador de red estandarizado, evitando así la fragmentación de los protocolos y softwares de redes, una de las principales preocupaciones a la hora de hablar de redes abiertas y programables [6].

OpenFlow se define como un protocolo de comunicaciones abierto, siendo la primera interfaz de comunicaciones estándar entre los planos de control y una arquitectura de red SDN. El protocolo OpenFlow permite el acceso y manipulación del plano de reenvío de datos en dispositivos como switches y routers. Abriendo la posibilidad de brindar un control centralizado separándolo de los dispositivos de red[6] .

2.2.3. Tipos de Switches OpenFlow

Actualmente es posible encontrar una amplia gama de dispositivos que soporten el protocolo OpenFlow, pero que se divide en dos tipos particulares al momento de utilizar el Protocolo OpenFlow siendo estas:

- Switches OpenFlow-Dedicados

  Este tipo de equipos únicamente soportan el protocolo OpenFlow es decir que trabajan únicamente con las directrices que emite el controlador de red. Al depender únicamente de OpenFLow carecen de cualquier tipo de característica de control propia a diferencia de otros equipos.

  Los switch dedicados pueden contener múltiples tablas de flujos y que a su vez contienen múltiples entradas de flujos donde el procesado de la información por parte del controlador de la red definirá la forma en que los paquetes de datos interactuarán con las tablas de flujo, siendo necesario que el dispositivo contenga por lo menos una tabla de flujo y tomando en cuenta que entre menos entradas tenga la tabla el proceso será mucho más simplificado.

- Switches OpenFlow-Híbridos

  Este tipo de equipos son capaces de soportar el protocolo OpenFLow y a la vez tener tocas y cada una de las funcionalidades de un Switch convencional es decir que podrá trabajar como un dispositivo de capa 2 o capa 3.

Estos equipos son capaces de separar la información que se procesara gracias a OpenFlow como la información que se procesara con sus lineamientos nativos, inclusive pueden separar de forma total el funcionamiento de OpenFLow y operando de forma paralela como un switch convencional [12].

- Aplicaciones que impulsan las redes SDN

  Dentro de SDN se destacan ciertas aplicaciones que impulsaran el desarrollo de este tipo de redes, estas aplicaciones son:

  - Cloud Computing

    Diferente del mundo cliente-servidor, en que la comunicación es bidireccional entre dos puntos, en el mundo Cloud las aplicaciones necesitan acceder a múltiples bancos de datos y servidores, generando tráfico en múltiples sentidos y ampliando la complejidad del ambiente.

  - Movilidad

    No resulta una novedoso Tipos de Switches OpenFlow Actualmente podemos encontrar aunque los usuarios corporativos estén demandando más movilidad, exigiendo la posibilidad de acceder a sistemas e informaciones en cualquier momento, en cualquier lugar y usando cualquier dispositivo personal. Este cambio de comportamiento de los usuarios está alterando también los patrones de tráfico en las redes.

  - Big Data

    El gerenciamiento, almacenamiento y acceso a los datos corporativos viene cambiando significativamente en los últimos años. Al mismo tiempo en que el volumen de informaciones aumentó exponencialmente, las arquitecturas de los Data Warehouses está cambiando, adaptándose a las necesidades y a las nuevas tecnologías. Esos cambios también demandan alteraciones en la arquitectura de las redes y en el standard de enrutamiento de los datos.

  - Internet of Things

    Las llamadas “Redes de sensores” y/o “Redes de comunicación entre máquinas”, que darán origen a Internet de las Cosas (IoT, en la sigla en inglés), exigen que la latencia de las redes sea la menor posible, la disponibilidad debe ser cercana a 100 % y que los sistemas de seguridad no interfieran en las operaciones normal. En ese contexto, características como disponibilidad, confiabilidad, flexibilidad y seguridad son fundamentales [6].

2.2.4. Controladores SDN

El controlador es el núcleo central de la arquitectura SDN, ya que es el que tiene toda la lógica de la red. En el controlador se definen las reglas para administrar el flujo de datos en la red. Esto permite una configuración más rápida que en las redes tradicionales, donde se espera a que el fabricante lo haga, al igual que la implementación de nuevas aplicaciones y servicios[13].

El controlador realiza la función de una capa de abstracción de la infraestructura física, lo que facilita la creación de aplicaciones y servicios que gestionan la red de flujos de entradas. Este modelo es similar al de otros sistemas de software que proporcionan abstracción de hardware y funcionalidad reutilizable[15][3].

NOX

Hay un gran número de SDN controladores de código abierto que han estado disponibles desde hace algún tiempo. El primer controlador OpenFlow altamente popular fue NOX. Al igual que con la mayoría de las implementaciones de tecnología temprana, fue la chispa inicial, pero no fue fuertemente implementado o utiliza debido a deficiencias en su entorno la aplicación y el desarrollo; mucho de esto tiene que ver con NOX está programando principalmente en C++ y la falta de una buena documentación[16].

POX

El sucesor de NOX, POX, fue construido como una alternativa más amigable y se ha utilizado y aplicado por un número de desarrolladores SDN e ingenieros. En comparación con los NOX, POX tiene un entorno de desarrollo más fácil de trabajar y una API y documentación razonablemente bien escrito. También proporciona una interfaz gráfica de usuario basada en web y está escrito en Python, que normalmente reduce sus ciclos experimentales y de desarrollo[17].

Beacon

El siguiente gran paso en los controladores de código abierto viene con Beacon que es un controlador SDN muy bien escrito y organizado escrito en Java y altamente integrado en el IDE de Eclipse. Beacon fue el primer controlador que hizo posible que los programadores principiantes trabajen y creen un entorno SND, sin embargo, se limitó a las topologías estrella (sin bucles)[18].

Floodlight

Si bien sus inicios se basó en Beacon fue construido usando Apache Ant que es una herramienta de construcción de software muy popular que hace que el desarrollo de Floodlight más fácil y flexible. Floodlight tiene una comunidad muy activa y tiene un gran número de características que se pueden agregar para crear un sistema que mejor se adapte a las exigencias de una organización específica. La mayor parte de su funcionalidad se expone a través de una API REST[19].

2.2.5. Sensores de Temperatura

Una gran cantidad de procesos requieren el control preciso de la temperatura para producir resultados de calidad o prevenir sobrecalentamientos, rupturas, explosiones y otros tipos de problemas. Las temperaturas elevadas, por ejemplo, son necesarias para ablandar metales y fundir plásticos antes de ser moldeados en formas específicas. Asimismo, las temperaturas bajas son necesarias para conservar productos en una industria procesadora de alimentos.

En la actualidad hay muchas formas de medir temperatura con todo tipo de sensores de diversas naturalezas. La ingeniería de control de procesos ha inventado, perfeccionado e innovado a la hora de disponer de sensores que les ayuden a controlar los cambios de temperatura en procesos industriales. La tabla 2.1 muestra la gran variedad de dispositivos capaces de medir la temperatura[1]:

DISPOSITIVOS DE MEDICIÓN DE TEMPERATURA			
Eléctricos	Mecánicos	Radiación Térmica	Varios
Termocuplas	Sistemas de Dilatación	Pirómetro de radiación	Indicadores de color
Termo Resistencias	Termómetros de	-Total (banda-ancha)	-Lápices
Termistores	vidrio con líquidos	-Óptico	-Pinturas
Sensores de Silicio	Termómetros bimetálicos	-Pasa Banda	Sondas neumáticas
con efecto resistivo		-Relación	Sensores ultrasónicos
		Termómetros Infrarrojos	Indicadores pirométricos
			Termómetros acústicos
			Cristales Líquidos

Tabla 2.1: Dispositivos de medición de Temperatura [1].

Cada proceso en la industria debe ser controlado de alguna manera, y esta necesidad incluye la medición de la temperatura. A fin de seleccionar el mejor sensor, para cada aplicación, se deben tener en cuenta varios factores:

- Temperatura Máxima
- Rango de Temperatura a medir
- Exactitud
- Velocidad de respuesta
- Costo
- Requerimiento de mantenimiento.

Los sensores de uso más frecuente, en las industrias de procesos son Termocuplas, Termo resistencias, Termistores, Sistemas de dilatación; a continuación se describen algunos sensores de temperatura con los distintos rangos de trabajo. Estos rangos no representan los extremos alcanzables, sino los límites que pueden medirse con los dispositivos disponibles, por lo general son suministrados por la mayoría de los fabricantes. Se pueden medir mayores y menores temperaturas pero generalmente con una exactitud menor y un mayor costo [1]. La tabla 2.2 detalla los rangos de operación de algunos sensores de temperatura.

Sensor de Temperatura	*Temp. Mínima*	*Temp. Máxima*
Termocuplas	-220 ºC	2800 ºC
Sistemas de Dilatación	-195 ºC	760 ºC
Termo Resistencias	-250 ºC	850 ºC
Termistores	-195 ºC	450 ºC
Pirómetros de Radiación	-40 ºC	4000 ºC

Tabla 2.2: Sensores de temperatura Rangos de Operación [1]

Los sensores de temperatura de precisión, proporcional la temperatura en grados centígrados con una tensión de salida lineal. Así tienen una ventaja sobre los sensores de temperatura lineal calibrados en grados Kelvin, ya que no se requiere una normalización para obtener la temperatura en grados centígrados.

Estos sensores no requiere ninguna calibración externa o recorte para proporcionar precisiones típicas de ± ¼ ° C a temperatura ambiente y ± ¾ ° C durante un rango de -55 ° C a + 150 ° C de temperatura. presentan una baja impedancia de salida, salida lineal, y la calibración inherente permite una interconexión a circuitos de control especialmente fácil[20].

2.2.6. Plataformas de Hardware Libre para Adquisición de Información

Arduino

Arduino es una plataforma electrónica de código abierto (open-source) basada en hardware y software flexibles. Arduino está basado en un microcontrolador AVR, en particular el ATmega8, ATmega 168, ATmega328 y el ATmega1280. Los programas de Arduino están basados en C/C++ y compilados con avr-gcc y enlazado con la Libc de AVR de código abierto. En la figura 2.1 se muestra un ejemplo de la plataforma Arduino.

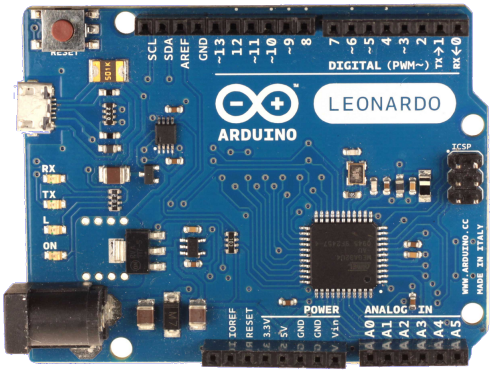


Figura 2.1: Arduino Leonardo [4].

Arduino actualmente posee una amplia variedad de sensores y permite el control de luces, motores y otros artefactos. El microcontrolador de la placa se programa usando el Arduino Programming Language (basado en Wiring) y el Arduino Development Environment (basado en Processing). Los proyectos de Arduino pueden ser autónomos o se pueden comunicar con software en ejecución en un ordenador (por ejemplo con Flash, Processing, MaxMSP, etc.). Las placas se pueden ensamblar a mano o encargarlas pre ensambladas; el software se puede descargar gratuitamente. Los diseños de referencia del hardware (archivos CAD) están disponibles bajo licencia open-source, por lo que eres libre de adaptarlas a tus necesidades. Arduino recibió una mención honorífica en la sección Digital Communities del Ars Electronic Prix en 2006[4].

Raspberri Pi

El Raspberry Pi es un computador de bajo costo con un tamaños similar al de una tarjeta de crédito que se conecta a un monitor o un televisor, y utiliza un teclado y un ratón estándar. El Raspberry Pi que se muestra en la figura 2.2 es capaz de hacer

todo lo que se espera de una computadora de escritorio, desde navegar por Internet y reproducir de vídeo de alta definición, hacer hojas de cálculo, usar procesadores de texto, y jugar juegos[21].

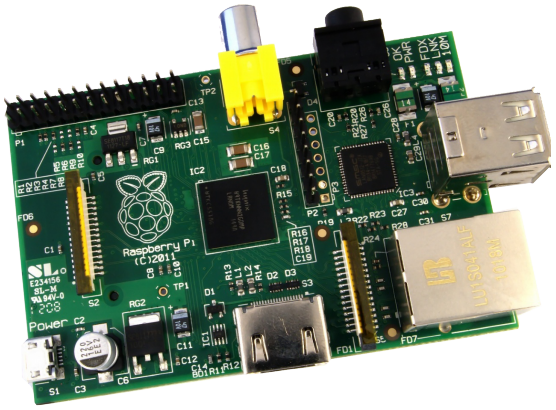


Figura 2.2: Raspberri Pi 2

Al poder instalar Linux se puede utilizar Raspberry Pi para casi cualquier proyecto en el que no se necesite una gran potencia de procesador. Pese a ello lleva un procesador ARM a 700MHz y una GPU capaz de decodificar vídeo a 1080p y compatible con OpenGL. La belleza de la Raspberry Pi es que es sólo un muy pequeño ordenador de propósito general que puede ser un poco más lento de lo que estamos acostumbrados para algunas aplicaciones de escritorio, pero mucho mejor en otras cosas que un PC normal[22].

Intel Galileo

Galileo es una placa electrónica basada en el procesador de aplicación Intel Quark SoC X1000, siendo un sistema de clase Pentium Intel de 32 bits en un chip, también es la primera plataforma basada en la arquitectura Intel diseñada para ser hardware y software compatible con las Arduino Shields diseñadas para el modelo Uno R3[23]. La figura 2.3 muestra la plataforma Galileo de segunda generación.

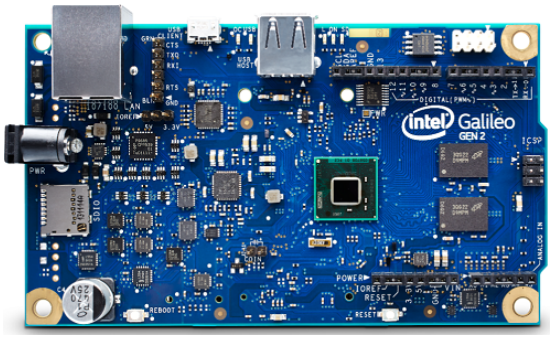


Figura 2.3: Intel Galileo Gen 2

Además posee la compatibilidad con el hardware y software de Arduino, el procesador Intel Galileo tiene varios puertos I/O estándar y características para expandir el uso y la capacidad nativa. Lo que el Galileo intenta hacer es fusionar la facilidad de manipulación de hardware de Arduino con el poder de un sistema operativo Linux en pleno funcionamiento. La mayoría de los sketches escritos para Arduino Uno, Leonardo, y otros se pueden trasladar directamente a la Galileo. Posee acceso a populares bibliotecas Arduino como SD, Ethernet, WiFi, EEPROM, SPI, y Wire, pero también se puede acceder a la parte de Linux de la placa que es compatible con cosas como Python , Node.js , SSH, Telnet, y otros[24].

2.2.7. Sistemas de Gestión de Contenidos

Un sistema de gestión de contenidos es un software que realiza un seguimiento de cada pieza de contenido en un sitio web, al igual que una biblioteca pública local mantiene un registro de los libros y las almacena. El contenido puede ser texto simple, fotos, música, vídeos o documentos. Una ventaja importante de usar un CMS es que prácticamente no requiere habilidad o conocimiento para crear sitios web. Desde el CMS administra todo el contenido, el usuario no tiene que hacerlo[25].

En el transcurso de la web 1.0 a la web 2.0 surgió la necesidad de tener herramientas que permitiesen a los usuarios de internet poder publicar contenidos sin necesidad de tener conocimientos de HTML, CSS, lenguajes de programación, o bases de datos. Uno de los elementos que definen la web 2.0 es la participación ciudadana en la creación de dichos contenidos, y los gestores de contenidos son las herramientas que han logrado esta realidad[26].

Las labores de gestión de contenidos se pueden delegar o compartir con los usuarios. Basta con asignarles un perfil determinado (editor, administrador, autor, etc.). El CMS proporciona la infraestructura necesaria para que se comuniquen y garantiza la consecución del ciclo de trabajo. Un CMS simplifica la creación y mantenimiento de una página web al poder adaptarse a cada tipo de aplicación[27].

En la actualidad existen diversos gestores de contenidos y a continuación se detallan algunos de los gestores más destacados :

Joomla

¿Qué es Joomla?

Joomla es un sistema de gestión de contenido premiado (CMS), que permite construir sitios Web y aplicaciones en línea potentes. Muchos aspectos, incluyendo su facilidad de uso y extensibilidad, Joomla han hecho el software del sitio Web más popular disponible. Lo mejor de todo, Joomla es una solución de código abierto que está disponible gratuitamente.

¿Cuáles son algunos ejemplos del mundo real de lo que Joomla! puede hacer?

Joomla es utilizado en todo el mundo a los sitios Web de potencia de todas las formas y tamaños. Por ejemplo:

- Sitios Web o portales corporativos.
- Intranets y extranets corporativas.
- Revistas, periódicos y publicaciones online.
- El comercio electrónico y reservas en línea.
- Aplicaciones de Gobierno.
- Los sitios web de pequeñas empresas.
- Los sitios web de la organización sin ánimo de lucro.
- Los sitios web de la escuela y la iglesia.
- Páginas personales o familiares.

Joomla está diseñado para ser fácil de instalar y configurar, incluso sin ser un usuario avanzado. Muchos servicios de alojamiento web lo ofrecen un solo clic en instalar, conseguir su nuevo sitio en total funcionamiento en pocos minutos.

Joomla es tan fácil de usar, como diseñador web o programador, se puede crear rápidamente sitios. Luego, con una cantidad mínima de instrucción, puede potenciar y manejar fácilmente sus propios sitios. Joomla es altamente extensible y existen miles de extensiones que están disponibles de forma gratuita bajo la licencia GPL en el Directorio de Joomla [25].

Drupal

Drupal es un gestor de contenido de código abierto que nació en enero de 2001 de la mano de Dries Buytaert y Hans Snijder, dos estudiantes de la Universidad de Amberes. El nombre de Drupal, pronunciado en inglés "Droo-puhl", deriva de la pronunciación inglesa de la palabra flamenca "druppel" que significa arrastrar. Inicialmente nació en el año 2000 como intento de red social para compartir ficheros e ideas entre grupos de estudiantes y no fue hasta 2001 cuando derivó en lo que es ahora, gracias al desarrollo y colaboración de la comunidad[28].

Principalmente Drupal es utilizado en configuraciones Apache, MySQL y PHP, pero puede ser instalado en otros tipos de servidores Web tales como IIS o Nginx y utilizar PostgreSQL o SQLite como motores de bases de datos. Existen módulos para la versión 7 de Drupal que permiten utilizar otros motores de base de datos tales como Microsoft SQL Server y Oracle.

¿Quiénes lo desarrollan?

Drupal es un proyecto de código abierto que en la actualidad cuenta con más de 20.000 desarrolladores que trabajan voluntariamente en los diferentes módulos, temas y la distribución base o núcleo. El desarrollo está centralizado en el sitio Drupal.org donde se publican y actualizan todos los proyectos. Drupal.org es gestionado por la Asociación Drupal, quien por medio de membresías, auspiciantes y eventos financia el funcionamiento de Drupal.org y múltiples actividades en torno a Drupal[29].

Características a destacar.

Aquí algunas de las principales características que posee Drupal:

- Drupal es muy flexible y permite desarrollar portales en un tiempo mínimo.
- Existen miles de módulos para implementar las funcionalidades deseadas.
- La comunidad Drupal es la base del suceso de este CMS.
- Drupal implementa características de vanguardia en funcionalidad y diseño.
- Es un software seguro, extensible y escalable. - Drupal cumple con los estándares actuales de diseño web.

WordPress

WordPress es una avanzada plataforma de publicación personal orientada a la estética, los estándares web y la usabilidad. WordPress es el sistema que utilizas cuando deseas trabajar con tu herramienta de publicación en lugar de pelearte con ella. No es el programa más fácil pero si muy completo y permite un nivel de personalización y configuración muy elevado. Permite proteger los mensajes con contraseña, agregar usuarios con distintas funciones, control de comentarios, gran variedad de plantillas, etc[30].

WordPress posee diversas características que se detallan a continuación:

1. Es fácil de aprender.
2. Es estable.
3. Se usa por millones de personas en todo el mundo.
4. Escala bien.
5. Es muy sencillo, pero flexible.
6. Desarrolladores encanta trabajar con él.
7. Tiene una amplia gama de temas libres y premium que pueden añadir funcionalidad y estilo a su sitio.
8. Cuenta con una amplia selección de plugins que se pueden añadir nuevas características y trucos para su sitio.
9. Si lo hace aprender html, css o php se puede ampliar lo que tienes.

Para su instalación es necesario disponer de PHP y MySQL en el servidor. Estos servicios están disponibles en el Servicio de Alojamiento de Páginas Web de la Consellería de Educación[31].

2.2.8. Virtual Private Networks (VPN)

Una red privada virtual (VPN) permite el uso de servicios de red privados para una organización u organizaciones a través de una infraestructura pública o compartida, como la red principal del proveedor de Internet. El proveedor de servicio de red backbone se conoce como la columna vertebral VPN y se utiliza para transportar el tráfico de datos para múltiples VPNs.

Las VPNs usan tecnologías como Frame Relay y modo de transferencia asíncrono (ATM) y han estado disponibles desde hace mucho tiempo, pero en los últimos años

IP e IP/Multiprotocol Label Switching (MPLS) basadas en VPNs se han vuelto más y más populares[32][33].

Tecnologías VPN y protocolos

Una serie de tecnologías y protocolos se utilizan para permitir VPNs Site-to-Site y VPNs de acceso remoto. Estos protocolos y tecnologías se describen a continuación[32]:

- ***IPsec:*** Consiste en un conjunto de protocolos diseñados para proteger el tráfico IP entre gateways de seguridad o hosts, ya que transita por una red intermedia. Túneles IPsec a menudo se utilizan para construir un sitio a sitio entre los dispositivos de CE (VPN basada CE).
- ***GRE***: Se puede utilizar para construir túneles y transportar tráfico multiprotocolo entre dispositivos CE en una VPN. GRE tiene poca o ninguna seguridad inherente, pero los túneles GRE puede ser protegido mediante IPsec.
- ***Proyecto de Martini:*** Permite el transporte de punto a punto de protocolos como Frame Relay, ATM, Ethernet, Ethernet VLAN (802.1Q) y el tráfico PPP a través de MPLS.
- ***L2TPv3:*** Permite el transporte de punto a punto de protocolos como Frame Relay, ATM, Ethernet, Ethernet VLAN, HDLC, y el tráfico PPP a través de una IP u otro Backbone.
- ***Túnel IEEE 802.1Q (Q-in-Q)***: Permite a un proveedor de servicios para hacer un túnel con tráfico Ethernet (802.1Q) etiquetando al cliente a través de una red troncal compartida.
- ***MPLS LSP:*** Un LSP es un camino a través de routers Label Switch (LSR) en una red MPLS. Los paquetes se conmutan basado en etiquetas antepone al paquete.

Debido a las ventajas económicas que ofrecen las Redes Privadas Virtuales se trata de una excelente tecnología para el acceso remoto, puesto que el uso de una VPN constituye un sustituto indispensable a los métodos tradicionales. Además, constituye una buena solución alterna a los métodos de implementación de redes WAN tradicionales. El auge de Internet ha hecho que muchas de las tecnologías de información giren alrededor de la red de redes, y una VPN no es la excepción, ya que se está convirtiendo en una tecnología casi exclusiva para redes IP y dejando

atrás sus orígenes en ATM y Frame Relay. Es por eso que se espera que el término VPN se aplique solamente a las redes IP, y por lo tanto, a Internet[33].

2.3. Propuesta de Solución

Se diseñó un control de flujo para una red definida por software usando sensores térmicos, con el objetivo de tener una red no dependiente únicamente del controlador y así más autónoma siendo capaz de cambiar sus flujos en dependencia de la temperatura de los equipos de red.

CAPÍTULO 3

Metodología

3.1. Modalidad Básica de la investigación

Se realizó una investigación bibliográfica documental que registró una descripción concreta y concisa de los avances que se obtuvieron durante el desarrollo del presente proyecto de investigación, estudiando los problemas con el propósito de solucionarlos y hacer énfasis en el conocimiento según los requerimientos del proyecto de investigación presento.

Además se realizó una investigación de campo la que determino los problemas que existieron, con el fin de obtener y recopilar información que fue útil para el desarrollo y necesidades del proyecto.

Se tomó en cuenta una investigación aplicada, cumpliendo objetivos fundamentales como el resolver problemas prácticos que otorgue el desarrollo de la investigación

3.2. Población y muestra

En este proyecto de investigación no requiere la utilización de población y muestra considerando que la información necesaria se encuentra disponible en medios físicos y electrónicos accesibles.

3.3. Recolección de información

Para la recolección de información se tomó en cuenta fuentes bibliográficas, también llamadas de fuentes referencia, las cuales son documentos secundarios que recogen la referencia, esto es, la presentación de otros documentos como los repositorios de publicaciones disponibles dentro de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial en la Universidad Técnica de Ambato.

Se acudió a diversos lugares de fuentes de información como institutos de investigación, videotecas, librerías, archivos, revistas.

3.4. Procesamiento y Análisis de Datos

Para la realización del procesamiento y análisis de datos se tomó en cuenta las siguientes actividades:

La recolección de información se realizó directamente desde sensores ubicados en distintos equipos hacia una base de datos localizada en un servidor local para luego realizar el procesamiento, clasificación y presentación de los datos utilizado procedimientos matemáticos y estadísticos.

3.5. Desarrollo del Proyecto

- Análisis de los rangos máximos y mínimos de la temperatura de los equipos.
- Determinación de un conjunto de nuevos flujos en dependencia de las temperaturas.
- Creación una base de datos para alojar los datos de temperatura.
- Establecimiento de la Plataforma de Hardware Libre.
- Adquisición los Datos de Temperatura Mediante la Plataforma de Hardware Libre.
- Diseños de una Aplicación Web para Visualizar los Datos de Temperatura.
- Establecimiento del Controlador Principal para la Red SDN.
- Inserción de los Nuevos Módulos Dentro del Controlador.
- Conexión Remota de la Red con el Controlador de la red SDN.
- Determinación de Pruebas de Funcionamiento.

CAPÍTULO 4

Desarrollo de la Propuesta

Introducción

En la presente investigación se plantea una arquitectura de red de datos dependiente de sensores térmicos, que consta de dos switchs convencionales y un switch habilitado que soporta el protocolo openflow, con el objetivo de comprobar el correcto funcionamiento de un módulo de balanceo de cargar desarrollado para el controlador Floodlight, el cual fue modificado para ser dependiente de los valores de temperatura entregados por la tarjeta Arduino además del uso de un módulo de filtro MAC. Demostrando que este tipo de redes dependientes puede ser implementada en cualquier infraestructura existente mejorando el rendimiento de la misma. También se plantea el uso de tunneling para la comunicación de la red de datos con el controlador de la red, proyectando una conexión remota a través de internet. En la Figura 4.1 se muestra la topología de red a implementar:

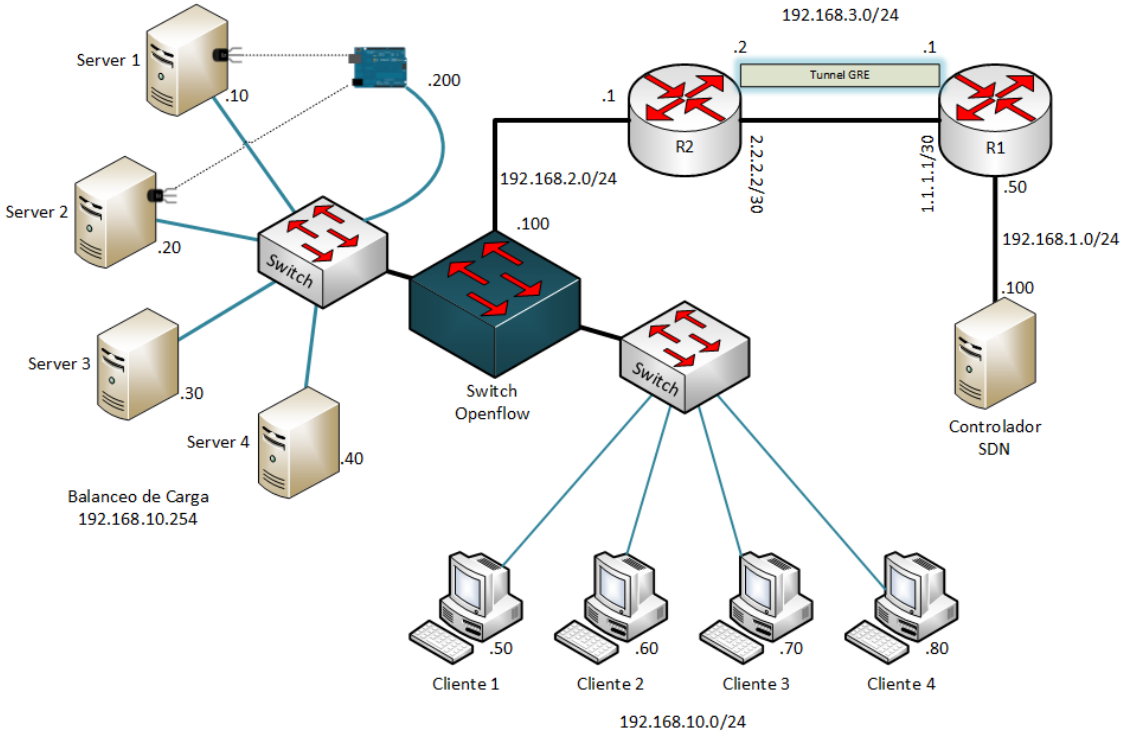


Figura 4.1: Topología de Red a Implementar

4.1. Análisis de los Rangos Máximos y Mínimos de la Temperatura de los Equipos.

Cada equipo de los que dispone una infraestructura de red están diseñados para funcionar dentro de un rango de temperaturas. Al sobrepasar el rango máximo de temperatura estos equipos presentan problemas en su funcionamiento. Estas temperaturas puede variar de gran manera de unos casos a otros, dependiendo del modelo, la carga de trabajo o el tiempo de uso. Uno de los dispositivos que se ve más afectado por alzas de temperatura son los equipos servidores. Un servidor como definición es un ordenador o máquina informática que está al "servicio" de otras máquinas, ordenadores o personas llamadas clientes y que le suministran a estos, todo tipo de información. En la Tabla 4.1 se resume los principales tipos de servidores:

SERVIDOR	DESCRIPCIÓN
Servidor Proxy	Servidor que almacena, envía, recibe y realiza todas las operaciones relacionadas con el e-mail de sus clientes.
Servidor Web	Servidor que actúa de intermediario de forma que el servidor que recibe una petición no conoce quién es el cliente que verdaderamente está detrás de esa petición.
Servidor de Base de Datos	Almacena principalmente documentos HTML, imágenes, vídeos, texto, presentaciones, y en general todo tipo de información. Además se encarga de enviar estas informaciones a los clientes.
Servidores Clúster	Da servicios de almacenamiento y gestión de bases de datos a sus clientes.
Servidores Dedicados	Servidores especializados en el almacenamiento de la información teniendo grandes capacidades de almacenamiento y permitiendo evitar la pérdida de la información por problemas en otros servidores.
Servidores de Imágenes	Son exclusivos para una persona o empresa.

Tabla 4.1: Servidores Usuales

Entre los problemas más comunes que presenta un servidor es el alza de temperatura en su disco duro, comúnmente causado por la carga de trabajo que este puede manejar por lo mismo esta temperatura debe ser controlada y monitorizada para evitar fallos dentro del servidor y seguir manteniéndolo operacional. Una temperatura alta tiene un efecto negativo sobre casi cualquier dispositivo electrónico o electromecánico, incluidas las unidades de disco duro. El impacto de la temperatura sobre la fiabilidad y el tiempo que existe entre fallas se relacionan de casi de forma directa y siempre se considera en el proceso de diseño y prueba de dispositivos[2]. La regla general es mantener las unidades de disco duro tan frías como sea posible a la vez que permanece en la gama especificada por el producto. La gama típica de temperaturas en operación para las unidades de disco duro es desde 5 °C a 55 °C, independientemente de la unidad seleccionada. La Tabla 4.2 presenta los rangos de temperaturas que se pueden encontrar en una unidad de disco duro:

Temperatura	Estado	Funcionamiento
5 - 30 ºC	Normal	Operación Normal del Disco
30-55 ºC	Ligeramente Caliente	Operación Normal del Disco
55-70 ºC	Caliente	Operación Media con Posible Fallas
70-90 ºC	Muy Caliente	Alta Posibilidad de Fallos en el Disco

Tabla 4.2: Rangos de Temperaturas en Discos Duros [2]

El presente proyecto plantea el uso de un rango de temperatura fijo procurando el correcto funcionamiento de los distintos servidores que se utilizarán. El rango de temperatura se establecerá entre 5 ºC que será la temperatura más baja y 45 ºC la temperatura más alta que podrá operar el disco duro del servidor . Si la temperatura de la unidad de disco duro sobrepasa el rango de temperatura permitido el módulo de balanceo de carga activara los servidores de apoyo distribuyendo el tráfico hacia todos los servidores. En caso que la temperatura sobrepasara el valor más alto el servidor tendrá una desconexión inmediata y se entregara un aviso al administrador de red.

4.2. Determinación de un Conjunto de Nuevos Flujos en Dependencia de las Temperaturas.

La infraestructura de la figura 4.1 muestra una red que consta de un conjunto de servidores, de hosts al extremo opuesto de la red y un switch habilitado que se encargara del balanceo de carga. El funcionamiento convencional de la red permite la conexión de los hosts con los servidores y en cierto momento el funcionamiento de la red se puede ver afectado por una saturación de los servidores.

La implementación de esta infraestructura como una red definida por software permitiría un control mucho más detallado de los flujos que maneja la red así como la optimización de los recursos ya disponibles. La incorporación de sensores añade una variable al controlador con la que puede tomar nuevas decisiones y permitir un comportamiento más eficiente de la red.

La saturación de los servidores se puede ver reflejada en problemas como difícil acceso a los servicios que dispone y físicamente se ve reflejado en la temperatura que puede alcanzar el disco duro al tener una sobrecarga de trabajo, siendo su disco duro el dispositivo que se ve más afectado[2]. La incorporación de un módulo de balanceo de carga a la infraestructura proporcionara una disminución en la saturación de los servidores al ser capaz de distribuir el tráfico de la red y una disminución en la temperatura de los mismos por tener una carga de trabajo menor.

La Figura 4.2 muestra un ejemplo de balanceo de carga, donde los clientes pueden acceder a la información de los servidores por medio del balanceo de carga.

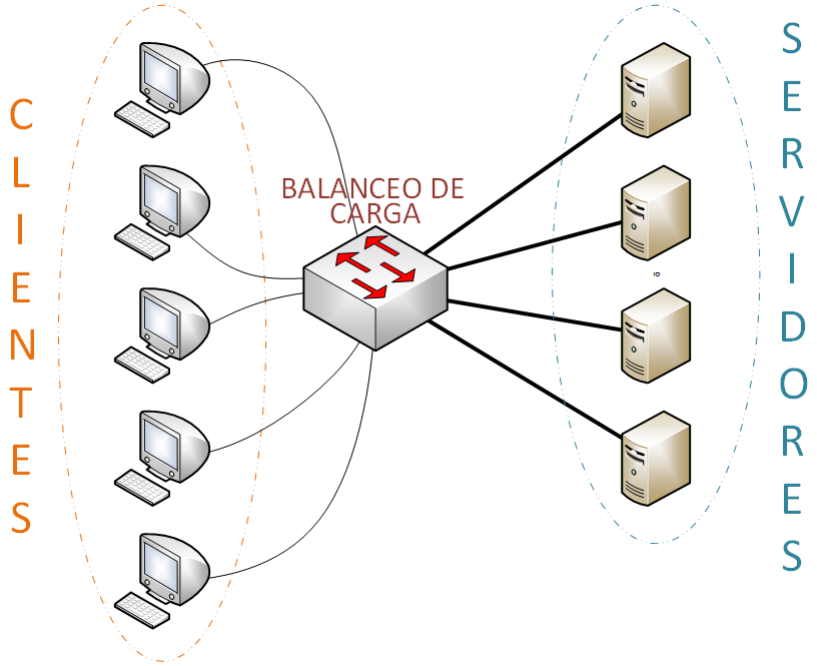


Figura 4.2: Ejemplo de Balanceo de Carga

Para determinar el nuevo conjunto de flujo de datos es necesario cumplir con diversos requerimientos tanto en hardware como en Software:

- Requerimientos de Software
  - Creación de una Base de Datos.
  - Establecimiento del Controlador Principal de la Red.
  - Inserción de los Nuevos Módulos dentro del Controlador.
- Requerimientos de Hardware
  - Establecimiento de la Plataforma de Hardware Libre.
  - Adquisición de Datos de Temperatura Mediante la Plataforma de Hardware Libre

El desarrollo de los requerimientos tanto de Hardware como de Software se describen en los siguientes apartados (4.3, 4.4, 4.5, 4.6 y 4.7).

4.3. Creación una Base de Datos para Alojar los Datos de Temperatura.

Para el desarrollo del proyecto se utilizará la herramienta PhpMyAdmin ya que al ser un prototipo no es necesaria una herramienta con capacidades elevadas por lo mismo se adapta perfectamente al proyecto.

PhpMyAdmin es uno de los servicios más comunes dentro de servidores, mismo que permitirá el acceso a una interface para la creación y administración de bases de datos mucho más sencilla, esta herramienta permitirá el almacenamiento de los distintos datos de temperatura que entrega la placa Arduino.

4.3.1. Creación Base de Datos y Tablas

La Figura 4.3 muestra la interface de PhpMyAdmin donde se procede a la pestaña *Bases de Datos* situada en la parte superior de la interface.

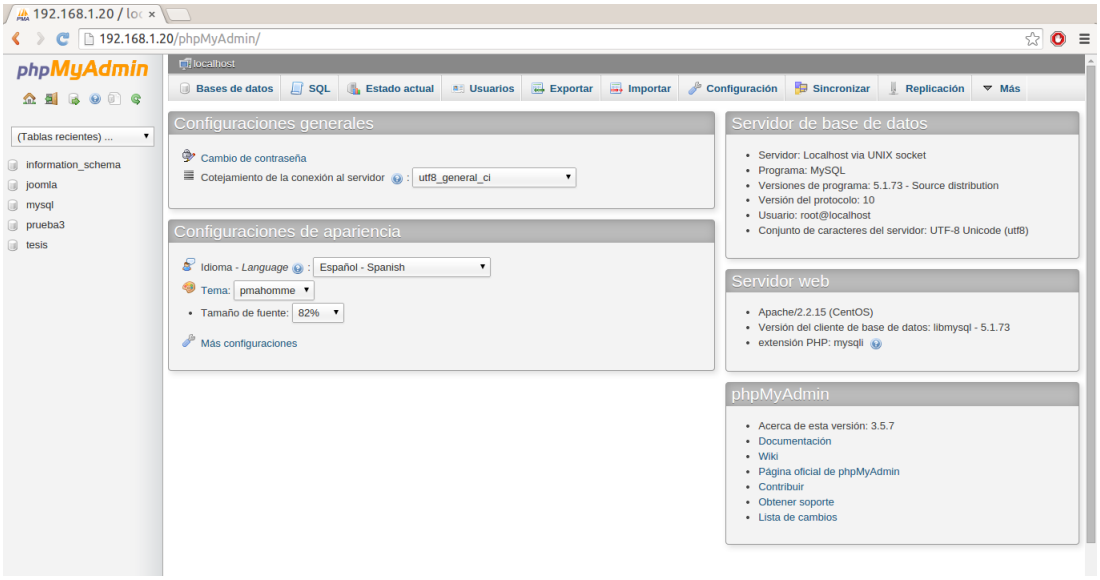


Figura 4.3: Interface PhpMyAdmin.

Dentro de esta pestaña se visualiza las bases de datos existentes como se muestra en la Figura 4.4, al igual que permite la creación de nuevas bases de datos. Para la creación de una nueva base de datos es necesario simplemente ingresar el nombre de la base de datos y seleccionar el tipo de base que se manejara, consecuentemente el listado de las bases se actualizara de forma inmediata tanto en la pestaña de *Bases de Datos* como en la parte izquierda donde también se puede visualizar las bases de datos existentes.

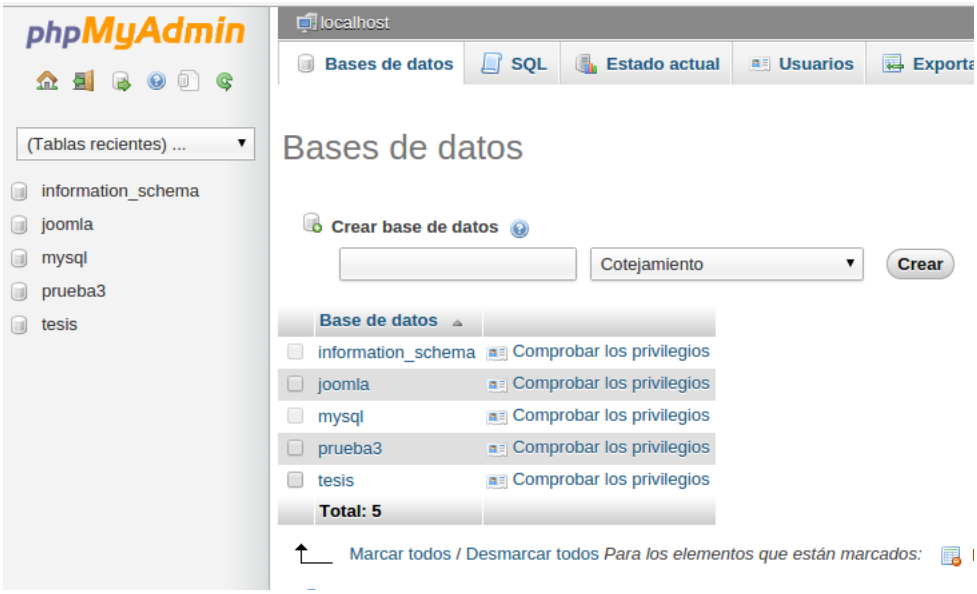


Figura 4.4: Pestaña Bases de Datos PhpMyAdmin

Se trabajará con una base de datos que lleva por nombre *Tesis* que será de tipo cotejamiento. Una vez creada la nueva base de datos es posible acceder a ella desde el listado de bases de datos en la parte izquierda de la interface, esta al ser una base de datos nueva permitirá la creación de tablas para almacenar información. En la Figura 4.5 se crea una nueva tabla donde basta con ingresar el nombre que llevara dicha tabla y el número de columnas que esta tendrá.

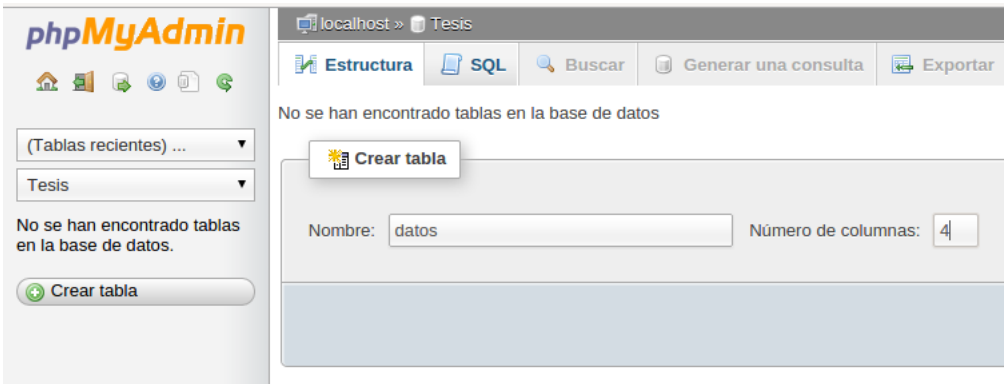


Figura 4.5: Creación de Tabla PhpMyAdmin

Posteriormente es necesario especificar los parámetros para cada una de las columnas que posee la nueva tabla, con esto es posible definir los distintos formatos con los que se almacenaran los datos y que podrán ser consultados más adelante. La Figura 4.6 muestra los parámetros que se utilizan para el proyecto.

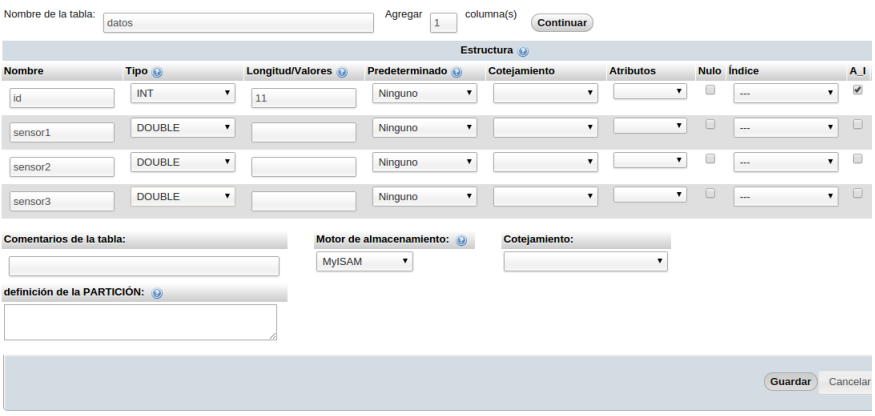


Figura 4.6: Parámetros de Tabla PhpMyAdmin.

Una vez finalizado este proceso es posible visualizar los distintos datos que posee la nueva tabla, así también es posible visualizar su estructura, acompañado de información general acerca de la tabla como lo evidencia la Figura 4.7.

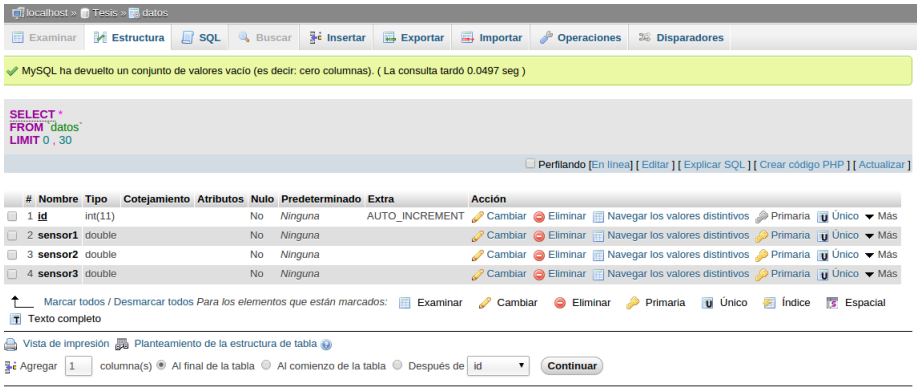


Figura 4.7: Nueva Tabla Creada PhpMyAdmin.

4.4. Establecimiento de la Plataforma de Hardware Libre.

Actualmente en el mercado se puede encontrar un sin número de plataformas las cuales presentan características bastante diversas llegando a compararse con mini computadores. Las selección de una plataforma de hardware siempre dependerá de las necesidades de un proyecto, no siempre la plataforma con las más grandes características se adaptara a un proyecto. En la Tabla 4.3 se presenta las plataformas más representativas actualmente en el mercado y se analiza cual es la más adecuada para el presente proyecto.

Plataformas de Hardware Libre			
Características	**Arduino**	**Rapberry Pi**	**Intel Galileo**
Precio	$29.95	$35	$89
Procesador	ATMega2560	ARM11	Quark SoC X1000
Clock Speed	16MHz	700MHz	700MHz
RAM	2Kb	256Mb	256Mb
Flash	32Kb-256Kb	SD Card	4GB SD Card
EEPROM	1-4Kb	NO	NO
Alimentación	7-12V	5V	5V
Consumo	42mA (0.3W)	700mA(3.5W)	170mA(0.85W)
Puertos Digitales	14-54	14	14
Puertos PWM	6-14	NO	6
Entradas Analógicas	6-16	Ninguna	6
USB	1,Solo Entrada	4,Periféricos	1, Entrada
Programador	Arduino IDE	Scratch,IDLE	Arduino IDE, Eclipse
SPI	1	1	1
Ethernet	Shield	10/100	10/100
Video Out	NO	HDMI	NO
Audio Out	NO	HDMI,Anlog	NO

Tabla 4.3: Plataformas de Hardware Libre Características

De las plataformas anteriormente mencionadas, cada una posee características que las convierten en la mejor para distintos propósitos, Arduino es la plataforma que actualmente posee más versiones a la hora de elegir una placa, desde la LilyPad, que posee un procesador de 8 MHz hasta la Due con 84Mhz.

El código generado para esta plataforma se ejecuta en tiempo real y se tiene acceso directo al hardware, lo cual puede ser bueno para su comportamiento o malo si no se programa adecuadamente. Posee muchísimas librerías, módulos y sensores, y hasta se pueden conectar dispositivos que no sean específicamente para Arduino. Al ser una plataforma basada en un microcontrolador de muy bajo costo se convierte en la plataforma adecuada para el presente proyecto, que se puede explotar su potencial al máximo.

4.5. Adquisición los Datos de Temperatura Mediante la Plataforma de Hardware Libre.

Las plataformas de Hardware libre permiten la integración de sensores de tipo analógico o digital, gracias a sus diversas entradas de ambos tipos que permiten un

acoplamiento directo y así mismo un tratamiento de la información mucho más fácil y dinámica.

En la Tabla 4.4 se destacan algunos de los sensores más comunes para la implementación de sistemas de control de temperatura en conjunto con plataformas de hardware libre.

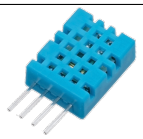
Sensores de Temperatura					
Características	**LM35**	**DHT11**	**DHT22**	**DS18B20**	**Termistor**
					
Precisión	0,4 °C	2 °C	0,5 °C	0,5 °C	1 °C
Rango	-55 °C a 150 °C	0 °C a 50 °C	-40 °C a 80° C	-55 °C a 125 °C	-195 °C a 450 °C
Librerías	Ninguno	Dhtlib	Dhtlib	OneWrire Dallas Temperature	Ninguno
Precio	$ 1.5	$5	$ 10	$4.25	$ 3
Linealidad	Excelente	Buena	Buena	Buena	Pobre
Alimentación	4V a 30V	3V a 5.5V	3V a 5.5V	3V a 12V	1.92V a 12V
Salida	Analógica	Digital	Digital	Digital	Analógica
Dimensiones	4.5*17.3*3.3mm	12*15.5*5.5mm	12*15.5*5.5mm	4.9*14.7*3.9mm	2*3*5mm

Tabla 4.4: Sensores de Temperatura Características

Una vez analizadas las características de los distintos sensores se concluye que para el desarrollo del proyecto se utilizará el sensor LM35 por presentar las características que más se adecuan al proyecto como su bajo costo, tamaño reducido, alta precisión y un amplio rango de temperatura.

La Figura 4.8 presenta es esquema básico que se manejara para el proyecto, detallando la utilización de una placa Arduino y los sensores de temperatura LM35.

Así también la incorporación de un módulo Shield Ethernet Arduino al esquema de la figura anterior facilitara el envío de información de forma directa hacia una base de datos alojada en un servidor.

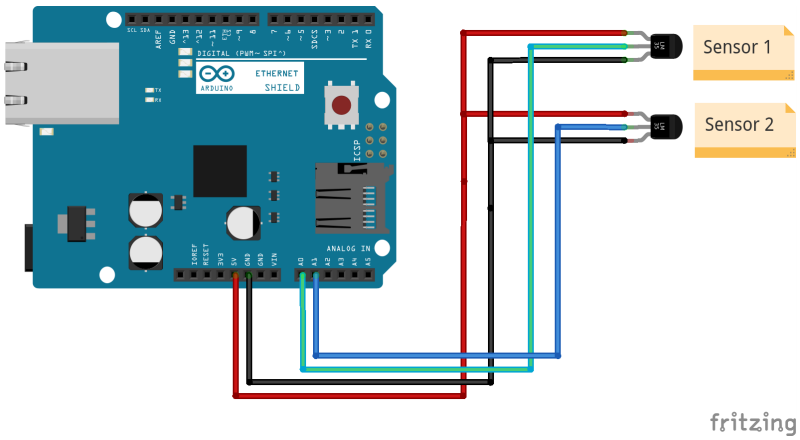


Figura 4.8: Esquema Sensores/Arduino.

La plataforma Arduino trabajará en el modo WebClient para poder tener una conexión con el servidor y almacenar los distintos datos de temperatura en la base de datos establecida. Arduino se apoya en un método de programación mismo que comprueba la conexión con el servidor y se encarga de enviar los datos de temperatura.

El siguiente extracto de código es el encargado de comprobar la conexión con el servidor y posteriormente envía los datos hacia el archivo *datos.php* que almacena los datos en el servidor.

```
void connectToServer(double sensor1,double sensor2,double sensor3){
  if (client.connect(serverName, 80)) {
        Serial.println("Realizando peticion HTML...");
        client.print("GET /cleantype/datos.php?sensor1=");
        client.print(sensor1);
        client.print("&sensor2=");
        client.print(sensor2);
        client.print("&sensor3=");
        client.print(sensor3);
        client.println(" HTTP/1.1");
        client.println("HOST: 192.168.1.200");
        client.println();
}
```

4.5.1. Descripción de Scripts para la Adquisición de Datos desde Arduino hacia phpMyAdmin

Para la interconexión de la plataforma de adquisición con la base de datos existente se necesitan de los siguientes archivos PHP que se alojaran en el servidor donde se encuentra la base de datos:

- Archivo de configuración.

  Se definen parámetros de la base de datos como el usuario con el que se ingresara, la contraseña de ser necesaria y el nombre de la base de datos.

```
var $serv="localhost";
var $usuario="root";
var $contra="andres";
```

  Se crea una nueva función que ayudara el envío de los datos anteriores para gestionar la conexión con la base de datos.

```
function conecta(){
$s=$this->serv;
$u=$this->usuario;
$c=$this->contra;
$conex=mysql_connect($s,$u,$c);
$this->conexi=$conex;
}
```

  Se procede a crear una nueva conexión y a llamar función anteriormente creada y se selecciona la base de datos con la que se trabajará.

```
$cono= new conexion();
$cono->conecta();
$c=$cono->conexi;
$select=mysql_select_db("tesis",$c);
```

- Archivo de Inserción de Datos en la Base

  Este archivo utiliza el código del archivo de configuración anteriormente descrito para realizar la conexión a la base de datos al igual que se declaran las variables a insertarse dentro de la teniendo en cuenta que deberán llevar el mismo nombre que en la base de datos.

```
include($_SERVER['DOCUMENT_ROOT'] ."/joomla/datos/clase_conexion.php");
$type=$_GET['sensor1'];
$type1=$_GET['sensor2'];
$type2=$_GET['sensor3'];
```

Posterior a esto es necesario realizar la inserción en la tabla dentro de la base usando comandos SQL, procurando mantener el orden en el que se insertaran los datos.

```
$consulta="insert into datos values (NULL,'".$type."','".$type1."','".$type2."')";
```

4.6. Diseños de una Aplicación Web para Visualizar los Datos de Temperatura.

El diseño de una aplicación Web se ve facilitada de gran forma con la utilización de un sistema de gestión de contenidos por presentar muchas facilidades y por brindar una amplia documentación para el desarrollo de aplicaciones que mejor se adapten. En la Tabla 4.5 se analizan los tres sistemas de gestión de contenidos más usados en el medio.

Sistemas de Gestión de Contenidos

Características	***Joomla***	***Drupal***	***WordPress***
Tipo de Websites	Grandes Pequeños	Grandes	Grandes Pequeños
Lenguaje	PHP	PHP	PHP
Dificultad	Baja	Media	Baja
Flexibilidad	Alta	Alta/Media	Alta
Dificultad para Administrador	Baja	Alta	Media
Actualizaciones	SI	SI	SI
Soporte Móvil	SI	SI	SI
Escalable	SI	NO	SI
Base de Datos	MySQL	MySQL	MySQL
Tiempo de Instalación	5 min	15 min	10 min

Tabla 4.5: Gestores de Contenidos Características

Para el desarrollo del proyecto se optará por el sistema de gestión de contenidos Joomla por presentar las características necesarias para el proyecto, además de contar con una comunidad de desarrolladores bastante amplia y activa. Joomla será instalado directamente en un servidor desde donde se administraran los distintos contenidos que poseerá la aplicación web.

4.6.1. Creación de Scripts para Visualizar la Tabla Principal de la Base de Datos

Dentro de la aplicación web será necesario visualizar los distintos datos que se encuentran almacenados dentro de la base de datos por lo que son necesario archivos que permitan esta visualización. Estos archivos serán añadidos a la aplicación gracias a la herramienta Wrappers que posee por defecto Joomla, a continuación se detallan los scrips para la visualización.

- Archivo para Consultar la Base de Datos

  Inicialmente se declaran parámetros con los cuales se podrá acceder a la base de datos.

```
$host = "localhost";
$user = "root";
$pw = "andres";
$db = "tesis";
```

  Seguido se usan los parámetros anteriores para verificar la conexión con la base de datos caso contrario se recibirá un mensaje de error y si todo es correcto se realiza la consulta almacenándola en una variable que contendrá todos los datos que se encuentran almacenados en la tabla seleccionada.

```
$con = mysql_connect($host, $user, $pw)
or die("No se pudo conectar a la base de datos ");

mysql_select_db($db, $con)
or die ("No se encontro la base de datos. ");

$query = "SELECT * FROM datos";
$resultado = mysql_query($query);
```

Una vez realizada la consulta es necesaria la visualización de estos datos para lo cual se utiliza un ciclo WHILE, así también se establece el formato en el que se presentaran los datos.

```
while ($fila = mysql_fetch_array($resultado)) {
echo " <tr>";
echo "ID=<td> $fila[id]</td> |
SENSOR 1 = <td>$fila[sensor1]</td> ºC |
SENSOR 2 = <td>$fila[sensor2]</td> ºC |
SENSOR 3 = <td>$fila[sensor3]</td> ºC <br>";
echo " </tr> ";
```

- Archivo para Visualización.

  Este archivo se encarga de llamar y ejecutar el archivo anterior siendo mucho más fácil para la visualización dentro de la aplicación Web.

  Así también se encarga de generar una mejor presentación para la información anteriormente adquirida.

```
<?php
include("conexion-arduino.php");
$Con = new conexion();
$Con->recuperarDatos();
?>
```

4.6.2. Asignación de Wrappers en Joomla para Visualización de Datos

El CMS Joomla permite la asignación de wrappers en una página web ya desarrollada con lo que facilita el tener páginas embebidas. Para el proyecto esta herramienta agiliza la visualización de las distintas temperaturas almacenadas en la base de datos existente. Para usar esta herramienta es necesario realizar el siguiente procedimiento:

- Como primer paso a seguir es necesario ingresar a la parte administrativa de la página ya creada.
- Es necesario ingresar al gestor de menús y seleccionar el menú en donde se agregar la página embebida, esto desplegará una ventana como se ve en la Figura 4.9.

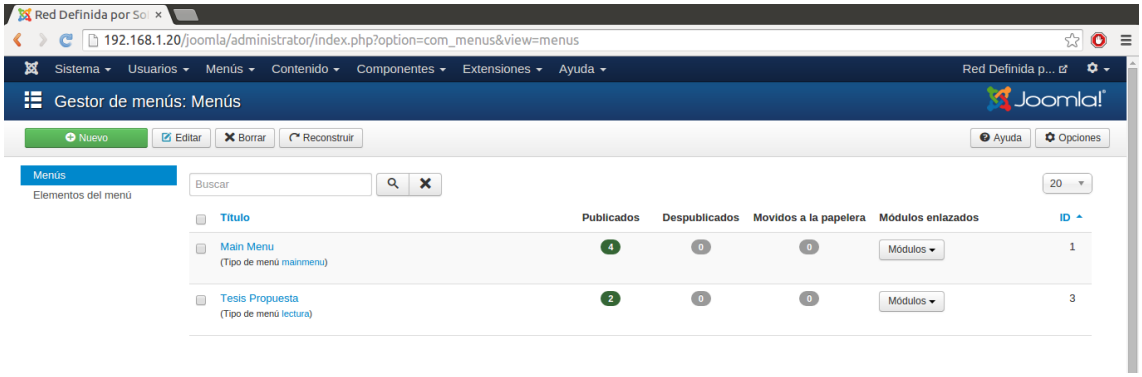


Figura 4.9: Gestor de Menús Joomla.

- Una vez dentro del menú seleccionado se crea un nuevo ítem mismo, a este ítem se le agrega la página embebida usando el URL de dicha página. La Figura 4.10 muestra el Gestor de Menús con varios ya creados.

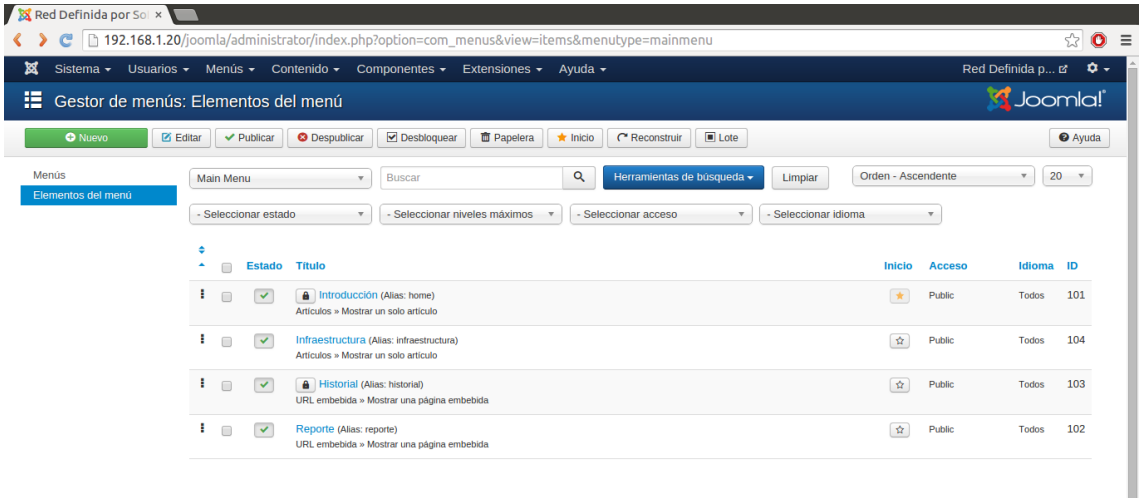


Figura 4.10: Nuevo Ítem de Menú.

- Dentro de las configuraciones del nuevo Ítem se debe establecer que será un ítem con una página embebida y se especificara el URL donde se encuentra alojado para el proyecto se llamará al archivo ***arduino.php (http://192.168.1.20/joomla/datos/arduino.php)*** que permite visualizar los datos de temperatura de los distintos sensores. La Figura 4.11 muestra este proceso.

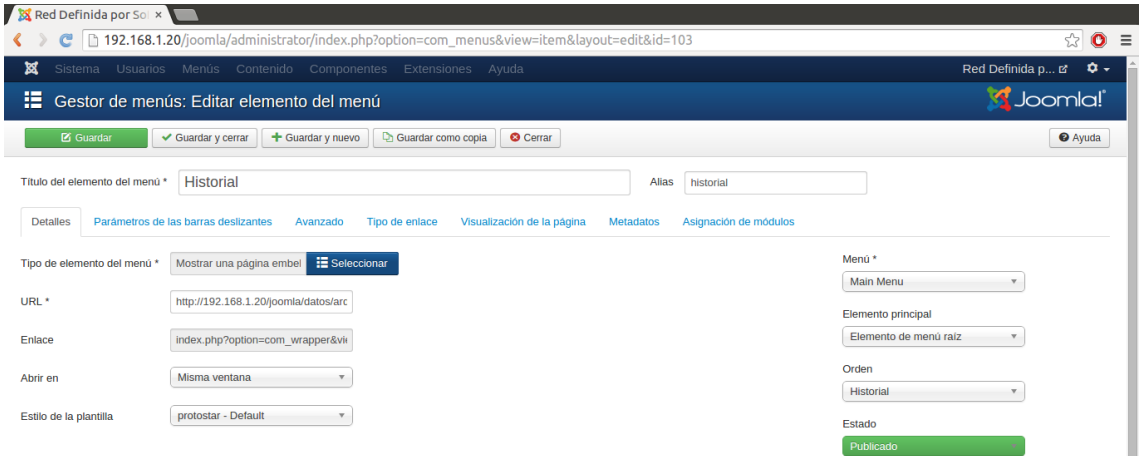


Figura 4.11: Configuración de Nuevo Ítem.

- Posterior al proceso antes mostrado se visualizará en la aplicación WEB el nuevo ítem dentro del menú y se mostrará la página embebida como se puede ver en la figura 4.12.

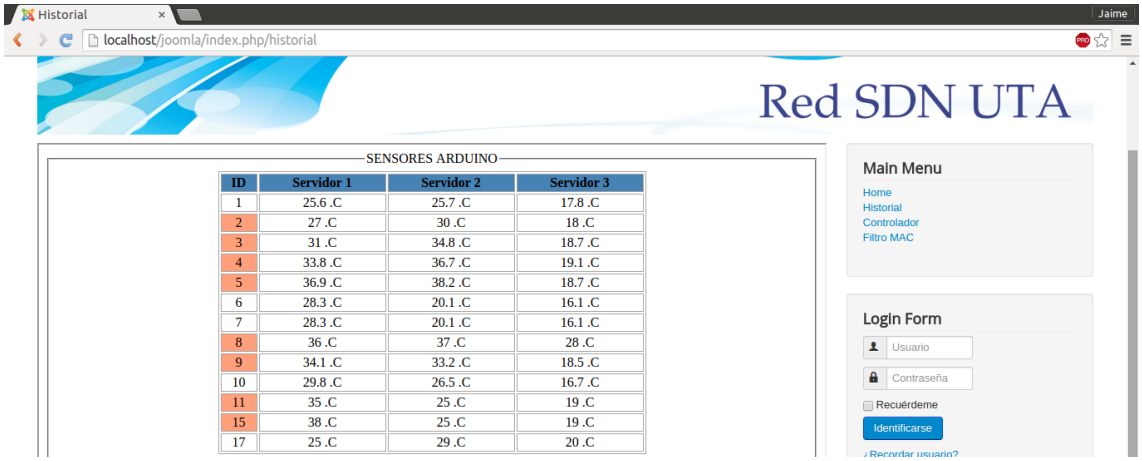


Figura 4.12: Página Embebida en la Aplicación WEB.

- En la Figura 4.13 se muestra la interfaz gráfica del módulo de Filtro MAC el cual también se agrega a la interfaz gráfica para controlar la conexión entre los host y los servidores sin pasar por el módulo de Balanceo de carga.

Figura 4.13: Módulo de Filtro Mac Interfaz WEB.

4.7. Establecimiento del Controlador Principal para la Red SDN.

El componente principal de la infraestructura de una red SDN siempre será el Controlador de red por lo que la elección de uno de ellos debe seleccionarse entorno a los requerimientos de la red así como las capacidades que pueda tener.

En la Tabla 4.3 se presentan las distintas características de 4 de los controladores más usados para la implementación de redes SDN de los cuales se seleccionara el más adecuado para el desarrollo del proyecto.

CONTROLADORES DE RED SDN				
CARACTERÍSTICAS	**NOX**	**POX**	**BEACON**	**FLOODLIGHT**
Lenguaje de Desarrollo	C++	Python	Java	Java
Lenguajes Soportados	C C++ Python	Python	Java	Java Python
Facilidad de Instalación	Regular	Fácil	Fácil	Fácil
Simplicidad	Regular	Regular	Fácil	Fácil
Soporte	Malo	Bueno	Regular	Excelente
Módulos para desarrollo	Malo	Bueno	Regular	Bueno
Comunidad Activa	No	Si	Si	Si
Documentación	Mala	Mala	Buena	Buena
Provee REST API	No	No	No	Si
Bucles en la Topología	No	No	No	Si
Interfaz Gráfica	Python+ QT4	Python+ QT4 Web	Web	Java Web

Tabla 4.6: Características de Controladores SDN[3]

Una vez analizadas todas las características propuestas en la tabla 4.6 de los distintos controlados se ha decidió establecer a Floodlight Controller como el controlador de la red a implementar ya que presenta las mejores características así mismo de poseer muchas herramientas para el correcto desarrollo de proyecto.

4.7.1. Instalación de Floodlight Controller

Para la instalación de Floodlight se tomó en cuenta algunos prerrequisitos que debe poseer el sistema antes de la instalación, por ende es necesario instalar diversos paquetes en un sistema Linux Ubuntu 10.01 o superior.

La instalación del controlador en Ubuntu se realiza vía terminal para evitar cualquier tipo de error, además es necesario la instalación de Eclipse que será la plataforma que permitirá la ejecución del controlador así como la edición y creación de módulos[19].

- Paquetes esenciales

  sudo apt-get install build-essential default-jdk ant python-dev

  sudo apt-get install eclipse

  Una vez instalado eclipse es necesario exportar el proyecto Floodlight desde el repositorio oficial de Floodlight

  $ sudo git clone git://github.com/floodlight/floodlight.git

  El comando anterior exportara el proyecto de forma completa en su última versión.

  Una vez descargado el controlador es necesario posicionarnos en la carpeta reciMismo én descargada y ejecutar el comando “ant” para generar diversos archivos que permitirán la ejecución del controlador

  $ cd floodlight && ant

  Realizados todo el procedimiento anterior se puede ejecutar el controlador de forma directa cargando la configuración que trae por default con el comando:

  $ java -jar target/floodlight.jar

  En este caso particular es necesario modificar los parámetros de configuración del controlador por lo que se debe importar el proyecto hacia Eclipse. Para poder importar el proyecto se debe ejecutar el siguiente comando que permitirá importar el proyecto dentro de un workspace de Eclipse.

  $ sudo ant eclipse

4.7.2. Importación y Ejecución de Floodlight Controller a Eclipse

Dentro de Eclipse se procede a la pestaña File -> Import lo cual desplegará la siguiente ventana como se muestra en la Figura 4.14:

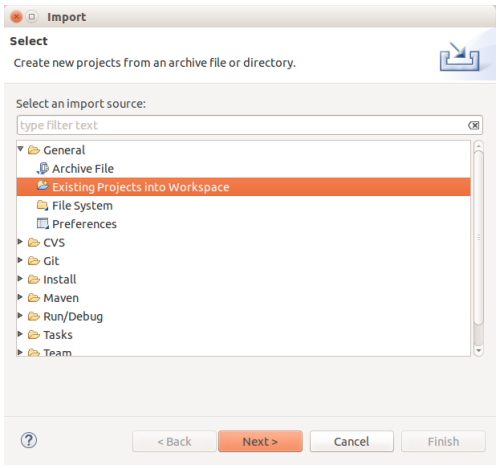


Figura 4.14: Ventana Import Eclipse.

En la cual se selecciona “Existing Projects into Workspace” y luego click en “Next”.

En la siguiente ventana se especifica la carpeta donde se encuentra alojado el proyecto permitiendo terminar el proceso de importación como se muestra en la Figura 4.15.

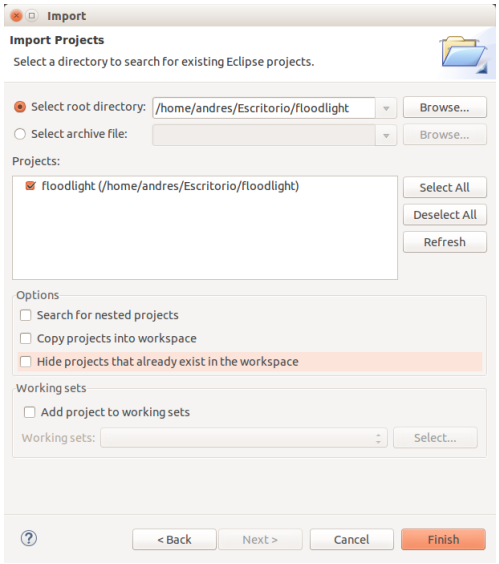


Figura 4.15: Dirección Origen Floodlight Eclipse.

Se selecciona el proyecto Floodlight y se presiona el botón “Finish” para finalizar. La Figura 4.16 muestra la interface de eclipse con el proyecto importado.

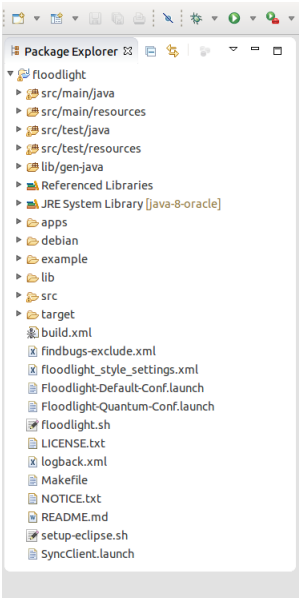


Figura 4.16: Proyecto Importado a Eclipse.

Ya importado el proyecto es necesario establecer una ejecución específica para el controlador para lo cual se procede a la pestaña Run->Run Configurations lo que despegara la ventana que se muestra en la Figura 4.17:

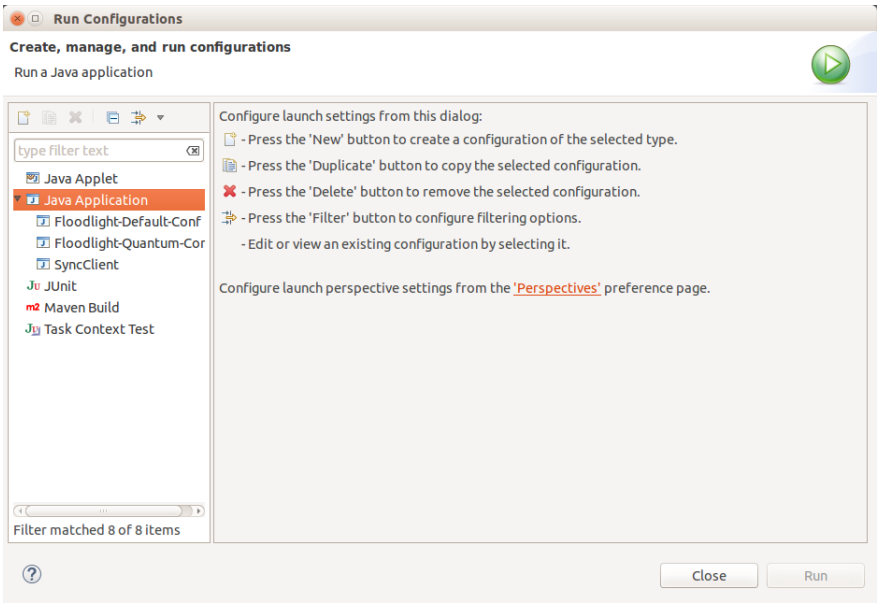


Figura 4.17: Ventana Run Configuration Eclipse.

Se crea una nueva *Java Application* dando click derecho sobre "Java Application" y luego "New" y donde se establecerá los siguientes parámetros tal como se indica en la Figura 4.18:

- Nombre: *FloodlightLaunch*
  - Proyecto: *floodlight*
  - Main Class: *net.flooflightcontroller.core.Main*

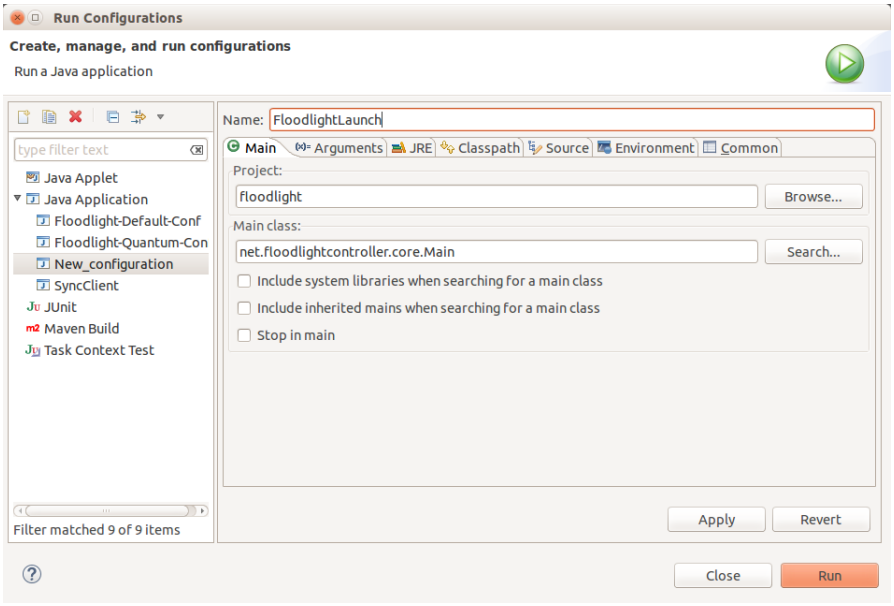


Figura 4.18: Configuración Run para Floodlight

Se pulsa "Apply" y posteriormente "Run" luego de esto Eclipse compilara el proyecto y ejecutara el controlador. La Figura 4.19 muestra la ejecución del controlador en la consola de Eclipse.

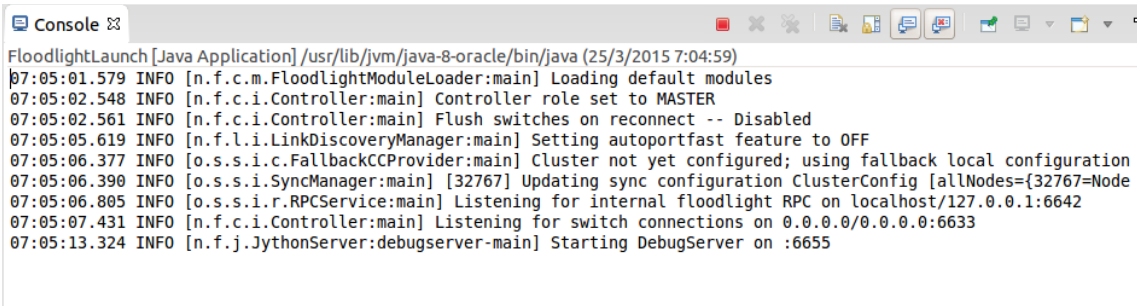


Figura 4.19: Proyecto Compilado y en Ejecución.

4.8. Inserción de los Nuevos Módulos Dentro del Controlador.

4.8.1. Creación de Módulos dentro de Floodlight

Para la creación de nuevos módulos es necesario expandir la carpeta Floodlight en el Package Explorer y buscar la carpeta *"src / main / java"*.

Es necesario hacer click derecho sobre la carpeta *"src / main / java"* y seleccionar *"New/Class"* .

Se desplegara la siguiente ventana donde se llenara los siguientes parámetros:

- *"Package"* es el nombre del paquete especifico que contendrá los módulos necesarios, el recuadro llevara el texto con la siguiente estructura *"net.floodligthcontroller.balanceotesis"* siendo balanceotesis el nombre del paquete.
- "Name" es el nombre del primer módulo que se creara dentro del paquete, en este caso será *"BalanceotesisSW1"* donde específicamente debe iniciar con una letra mayúscula diferenciándolo así del nombre del paquete. Todos los módulos deben llevar la primera letra en mayúscula.

Llenados estos campos es necesario dar click en "Finish" con lo que se creara tanto el nuevo paquete con el primer módulo, este proceso se muestra en la Figura 4.20.

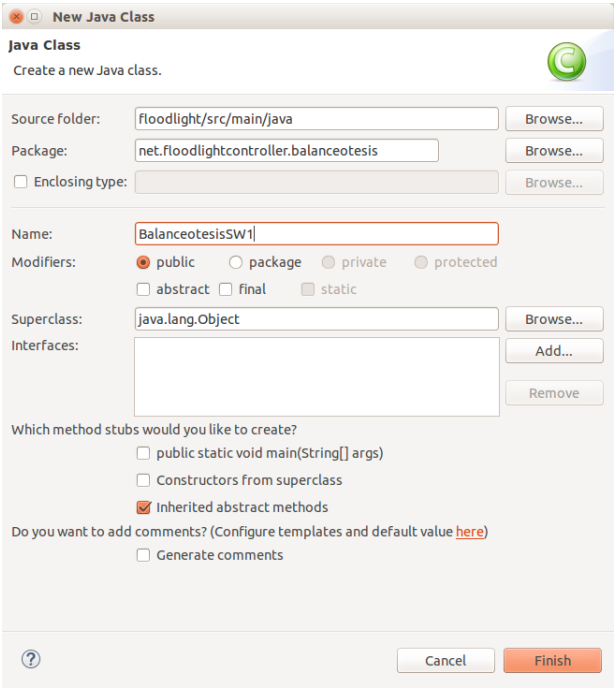


Figura 4.20: Creación de Nuevo Paquete/Módulo.

Se obtiene como resultado un nuevo módulo en blanco mismo donde se insertará el código para el balanceo de carga. La Figura 4.21 muestra el nuevo módulo en blanco.

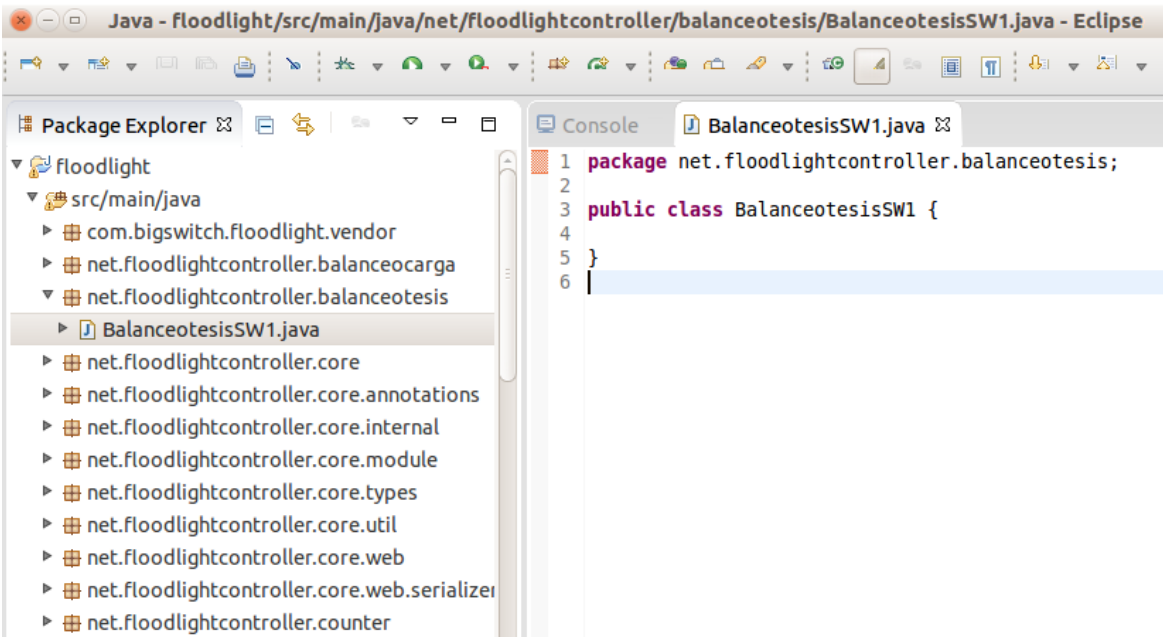


Figura 4.21: Módulo de Controlador en Blanco.

Uno de los modelos de balanceo de carga más utilizados en el ámbito de las redes de datos es el modelo Round-Robin mismo que ya existe para la implementación con el controlador Floodlight, es decir que se necesita insertar el código existente dentro del nuevo módulo para su posterior modificación a los parámetros necesarios.

4.8.2. Descripción del Módulo Creado

Una vez creado el paquete y las clases se importan las librerías necesarias para el balanceo. El módulo existente utiliza el conjunto de librerías openflow para el manejo de la red SDN y adicionalmente se importan las librerías correspondientes a SQL que permitirán el uso de sentencias SQL para poder realizar consultas a las distintas bases de datos:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
```

El módulo existente para el balanceo de carga consta de distintas etapas que pueden ser modificadas a conveniencia del controlador de la red siendo las más importantes las siguientes:

- Dirección IP y MAC address para el balanceo de carga lógico.

```
private final static int LOAD_BALANCER_IP = IPv4.toIPv4Address("192.168.10.254");
private final static byte[] LOAD_BALANCER_MAC = Ethernet.toMACAddress("::::FE");
```

En esta sección de código se puede modificar tanto la dirección Ip como la dirección MAC que dispondrá el balanceador, para el proyecto se tomo la última dirección de red que se puede utilizar y se asigno una dirección MAC bastante simple para identificar de forma rápida el balanceador.

- Los tiempos de vida de las reglas.

```
private final static short IDLE_TIMEOUT = 20; // in seconds
private final static short HARD_TIMEOUT = 60; // infinite
```

El extracto anterior de código permite establecer los tiempos que se grabarán las distintas reglas de direccionamiento dentro de los distintos equipos de la red, al igual que permite el grabador permanente de reglas de direccionamiento.

- Servidores Reales.

```
Server[] SERVERS = {
            new Server("192.168.10.10","00:1c:c0:de:08:31", (short)1),
            new Server("192.168.10.20","00:1c:c0:de:0a:e1", (short)1),
            new Server("192.168.10.30","00:1c:c0:de:08:2d", (short)1),
            new Server("192.168.10.40","00:1c:c0:55:2e:b4", (short)1)
        };
```

El extracto de código anterior permite ingresar la dirección IP y MAC de cada uno de los servidores con los que se realizará el balanceo de carga. Es necesario especificar el puerto exacto del equipo en donde se encuentra conectado el servidor.

Luego del análisis de la estructura del módulo de balanceo se consideró necesario modificar el último método debido a que este otorga el número de servidores reales con los que se hará el balanceo.

Estas modificaciones finalizan con la adición de código que permita realizar una consulta directa hacia la base de datos y así tomar decisiones conforme a la última temperatura registrada.

El siguiente extracto de código muestra el método getNextServer ya modificado:

```
private Server getNextServer() {
try  {
Connection conn = DriverManager.getConnection("jdbc:mysql
    ://192.168.1.20/tesis","root","andres");
String query = "SELECT * FROM datos";
Statement stmt = conn.createStatement();
ResultSet rs = stmt.executeQuery(query);
     if(rs.last()){
     temS1=rs.getDouble("sensor1"); temS2=rs.getDouble("sensor2");
     }
```

```
conn.close();
}
catch (Exception e){
System.err.println(e);
}

if(temS1>30 || temS2>30){
    lastServer = (lastServer + 1) % 4;
    System.out.println("Sensor 1 ="+temS1);
    System.out.println("Senser 2 ="+temS2);
}

else{
    lastServer = (lastServer + 1) % 2;
    System.out.println("Sensor 1 ="+temS1);
    System.out.println("Senser 2 ="+temS2);
}
return SERVERS[lastServer];
}
```

En la Figura 4.22 se muestra el diagrama de funcionamiento del método modificado.

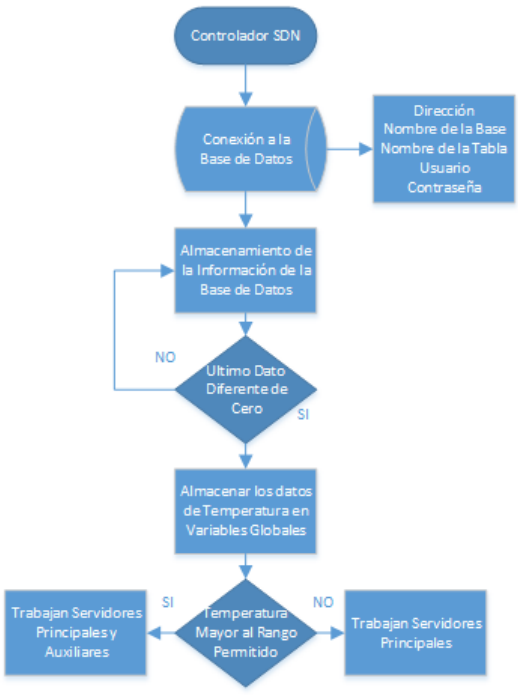


Figura 4.22: Diagrama Del Método getNextServer.

4.9. Conexión Remota de la Red con el Controlador de la red SDN

Se plantea una conexión remota de la red con el controlador de la red a través de internet con lo cual es posible monitorizar la red a grandes distancias. Para la conexión del controlador y la red es necesaria la implementación de una comunicación segura a través de internet, para lo cual la utilización de tunneling resulta la alternativa más viable y ágil.

4.9.1. Configuración de Equipos Destinados al Tunneling

Para la configuración de los equipos es necesario determinar que soporten este tipo de protocolos, para lo cual se puede verificar dentro del manual correspondiente a cada equipo. Para facilidad del proyecto se decidió utilizar los equipos Cisco disponibles de la serie 2800, a los que se verifico el soporte para realizar el tunneling usando el protocolo GRE.

Previo al proceso de configuración de los equipos se determinaron los distintos parámetros a utilizar que se detallan en la Figura 4.1.

- Router 1

  Se establece la configuración de las interfaces del router de la siguiente manera:

  ```
  GigabitEthernet0/0
  ipaddres: 192.168.10.50
  netmask: 255.255.255.0

  GigabitEthernet0/1
  ipaddres: 1.1.1.1
  netmask: 255.255.255.252
  ```

  Para la creación del tunel GRE dentro del Router 1 es necesario emplear los siguientes comando:

  - R1(config)# interface Tunnel0
  - R1(config-if)# ip address 192.168.3.1 255.255.255.0
  - R1(config-if)# ip mtu 1400
  - R1(config-if)# ip tcp adjust-mss 1360
  - R1(config-if)# tunnel source 1.1.1.1
  - R1(config-if)# tunnel destination 2.2.2.2
  - R1(config-if)# keepalive

- R1(config-if)# exit
- R1(config)# wr

- Router 2

  Se establece la configuración de las interfaces del router de la siguiente manera:

```
GigabitEthernet0/0
ipaddres: 192.168.2.1
netmask: 255.255.255.0

GigabitEthernet0/1
ipaddres: 2.2.2.2
netmask: 255.255.255.252
```

  Para la creación del tunel GRE dentro del Router 2 es necesario emplear los siguientes comando:

  - R2(config)# interface Tunnel0
  - R2(config-if)# ip address 192.168.3.2 255.255.255.0
  - R2(config-if)# ip mtu 1400
  - R2(config-if)# ip tcp adjust-mss 1360
  - R2(config-if)# tunnel source 2.2.2.2
  - R2(config-if)# tunnel destination 1.1.1.1
  - R2(config-if)# keepalive
  - R2(config-if)# exit
  - R2(config)# wr

Es necesario resaltar que las direcciones 2.2.2.2/30 y 1.1.1.1/30 son utilizadas solo por facilidad para las pruebas del proyecto, ya que se asume que estas interfaces son las que se conectan con el proveedor de internet, claro que en otro caso podrían ser también interfaces de tipo serial, sin que esto llegue a afectar el funcionamiento del túnel.

Una vez establecidas la configuraciones se puede comprobar el correcto funcionamiento de la red usando el comando ***show ip route*** con el que se mostrarán las distintas conexiones que posee el router y presentando el estado del túnel como una conexión de forma directa.

4.10. Determinación de Pruebas de Funcionamiento.

4.10.1. Pruebas de Tunel GRE

Para determinar el correcto funcionamiento del túnel GRE es necesario verificar si se encuentra activo, para lo cual se utiliza el comando show ip route en cada uno de los routers donde se realizo esta configuración. El comando anterior devolverá como respuesta que interfaces se encuentran activas, entre las cuales la interfaces Tunnel debe indicar una conexión directa.

Las figuras 4.23 y 4.24 muestran la respuestas del comando show ip route para cada uno de los routers configurados.

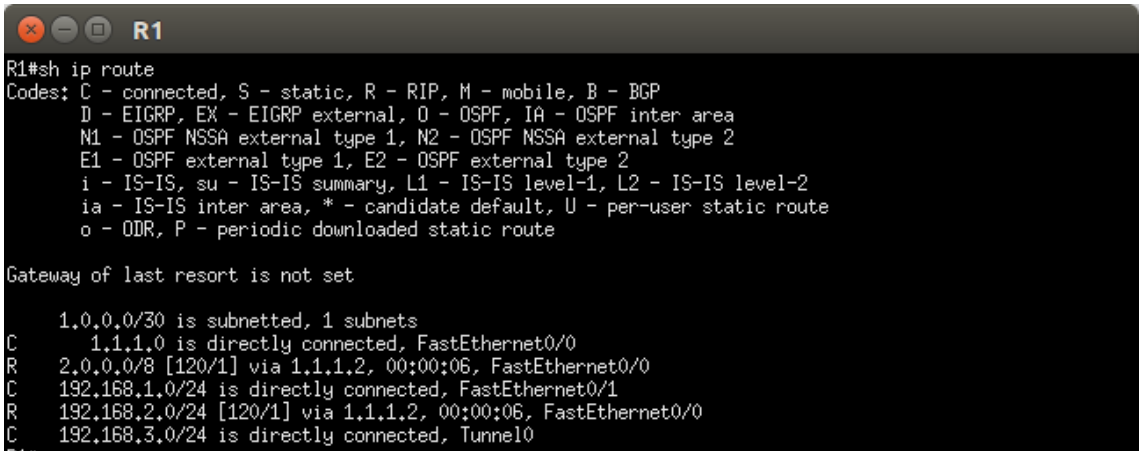


Figura 4.23: Router R1 Tunnel Activo.

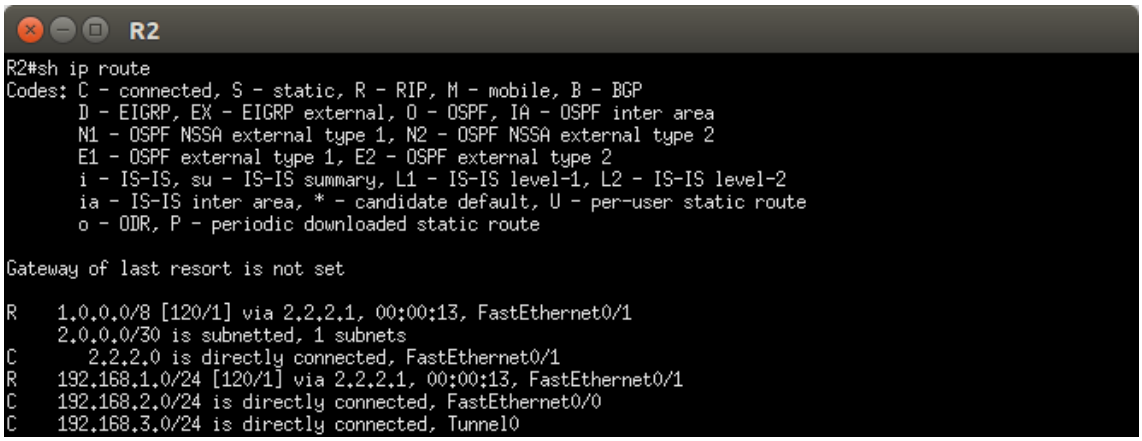


Figura 4.24: Router R2 Tunnel Activo.

En caso de existir alguna falla en la configuración del tunnel al aplicar el comando show ip route no se visualizará la interface Tunnel. Adicionalmente para poder establecer la conexión del controlador con la red es necesario establecer un

direccionamiento estático, mismo que dirigira el tráfico de la vlan del controlador a través del túnel GRE.

Este direccionamiento estático se realiza usando los siguientes comandos:

- R1

  ip route 192.168.1.0 255.255.255.0 192.168.3.1

- R2

  ip route 192.168.2.0 255.255.255.0 192.168.3.2

4.10.2. Pruebas de Balanceo de Carga

Como primer paso es necesario comprobar las tablas ARP de cada uno de los hosts y de los servidores para verificar que se encuentren todos los dispositivos de la red detallados en la tabla ARP con su respectiva IP y MAC Address. En la Figura 4.25 se muestra la tabla ARP de una de los Host de la red.

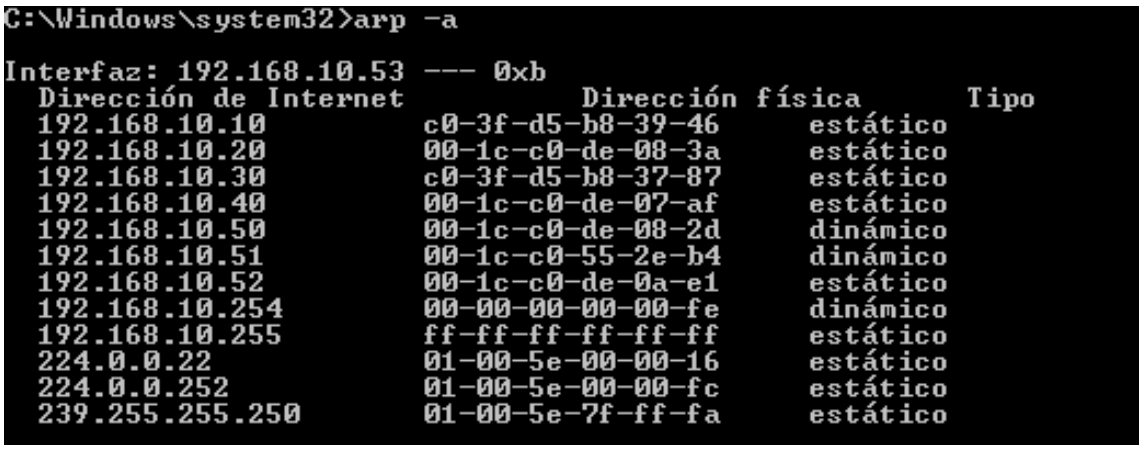


Figura 4.25: Tabla ARP Host 4.

El uso de switchs convencionales proporciona la conexión entre los dispositivos que se encuentran conectados en el mismo, de igual forma el uso de un switch habilitado limita la conexión con el otro grupo de dispositivo ya sean host o servidores. La única forma en la que se pude acceder hacia el otro grupo de dispositivos es mediante el uso del balanceo de carga o por medio de la activación del módulo de Filtro MAC. La figura 4.26 muestra la conexión del switch habilitado y los switch convencionales.

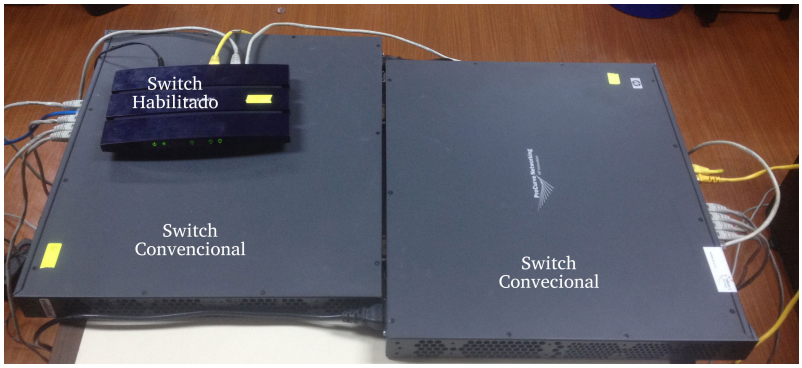


Figura 4.26: Switchs Convencionales y Habilitado.

Para comprobar el correcto funcionamiento del módulo de balanceo se prueba la conexión con el balanceo de carga haciendo ping hacia la dirección del balanceo es decir a la IP 192.168.10.254. La Figura 4.27 muestra la conexión de uno de los host hacia el Balanceo de Carga y la Figura 4.28 muestra al grupo de Host accediendo al balanceo simultáneamente.

```
C:\Windows\system32>ping 192.168.10.254

Haciendo ping a 192.168.10.254 con 32 bytes de datos:
Respuesta desde 192.168.10.254: bytes=32 tiempo=1ms TTL=128
Respuesta desde 192.168.10.254: bytes=32 tiempo=1ms TTL=128
Respuesta desde 192.168.10.254: bytes=32 tiempo=1ms TTL=128
Respuesta desde 192.168.10.254: bytes=32 tiempo=1ms TTL=128

Estadísticas de ping para 192.168.10.254:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
    (0% perdidos),
Tiempos aproximados de ida y vuelta en milisegundos:
    Mínimo = 1ms, Máximo = 1ms, Media = 1ms
```

Figura 4.27: conexión con Balanceo de Carga.

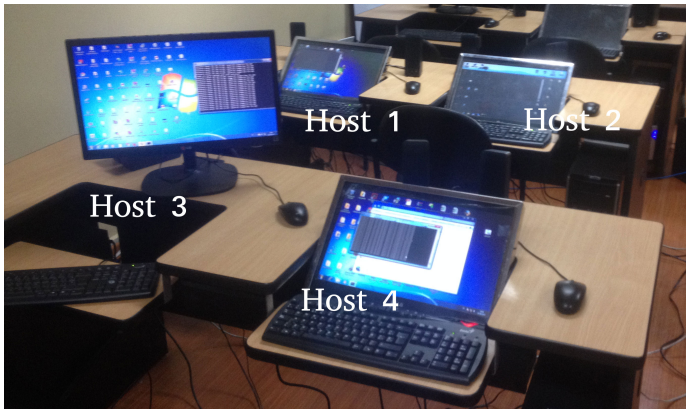


Figura 4.28: Grupo de Host.

Dentro de la página web del controlador se puede verificar los dispositivos conectados al igual que sus direcciones IP y puertos en donde se encuentran conectados como se evidencia en la Figura 4.29.

Switches (2)

DPID	IP Address	Vendor	Packets	Bytes	Flows	Connected Since
00:14:f0:92:1c:b6:f8:40	/192.168.1.21:54323	Nicira, Inc.	19	1862	6	30/5/2015 23:28:05
00:15:f0:a3:2d:c7:a1:50	/192.168.1.21:54322	Nicira, Inc.	196	19208	8	30/5/2015 23:28:05

Hosts (4)

MAC Address	IP Address	Switch Port	Last Seen
00:1c:c0:66:08:b3	192.168.10.50	00:15:f0:a3:2d:c7:a1:50-2	31/5/2015 0:28:42
00:1c:c0:33:3a:b5	192.168.10.51	00:15:f0:a3:2d:c7:a1:50-3	31/5/2015 0:28:57
00:1c:c0:77:09:c4	192.168.10.52	00:15:f0:a3:2d:c7:a1:50-4	31/5/2015 0:29:06
00:1c:c0:88:4b:d6	192.168.10.53	00:15:f0:a3:2d:c7:a1:50-5	31/5/2015 0:28:51

Figura 4.29: Controlador Activo Switchs y Hosts.

La figura 4.30 muestra los flujos que inserta el controlador teniendo una temperatura normal como se detalla en el ítem 6 y 7 del cuadro de temperaturas en la parte inferior de la figura y distribuyendo el tráfico entre los dos servidores principales. El balanceo de carga distribuye todas las peticiones hacia las IPs 192.168.10.10 y 192.168.10.20, estos flujos tienen un tiempo de vida de un minuto, una vez transcurrido este tiempo el controlador consulta la temperatura de los servidores y determinan un nuevo conjunto de flujos.correcto

Flows (8)

Cookie	Priority	Match	Action	Packets	Bytes	Age	Timeout
0	0	port=2, VLAN=-1, src=00:1c:c0:de:0a:e1, dest=00:1c:c0:77:09:c4, ethertype=0x0800, src=192.168.10.20, dest=192.168.10.52, TOS=0	src=00:00:00:00:00:fe, src=-1062728962, output 5	7	686	7 s	60 s
0	0	port=5, VLAN=-1, src=00:1c:c0:77:09:c4, dest=00:00:00:00:00:fe, ethertype=0x0800, src=192.168.10.52, dest=192.168.10.254, TOS=0	dest=00:1c:c0:de:0a:e1, dest=-1062729196, output 2	6	588	7 s	60 s
0	0	port=5, VLAN=-1, src=00:1c:c0:88:4b:d6, dest=00:00:00:00:00:fe, ethertype=0x0800, src=192.168.10.53, dest=192.168.10.254, TOS=0	dest=00:1c:c0:de:08:31, dest=-1062729206, output 1	9	882	9 s	60 s
0	0	port=2, VLAN=-1, src=00:1c:c0:de:0a:e1, dest=00:1c:c0:33:3a:b5, ethertype=0x0800, src=192.168.10.20, dest=192.168.10.51, TOS=0	src=00:00:00:00:00:fe, src=-1062728962, output 5	4	392	3 s	60 s
0	0	port=5, VLAN=-1, src=00:1c:c0:66:08:b3, dest=00:00:00:00:00:fe, ethertype=0x0800, src=192.168.10.50, dest=192.168.10.254, TOS=0	dest=00:1c:c0:de:08:31, dest=-1062729206, output 1	3	294	3 s	60 s
0	0	port=5, VLAN=-1, src=00:1c:c0:33:3a:b5, dest=00:00:00:00:00:fe, ethertype=0x0800, src=192.168.10.51, dest=192.168.10.254, TOS=0	dest=00:1c:c0:de:0a:e1, dest=-1062729196, output 2	3	294	3 s	60 s
0	0	port=1, VLAN=-1, src=00:1c:c0:de:08:31, dest=00:1c:c0:88:4b:d6, ethertype=0x0800, src=192.168.10.10, dest=192.168.10.53, TOS=0	src=00:00:00:00:00:fe, src=-1062728962, output 5	10	980	9 s	60 s
0	0	port=1, VLAN=-1, src=00:1c:c0:de:08:31, dest=00:1c:c0:66:08:b3, ethertype=0x0800, src=192.168.10.10, dest=192.168.10.50, TOS=0	src=00:00:00:00:00:fe, src=-1062728962, output 5	4	392	3 s	60 s

6	28.3 .C	20.1 .C	16.1 .C
7	28.3 .C	20.1 .C	16.1 .C
8	36 .C	37 .C	28 .C
9	34.1 .C	33.2 .C	18.5 .C

Figura 4.30: Flujos de Balanceo Usando Servidores Principales.

En la figura 4.31 se muestra el conjunto de flujos que se insertan cuando la temperatura de uno de los servidores principales supera el rango permitido, como es el caso del ítem 8 y 9 del cuadro de temperaturas mismos que se encuentran resaltados para evidenciar que el módulo activa la distribución de carga hacia las IPs de los servidores principales al igual que hacia las IPs de los servidores de apoyo que son 192.168.10.30 y 192.168.10.40.

Flows (8)

Cookie	Priority	Match	Action	Packets	Bytes	Age	Timeout
0	0	port=5, VLAN=-1, src=00:1c:c0:88:4b:d6, dest=00:00:00:00:00:fe, ethertype=0x0800, src=192.168.10.53, dest=192.168.10.254, TOS=0	dest=00:1c:c0:de:08:2d, dest=-1062729186, output 3	0	0	0 s	60 s
0	0	port=1, VLAN=-1, src=00:1c:c0:de:08:31, dest=00:1c:c0:77:09:c4, ethertype=0x0800, src=192.168.10.10, dest=192.168.10.52, TOS=0	src=00:00:00:00:00:fe, src=-1062728962, output 5	7	686	7 s	60 s
0	0	port=5, VLAN=-1, src=00:1c:c0:77:09:c4, dest=00:00:00:00:00:fe, ethertype=0x0800, src=192.168.10.52, dest=192.168.10.254, TOS=0	dest=00:1c:c0:de:08:31, dest=-1062729206, output 1	6	588	7 s	60 s
0	0	port=2, VLAN=-1, src=00:1c:c0:de:0a:e1, dest=00:1c:c0:33:3a:b5, ethertype=0x0800, src=192.168.10.20, dest=192.168.10.51, TOS=0	src=00:00:00:00:00:fe, src=-1062728962, output 5	5	490	5 s	60 s
0	0	port=5, VLAN=-1, src=00:1c:c0:66:08:b3, dest=00:00:00:00:00:fe, ethertype=0x0800, src=192.168.10.50, dest=192.168.10.254, TOS=0	dest=00:1c:c0:55:2e:b4, dest=-1062729176, output 4	7	686	8 s	60 s
0	0	port=3, VLAN=-1, src=00:1c:c0:de:08:2d, dest=00:1c:c0:88:4b:d6, ethertype=0x0800, src=192.168.10.30, dest=192.168.10.53, TOS=0	src=00:00:00:00:00:fe, src=-1062728962, output 5	1	98	0 s	60 s
0	0	port=4, VLAN=-1, src=00:1c:c0:55:2e:b4, dest=00:1c:c0:66:08:b3, ethertype=0x0800, src=192.168.10.40, dest=192.168.10.50, TOS=0	src=00:00:00:00:00:fe, src=-1062728962, output 5	8	784	8 s	60 s
0	0	port=5, VLAN=-1, src=00:1c:c0:33:3a:b5, dest=00:00:00:00:00:fe, ethertype=0x0800, src=192.168.10.51, dest=192.168.10.254, TOS=0	dest=00:1c:c0:de:0a:e1, dest=-1062729196, output 2	4	392	5 s	60 s

6	28.3 .C	20.1 .C	16.1 .C
7	28.3 .C	20.1 .C	16.1 .C
8	36 .C	37 .C	28 .C
9	34.1 .C	33.2 .C	18.5 .C

Figura 4.31: Flujos de Balanceo Usando todos los Servidores.

Como prueba para el correcto funcionamiento del módulo de balanceo se realizo un ataque de negación de servicios desde uno de los host de la red hacia uno de los servidores, obteniendo un funcionamiento adecuado del módulo al desplazar las peticiones hacia otros servidores sin interrumpir el funcionamiento. La figura 4.32 muestra el conjunto de flujos durante un ataque de negación de servicios y es claramente apreciable el alto numero de paquetes que maneja el controlador en los dos primeros flujos de red.

Flows (8)

Cookie	Priority	Match	Action	Packets	Bytes	Age	Timeout
0	0	port=1, VLAN=-1, prio=0, src=00:1c:c0:de:07:af, dest=08:00:27:f2:7b:a1, ethertype=0x0800, src=192.168.10.40, dest=192.168.10.55, TOS=0	src=00:00:00:00:00:fe, src=-1062728962, output 2	1505	98258	37 s	60 s
0	0	port=2, VLAN=-1, prio=0, src=08:00:27:f2:7b:a1, dest=00:00:00:00:00:fe, ethertype=0x0800, src=192.168.10.55, dest=192.168.10.254, TOS=0	dest=00:1c:c0:de:07:af, dest=-1062729176, output 1	2055	227750	37 s	60 s
0	0	port=1, VLAN=-1, prio=0, src=c0:3f:d5:b8:37:87, dest=00:1c:c0:de:0a:e1, ethertype=0x0800, src=192.168.10.30, dest=192.168.10.52, TOS=0	src=00:00:00:00:00:fe, src=-1062728962, output 2	10	9180	44 s	60 s
0	0	port=2, VLAN=-1, prio=0, src=00:1c:c0:de:0a:e1, dest=00:00:00:00:00:fe, ethertype=0x0800, src=192.168.10.52, dest=192.168.10.254, TOS=0	dest=c0:3f:d5:b8:37:87, dest=-1062729186, output 1	8	1317	44 s	60 s

Figura 4.32: Flujos de datos Negación de Servicios.

La implementación del módulo de Filtro MAC le permite a los host habilitados acceder hacia los distintos servidores de forma directa, la figura 4.33 muestra esta conexión desde uno de los host usando el comando ping hacia varios de los servidores de la red.

```
C:\Windows\system32>ping 192.168.10.10

Haciendo ping a 192.168.10.10 con 32 bytes de datos:
Respuesta desde 192.168.10.10: bytes=32 tiempo=3ms TTL=128
Respuesta desde 192.168.10.10: bytes=32 tiempo=1ms TTL=128
Respuesta desde 192.168.10.10: bytes=32 tiempo=1ms TTL=128
Respuesta desde 192.168.10.10: bytes=32 tiempo<1m TTL=128

Estadísticas de ping para 192.168.10.10:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
    (0% perdidos),
Tiempos aproximados de ida y vuelta en milisegundos:
    Mínimo = 0ms, Máximo = 3ms, Media = 1ms

C:\Windows\system32>ping 192.168.10.20

Haciendo ping a 192.168.10.20 con 32 bytes de datos:
Respuesta desde 192.168.10.254: bytes=32 tiempo=3ms TTL=128
Respuesta desde 192.168.10.254: bytes=32 tiempo=1ms TTL=128
Respuesta desde 192.168.10.254: bytes=32 tiempo=1ms TTL=128
Respuesta desde 192.168.10.254: bytes=32 tiempo=1ms TTL=128

Estadísticas de ping para 192.168.10.20:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
    (0% perdidos),
Tiempos aproximados de ida y vuelta en milisegundos:
    Mínimo = 1ms, Máximo = 3ms, Media = 1ms

C:\Windows\system32>ping 192.168.10.30

Haciendo ping a 192.168.10.30 con 32 bytes de datos:
Respuesta desde 192.168.10.30: bytes=32 tiempo=3ms TTL=128
Respuesta desde 192.168.10.30: bytes=32 tiempo<1m TTL=128
Respuesta desde 192.168.10.30: bytes=32 tiempo<1m TTL=128
Respuesta desde 192.168.10.30: bytes=32 tiempo<1m TTL=128

Estadísticas de ping para 192.168.10.30:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
    (0% perdidos),
Tiempos aproximados de ida y vuelta en milisegundos:
    Mínimo = 0ms, Máximo = 3ms, Media = 0ms
```

Figura 4.33: Filtro MAC Activado.

Al estar activo el módulo de filtro MAC este inserta otro tipo de flujos de datos mismo que se puede apreciar en la figura 4.34.

Flows (8)

Cookie	Priority	Match	Action	Packets	Bytes	Age	Ti
0	-1	port=5, VLAN=-1, src=00:1c:c0:de:08:2d, dest=00:1c:c0:de:07:af, ethertype=0x0800, proto=1, IP src port=8, IP dest port=0, src=192.168.10.50, dest=192.168.10.40, TOS=0	output 4	3	294	2 s	15
0	-1	port=5, VLAN=-1, src=00:1c:c0:de:08:2d, dest=c0:3f:d5:b8:37:87, ethertype=0x0800, proto=1, IP src port=8, IP dest port=0, src=192.168.10.50, dest=192.168.10.30, TOS=0	output 3	14	1372	13 s	15
0	-1	port=4, VLAN=-1, src=00:1c:c0:de:07:af, dest=00:1c:c0:de:08:2d, ethertype=0x0800, proto=1, IP src port=0, IP dest port=0, src=192.168.10.40, dest=192.168.10.50, TOS=0	output 5	3	294	2 s	15
0	-1	port=2, VLAN=-1, src=00:1c:c0:de:08:3a, dest=00:1c:c0:de:08:2d, ethertype=0x0800, proto=1, IP src port=0, IP dest port=0, src=192.168.10.20, dest=192.168.10.50, TOS=0	output 5	21	2058	21 s	15
0	-1	port=1, VLAN=-1, src=c0:3f:d5:b8:39:46, dest=00:1c:c0:de:08:2d, ethertype=0x0800, proto=1, IP src port=0, IP dest port=0, src=192.168.10.10, dest=192.168.10.50, TOS=0	output 5	53	5194	53 s	15
0	-1	port=3, VLAN=-1, src=c0:3f:d5:b8:37:87, dest=00:1c:c0:de:08:2d, ethertype=0x0800, proto=1, IP src port=0, IP dest port=0, src=192.168.10.30, dest=192.168.10.50, TOS=0	output 5	14	1372	13 s	15

Figura 4.34: Flujos Filtro MAC.

Estos nuevos flujos permiten el acceso de los usuarios habilitados a los distintos servicios que poseen los servidores replicados sin la necesidad de atravesar por el balanceo de carga.

Una vez realizadas las pruebas antes descritas se cumple con todos los requerimientos tanto de Hardware y Software que el apartado 4.2 solicita, evidenciando el cambio en los flujos de información de la red propuesta, mismos que dependen de los valores de temperatura registrados.

CAPÍTULO 5

Conclusiones y Recomendaciones

5.1. Conclusiones

Al finalizar el presente proyecto de titulación se obtuvieron las siguientes conclusiones:

- El uso de las Redes Definidas por Software como plataforma de trabajo abre la posibilidad de desarrollo de nuevas aplicaciones más personalizadas o diseñadas para un uso especifico, así mismo permite la integración de nuevos dispositivos y variables a las redes de datos, convirtiendo a las redes de datos en una infraestructura adaptable y dinámica.

- La integración de sensores de temperatura a la red de datos brinda un cierto grado de autonomía de trabajo a la red; debido al control más dinámico que se obtuvo con un balanceo de carga SDN, dependiente de la temperatura que poseían los equipos y así evitando llegar a los rangos de temperatura que perjudiquen el funcionamiento de los servidores.

- El uso de los sistemas de gestión de contenido ayuda en el desarrollo rápido y eficaz de aplicaciones para la visualización de datos, creación de portales WEB, blogs, etc, sin dejar de lado los conceptos básicos de programación WEB como HTML y PHP que en el proyecto permitieron la posterior visualización de los datos de temperatura y la integración del módulo de Filtro MAC.

5.2. Recomendaciones

- El uso de protocolos para conexiones remotas tienen gran utilidad si se plantea una red cuyo controlador se encuentre distante ya que se obtienen una conexión segura y un nivel de privacidad a través de redes publicas como Internet.
- Los distintos equipos de red (switchs o routers) deben poder manejar la misma versión del protocolo openflow que maneja el controlador SDN de la red para mantenerse completamente operativos.
- El uso de herramientas de simulación como mininet permiten obtener resultados bastante acercados a los que se espera de la implementación de módulos o infraestructuras desarrollas ayudando a prever errores.

Bibliografia

[1] F. de Ciencias Exactas, *Sensores de Temperatura.* Sensores de Temperatura. Cordoba: (s. f.).

[2] P. Andrei Khurshudov, *How HDD Workload Impacts TCO.* Cloud Modeling and Data Analytics Seagate Technology, 2013.

[3] A. N. Ramires, *RED DEFINIDA POR SOFTWARE (SDN) EN BASE A UNA INFRAESTRUCTURA DE SOFTWARE DE LIBRE DISTRIBUCION.* Universidad Tecnica de Ambato, 2015.

[4] R. E. Herrador, *GuÃa de Usuario de Arduino.* Cordoba: Universidad de Cordoba-I. T. I. Sistemas, 2011.

[5] A. M. R. C. y A. Pachon, *Software Defined Networks.* Dpto. de TecnologÃas de Informacion y Comunicaciones. Universidad Icesi, 2012.

[6] A. Logicalis, *SDN Como el nuevo Universo trazado por las redes definidas por software impactara en los negocios.* Logicalis Business and Technology Working as One, 2014.

[7] M. Canal, *(21 de Agosto de 2014)*, vol. 21. 2014.

[8] J. Networks and I., *DESCIFRANDO LAS REDES DEFINIDAS POR SOFTWARE.* Sunnyvale, CA 94089 EE. UU, 2012.

[9] A. D. Gante and M. A., *Smart Wireless Sensor Network Management Based on Software Defined Networking.* Mexico:CONACyT, 2014.

[10] D. G. Fueltala, *Implementacion de un prototipo de una red definida por software empleando una solucion basada en software.* Quito-Ecuador: Escuela PolitÃ©cnica Nacional, 2014.

[11] A. G. G, *Documentacion deinfraestructuras para redes definidas por software para el Hospital del Rio.* Cuenca Ecuador: UNIVERSIDAD TECNOLOGICA ISRAEL FACULTAD DE SISTEMAS INFORMATICOS, 2013.

[12] C. J. Carlos, *Implementacion de un prototipo de una red definida por software empleando una solucion basada en Hardware.* Quito Ecuador: Escuela Politecnica Nacional, 2013.

[13] A. G. Carlos, *Despliegue de una maqueta de red basada en openflow.* Universidad de Cantabria, 2014.

[14] O. N. Foundation, *Software-Defined Networking:The New Norm for Networks.* Palo Alto CA: Open Networking Foundation, 2012.

[15] C. E. Rothenberg, *OpenFlow e redes definidas por software: um novo paradigma de controle e inovacao em redes de pacotes.* Cad. CPqD Tecnologia, 2011.

[16] N. G. y Teemu Koponen, *NOX: Towards an Operating System for Networks.* YALE CCS, 2010.

[17] M. Rezazad, *A short walk-through of Mininet and POX.* National University of Singapore., 2010.

[18] D. Erickson, *The Beacon OpenFlow Controller.* Stanford University, 2010.

[19] P. Floodlight, "Project floodlight open source software for building software-defined networks," 2012.

[20] T. I. Incorporated, *LM35 Precision Centigrade Temperature Sensors.* 2013.

[21] G. H. Eben Upton, *Raspberry Pi User Guide.* John Wiley and Sons Ltd, 2012.

[22] A. Hague, *The Raspberry Pi Education Manual.* Computing at School (CAS), 2012.

[23] I. Corporation, *Intel Galileo Board User Guide.* Intel Corporation, 2014.

[24] I. Corporation, *Intel Galileo Datasheet.* Intel Corporation, 2014.

[25] P. Jommla, "Joomla content management system (cms)," 2014.

[26] J. Cuerda Garcia, Xavier y Minguillon Alfonso, *Introduccion a los Sistemas de Gestion de Contenidos (CMS) de codigo abierto.* Universidad Oberta de Catalunya, 2013.

[27] D. Julia, *Desarrollo de portales web con Drupal.* Centre Universitari de Disseny i Art, 2011.

[28] M. P. Martin, *Desarrollo de un sitio web corporativo accesible usando Drupal.* UNIVERSIDAD POLITECNICA DE MADRID FACULTAD DE INFORMATICA, 2012.

[29] J. M. Rivero, *Desarrollo de portales web con Drupal.* HiperMedio Uruguay, 2012.

[30] A. Dominguez, *Manual Basico de Wordpress v3.* CRA DE TEO., 2011.

[31] I. IT, *WordPress User Guide.* Interconnect IT Ltd (UK), 2014.

[32] I. NETGEAR, *Virtual Private Networking Basics.* NETGEAR, Inc, 2010.

[33] J. L. R. Gonzalez, *VPN - Redes Privadas Virtuales.* Universidad de Oviedo, 2002.

[34] Cisco, "(12 de 11 de 2012)," *Configuracion de Tuneles GRE con enrutadores Cisco. Recuperado el*, vol. 2, 2015.

[35] A. G. Morales, *Redes Privadas Virtuales.* Universidad Autonoma del Estado de Hidalgo, 2011.

Anexos

Anexo A

Módulo para Balanceo de Carga Modificado.

```
package net.floodlightcontroller.balanceotesis;

import java.io.IOException;
import java.util.ArrayList;
import java.util.Collection;
import java.util.List;
import java.util.Map;

import org.openflow.protocol.OFFlowMod;
import org.openflow.protocol.OFMatch;
import org.openflow.protocol.OFMessage;
import org.openflow.protocol.OFPacketIn;
import org.openflow.protocol.OFPacketOut;
import org.openflow.protocol.OFPort;
import org.openflow.protocol.OFType;
import org.openflow.protocol.action.OFAction;
import org.openflow.protocol.action.OFActionDataLayerDestination;
import org.openflow.protocol.action.OFActionDataLayerSource;
import org.openflow.protocol.action.OFActionNetworkLayerDestination;
import org.openflow.protocol.action.OFActionNetworkLayerSource;
import org.openflow.protocol.action.OFActionOutput;
import org.openflow.util.U16;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import net.floodlightcontroller.core.FloodlightContext;
import net.floodlightcontroller.core.IFloodlightProviderService;
import net.floodlightcontroller.core.IOFMessageListener;
import net.floodlightcontroller.core.IOFSwitch;
import net.floodlightcontroller.core.IListener.Command;
import net.floodlightcontroller.core.module.FloodlightModuleContext;
import net.floodlightcontroller.core.module.FloodlightModuleException;
```

```
import net.floodlightcontroller.core.module.IFloodlightModule;
import net.floodlightcontroller.core.module.IFloodlightService;
import net.floodlightcontroller.packet.ARP;
import net.floodlightcontroller.packet.Ethernet;
import net.floodlightcontroller.packet.IPacket;
import net.floodlightcontroller.packet.IPv4;


import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
//import java.sql.SQLException;
import java.sql.Statement;
/**
 * Module to perform round-robin load balancing.
 *
 */
public class BalanceotesisSW implements IOFMessageListener,
    IFloodlightModule {
    ///////Variables Globales/////
    double temS1=0;
    double temS2=0;
    ////////////////////////////
    // Interface to Floodlight core for interacting with connected
        switches
    protected IFloodlightProviderService floodlightProvider;

    // Interface to the logging system
    protected static Logger logger;

    // IP and MAC address for our logical load balancer
    private final static int LOAD_BALANCER_IP = IPv4.toIPv4Address("
        192.168.10.254");
    private final static byte[] LOAD_BALANCER_MAC = Ethernet.
        toMACAddress("00:00:00:00:00:FE");

    // Rule timeouts
    private final static short IDLE_TIMEOUT = 60; // in seconds
    private final static short HARD_TIMEOUT = 0; // infinite

    private static class Server
    {
        private int ip;
        private byte[] mac;
        private short port;
```

```
    public Server(String ip, String mac, short port) {
        this.ip = IPv4.toIPv4Address(ip);
        this.mac = Ethernet.toMACAddress(mac);
        this.port = port;
    }

    public int getIP() {
        return this.ip;
    }

    public byte[] getMAC() {
        return this.mac;
    }

    public short getPort() {
        return this.port;
    }
}

final static Server[] SERVERS = {
new Server("192.168.10.10","c0:3f:d5:b8:39:46", (short)1),
new Server("192.168.10.20","00:1c:c0:de:08:3a", (short)2),
new Server("192.168.10.30","c0:3f:d5:b8:37:87", (short)3),
new Server("192.168.10.40","00:1c:c0:de:07:af", (short)4)
};
private int lastServer = 0;

/**
 * Provides an identifier for our OFMessage listener.
 * Important to override!
 * */
@Override
public String getName() {
    return BalanceotesisSW.class.getSimpleName();
}

@Override
public boolean isCallbackOrderingPrereq(OFType type, String name) {
    // Auto-generated method stub
    return false;
}

@Override
```

```
public boolean isCallbackOrderingPostreq(OFType type, String name)
    {
    // Auto-generated method stub
    return false;
}

@Override
public Collection<Class<? extends IFloodlightService>>
    getModuleServices() {
    // Auto-generated method stub
    return null;
}

@Override
public Map<Class<? extends IFloodlightService>, IFloodlightService>
    getServiceImpls() {
    // Auto-generated method stub
    return null;
}

/**
 * Tells the module loading system which modules we depend on.
 * Important to override!
 */
@Override
public Collection<Class<? extends IFloodlightService>>
    getModuleDependencies() {
    Collection<Class<? extends IFloodlightService >>
        floodlightService =
        new ArrayList<Class<? extends IFloodlightService>>();
    floodlightService.add(IFloodlightProviderService.class);
    return floodlightService;
}

/**
 * Loads dependencies and initializes data structures.
 * Important to override!
 */
@Override
public void init(FloodlightModuleContext context)
        throws FloodlightModuleException {
    floodlightProvider = context.getServiceImpl(
        IFloodlightProviderService.class);
    logger = LoggerFactory.getLogger(BalanceotesisSW.class);
}
```

```
/**
 * Tells the Floodlight core we are interested in PACKET_IN
    messages.
 * Important to override!
 * */
@Override
public void startUp(FloodlightModuleContext context) {
    floodlightProvider.addOFMessageListener(OFType.PACKET_IN, this)
        ;
}

/**
 * Receives an OpenFlow message from the Floodlight core and
    initiates the appropriate control logic.
 * Important to override!
 */
@Override
public net.floodlightcontroller.core.IListener.Command receive(
        IOFSwitch sw, OFMessage msg, FloodlightContext cntx) {
    ////////////////////////////////////////
    if (sw.getId() != 5894009871857728L) {
        // Allow the next module to also process this OpenFlow
            message
        return Command.CONTINUE;
    }
    //System.out.println("DPID ="+sw.getId());
    ////////////////////////////////////////

    // We only care about packet-in messages
    if (msg.getType() != OFType.PACKET_IN) {
        // Allow the next module to also process this OpenFlow
            message

        return Command.CONTINUE;
    }
    OFPacketIn pi = (OFPacketIn)msg;

    // Parse the received packet
    OFMatch match = new OFMatch();
    match.loadFromPacket(pi.getPacketData(), pi.getInPort());

    // We only care about TCP packets
    if (match.getDataLayerType() != Ethernet.TYPE_IPv4 && match.
        getDataLayerType() != Ethernet.TYPE_ARP) {
```

```
            // Allow the next module to also process this OpenFlow
                message
            return Command.CONTINUE;
        }

        // We only care about packets which are sent to the logical
            load balancer
        if (match.getNetworkDestination() != LOAD_BALANCER_IP) {
            // Allow the next module to also process this OpenFlow
                message
            return Command.CONTINUE;
        }

        if (match.getDataLayerType() == Ethernet.TYPE_ARP) {

            // Receive an ARP request
            logger.info("Received an ARP request for the load balancer"
                );
            handleARPRequest(sw, pi, cntx);

        } else {

            logger.info("Received an IPv4 packet destined for the load
                balancer");
            loadBalanceFlow(sw, pi, cntx);
        }

        // Do not continue processing this OpenFlow message
        return Command.STOP;
    }

    /**
     * Sends a packet out to the switch
     */
    private void pushPacket(IOFSwitch sw, OFPacketIn pi,
            ArrayList<OFAction> actions, short actionsLength) {

        // create an OFPacketOut for the pushed packet
        OFPacketOut po = (OFPacketOut) floodlightProvider.
            getOFMessageFactory()
                        .getMessage(OFType.PACKET_OUT);

        // Update the inputPort and bufferID
        po.setInPort(pi.getInPort());
        po.setBufferId(pi.getBufferId());
```

```
        // Set the actions to apply for this packet
        po.setActions(actions);
        po.setActionsLength(actionsLength);

        // Set data if it is included in the packet in but buffer id is
            NONE
        if (pi.getBufferId() == OFPacketOut.BUFFER_ID_NONE) {
            byte[] packetData = pi.getPacketData();
            po.setLength(U16.t(OFPacketOut.MINIMUM_LENGTH
                    + po.getActionsLength() + packetData.length));
            po.setPacketData(packetData);
        } else {
            po.setLength(U16.t(OFPacketOut.MINIMUM_LENGTH
                    + po.getActionsLength()));
        }

        // Push the packet to the switch
        try {
            sw.write(po, null);
        } catch (IOException e) {
            logger.error("failed to write packetOut: ", e);
        }
    }

    /**
     * Handle ARP Request and reply it with load balancer's MAC address
     */
    private void handleARPRequest(IOFSwitch sw, OFPacketIn pi,
        FloodlightContext cntx) {

        logger.debug("Handle ARP request");
        Ethernet eth = IFloodlightProviderService.bcStore.get(cntx,
                IFloodlightProviderService.CONTEXT_PI_PAYLOAD);
        if (! (eth.getPayload() instanceof ARP))
            return;
        ARP arpRequest = (ARP) eth.getPayload();

        // generate ARP reply
        IPacket arpReply = new Ethernet()
            .setSourceMACAddress(BalanceotesisSW.LOAD_BALANCER_MAC)
            .setDestinationMACAddress(eth.getSourceMACAddress())
            .setEtherType(Ethernet.TYPE_ARP)
            .setPriorityCode(eth.getPriorityCode())
            .setPayload(
```

```
                new ARP()
                .setHardwareType(ARP.HW_TYPE_ETHERNET)
                .setProtocolType(ARP.PROTO_TYPE_IP)
                .setHardwareAddressLength((byte) 6)
                .setProtocolAddressLength((byte) 4)
                .setOpCode(ARP.OP_REPLY)
                .setSenderHardwareAddress(BalanceotesisSW.
                    LOAD_BALANCER_MAC)
                .setSenderProtocolAddress(BalanceotesisSW.
                    LOAD_BALANCER_IP)
                .setTargetHardwareAddress(arpRequest.
                    getSenderHardwareAddress())
                .setTargetProtocolAddress(arpRequest.
                    getSenderProtocolAddress()));

        sendARPReply(arpReply, sw, OFPort.OFPP_NONE.getValue(), pi.
            getInPort());
    }

    /**
     * Sends ARP reply out to the switch
     */
    private void sendARPReply(IPacket packet, IOFSwitch sw, short
        inPort, short outPort) {

        // Initialize a packet out
        OFPacketOut po = (OFPacketOut) floodlightProvider.
            getOFMessageFactory()
                    .getMessage(OFType.PACKET_OUT);
        po.setBufferId(OFPacketOut.BUFFER_ID_NONE);
        po.setInPort(inPort);

        // Set output actions
        List<OFAction> actions = new ArrayList<OFAction>();
        actions.add(new OFActionOutput(outPort, (short) 0xffff));
        po.setActions(actions);
        po.setActionsLength((short) OFActionOutput.MINIMUM_LENGTH);

        // Set packet data and length
        byte[] packetData = packet.serialize();
        po.setPacketData(packetData);
        po.setLength((short) (OFPacketOut.MINIMUM_LENGTH + po.
            getActionsLength() + packetData.length));

        // Send packet
```

```
        try {
            sw.write(po, null);
            sw.flush();
        } catch (IOException e) {
            logger.error("Failure writing packet out", e);
        }
    }

    /**
     * Performs load balancing based on a packet-in OpenFlow message
        for an
     * IPv4 packet destined for our logical load balancer.
     */
    private void loadBalanceFlow(IOFSwitch sw, OFPacketIn pi,
        FloodlightContext cntx) {

        Server server = getNextServer();

        Ethernet eth = IFloodlightProviderService.bcStore.get(cntx,
                IFloodlightProviderService.CONTEXT_PI_PAYLOAD);

        // Create a flow table modification message to add a rule
        OFFlowMod rule = new OFFlowMod();
        rule.setType(OFType.FLOW_MOD);
        rule.setCommand(OFFlowMod.OFPFC_ADD);

        // Create match
        OFMatch match = new OFMatch()
            .setDataLayerDestination(LOAD_BALANCER_MAC)
            .setDataLayerSource(eth.getSourceMACAddress())
            .setDataLayerType(Ethernet.TYPE_IPv4)
            .setNetworkDestination(LOAD_BALANCER_IP)
            .setNetworkSource(((IPv4) eth.getPayload()).
                getSourceAddress())
            .setInputPort(pi.getInPort());

        // Set wildcards for Network protocol
        match.setWildcards(OFMatch.OFPFW_NW_PROTO);
        rule.setMatch(match);

        // Specify the timeouts for the rule
        rule.setIdleTimeout(IDLE_TIMEOUT);
        rule.setHardTimeout(HARD_TIMEOUT);

        // Set the buffer id to NONE -- implementation artifact
```

```
rule.setBufferId(OFPacketOut.BUFFER_ID_NONE);

// Initialize list of actions
ArrayList<OFAction> actions = new ArrayList<OFAction>();

// Add action to re-write destination MAC to the MAC of the
    chosen server
OFAction rewriteMAC = new OFActionDataLayerDestination(server.
    getMAC());
actions.add(rewriteMAC);

// Add action to re-write destination IP to the IP of the
    chosen server
OFAction rewriteIP = new OFActionNetworkLayerDestination(server
    .getIP());
actions.add(rewriteIP);

// Add action to output packet
OFAction outputTo = new OFActionOutput(server.getPort());
actions.add(outputTo);

// Add actions to rule
rule.setActions(actions);
short actionsLength = (short)(OFActionDataLayerDestination.
    MINIMUM_LENGTH
        + OFActionNetworkLayerDestination.MINIMUM_LENGTH
        + OFActionOutput.MINIMUM_LENGTH);

// Specify the length of the rule structure
rule.setLength((short) (OFFlowMod.MINIMUM_LENGTH +
    actionsLength));

logger.debug("Actions length="+ (rule.getLength() - OFFlowMod.
    MINIMUM_LENGTH));

logger.debug("Install rule for forward direction for flow: " +
    rule);

try {
    sw.write(rule, null);
} catch (Exception e) {
    e.printStackTrace();
}
```

```
// Create a flow table modification message to add a rule for
    the reverse direction
OFFlowMod reverseRule = new OFFlowMod();
reverseRule.setType(OFType.FLOW_MOD);
reverseRule.setCommand(OFFlowMod.OFPFC_ADD);

// Create match
OFMatch reverseMatch = new OFMatch()
    .setDataLayerSource(server.getMAC())
    .setDataLayerDestination(match.getDataLayerSource())
    .setDataLayerType(Ethernet.TYPE_IPv4)
    .setNetworkSource(server.getIP())
    .setNetworkDestination(match.getNetworkSource())
    .setInputPort(server.getPort());

// Set wildcards for Network protocol
reverseMatch.setWildcards(OFMatch.OFPFW_NW_PROTO);
reverseRule.setMatch(reverseMatch);

// Specify the timeouts for the rule
reverseRule.setIdleTimeout(IDLE_TIMEOUT);
reverseRule.setHardTimeout(HARD_TIMEOUT);

// Set the buffer id to NONE -- implementation artifact
reverseRule.setBufferId(OFPacketOut.BUFFER_ID_NONE);

// Initialize list of actions
ArrayList<OFAction> reverseActions = new ArrayList<OFAction>();

// Add action to re-write destination MAC to the MAC of the
    chosen server
OFAction reverseRewriteMAC = new OFActionDataLayerSource(
    LOAD_BALANCER_MAC);
reverseActions.add(reverseRewriteMAC);

// Add action to re-write destination IP to the IP of the
    chosen server
OFAction reverseRewriteIP = new OFActionNetworkLayerSource(
    LOAD_BALANCER_IP);
reverseActions.add(reverseRewriteIP);

// Add action to output packet
OFAction reverseOutputTo = new OFActionOutput(pi.getInPort());
reverseActions.add(reverseOutputTo);
```

```
        // Add actions to rule
        reverseRule.setActions(reverseActions);

        // Specify the length of the rule structure
        reverseRule.setLength((short) (OFFlowMod.MINIMUM_LENGTH
                + OFActionDataLayerSource.MINIMUM_LENGTH
                + OFActionNetworkLayerSource.MINIMUM_LENGTH
                + OFActionOutput.MINIMUM_LENGTH));

        logger.debug("Install rule for reverse direction for flow: " +
            reverseRule);

        try {
            sw.write(reverseRule, null);
        } catch (Exception e) {
            e.printStackTrace();
        }

        pushPacket(sw, pi, actions, actionsLength);
    }

    /**
     * Determines the next server to which a flow should be sent.
     */


    private Server getNextServer() {

        ////////////////////////////////////
        try  {
            Connection conn = DriverManager.getConnection("jdbc:mysql
                ://192.168.1.20/tesis","root","andres");
            //System.out.println("Temperatura Actual de los Servidores
                ");

            String query = "SELECT * FROM datos";
            Statement stmt = conn.createStatement();
            ResultSet rs = stmt.executeQuery(query);

                if(rs.last()){
                    temS1=rs.getDouble("sensor1");
                    temS2=rs.getDouble("sensor2");
                }
                conn.close();
            } catch (Exception e){
```

```
                System.err.println(e);
            }
        ////////////////////////////////////////

        if(temS1>30|| temS2>30){
            lastServer = (lastServer + 1) % 4;
            System.out.println("Sensor 1 ="+temS1);
            System.out.println("Senser 2 ="+temS2);
        }else{
            lastServer = (lastServer + 1) % 2;
            System.out.println("Sensor 1 ="+temS1);
            System.out.println("Senser 2 ="+temS2);
        }
        return SERVERS[lastServer];
    }

}
```

Anexo B

Código de Programación para plataforma Arduino.

```
#include <SPI.h>          //Libreria Principal Arduino
#include <Ethernet.h>     //Libreria Ethernet
//Direccion Mac De la Shield Ethernet
byte mac[] = { 0x90, 0xA2, 0xDA, 0x0D, 0xAB, 0x04 };
//pin Analogico para el sensor 1
int ST1= A0;
//pin Analogico para el sensor 2
int ST2 = A1;
//pin Analogico para el sensor 3
int ST3 = A2;
//Pin de salidad para leds indicadores
int PWMpin = 3;
//Declaracion de la variable sensor1
double sensor1=0;
//Declaracion de la variable sensor2
double sensor2=0;
//Declaracion de la variable sensor3
double sensor3=0;
//Declaracion de la variable Last para comparar
double last=0;

//Definir a Arduino como Cliente
EthernetClient client;
//Ip para Arduino Shield Ethernet
IPAddress ip(192,168,1,200);
//Ip del Servidor al qye se Conectara
char serverName[] = "192.168.1.20";
//Intervalo de tiempo para respuesta
const unsigned long requestInterval = 6000;
unsigned long lastAttemptTime = 0;
boolean requested;
```

```
//Configuracion de Arduino void setup() {
  //Velocidad de Com. Serial
  Serial.begin(9600);
  //Modo de trabajo del Pin 2
  pinMode(2,OUTPUT);
  //Modo de trabajo del Pin 13
  pinMode(13, OUTPUT);
  //Configurar las direccionres de Arduino
  Ethernet.begin(mac,ip);
  //Imprime la direccion Ip de Arduino
  Serial.print("My IP address:");
    for (byte thisByte = 0; thisByte < 4; thisByte++) {
         Serial.print(Ethernet.localIP()[thisByte], DEC);
    Serial.print(".");
        }
     Serial.println();
    Serial.println(Ethernet.localIP());
    }
void loop() {
  //Temperatura del Sensor 1
  sensor1 =  (( 5.0 * analogRead(ST1) * 100.0) / 1024.0);
  //Temperatura del Sensor 2
  sensor2 = (( 5.0 * analogRead(ST2) * 100.0) / 1024.0);
  //temperatura del Sensor 3
  sensor3 = (( 5.0 * analogRead(ST3) * 100.0) / 1024.0);
  if(sensor1 != last){
     if(sensor1>=(last-3)&& sensor1<=(last+3)){
       Serial.println("El valor de temperatura se Mantiene ");
     }
     else{
          Serial.println("El valor de temperatura a cambiado ");
          connectToServer(sensor1,sensor2,sensor3);
      }
     }
    last=sensor1;
}

//método de conexión con la Base de Datos
void connectToServer(double sensor1, double sensor2, double sensor3 ) {
  //Ciclo para indicar la conexión con el servidor
  if (client.connect(serverName, 80)) {
    digitalWrite(13, HIGH);
         for (int i=0; i <= 255; i++){
        analogWrite(PWMpin, i);
         analogWrite(5, (255-i));//opcional
```

```
      delay(10);
         }
   //Imprime es tado de Peticion
   Serial.println("Realizando peticion HTML...");
   //Localiza la variable que almacenara el valor
   client.print("GET /joomla/datos/datos.php?sensor1=");
   //Almacena el valor del cada sensor
   client.print(sensor1);
   client.print("&sensor2=");
   client.print(sensor2);
   client.print("&sensor3=");
   client.print(sensor3);
   //Indica el formato html
   client.println(" HTTP/1.1");
   //Indica la direccion del cliente
   client.println("HOST: 192.168.1.200");
   client.println();
   //Se imprime todos los datos almacenados
   Serial.print("GET /cleantype/datos.php?sensor1=");
         Serial.print(sensor1);
   Serial.print("&sensor2=");
   Serial.print(sensor2);
   Serial.print("&sensor3=");
   Serial.print(sensor3);
      }
//Ciclo que indica la no conexión con el servidor
else{
         for(int j=0;j<=5;j++){
              digitalWrite(13, HIGH);
       digitalWrite(6, LOW);
       delay(500);
     digitalWrite(6, HIGH);
       digitalWrite(13, LOW);
         delay(500);
       }
    }
//Lapso de tiempo con el q se entregan los datos al Servidor
lastAttemptTime = millis();
client.stop();
client.flush();
delay(3000);
  }
```

Anexo C

Archivo de conexión Almacenado en el Servidor.

C.1. Arduino.php

```
<html>
        <head>
                <title>Recuperando datos de Mysql desde PHP</title>
        </head>
        <body>
                <div>
                        <fieldset>
                                <legend> SENSORES ARDUINO  </legend>
                                <div>
                                        <?php
                                            include("conexión-arduino.
                                                php");
                                            $Con = new conexion();
                                            $Con->recuperarDatos();
                                        ?>
                                </div>
<!-- end div#wrapper -->

                        </fieldset>
                </div>
        </body>
        <footer>
        </footer>
</html>
```

Anexo D

Archivo de conexión Almacenado en el Servidor.

D.1. clase_conexión.php

```
<?php
class conexion{
var $serv="localhost";
var $usuario="root";
var $contra="andres";
var $conexi;
function conecta()
{
$s=$this->serv;
$u=$this->usuario;
$c=$this->contra;
$conex=mysql_connect($s,$u,$c);
$this->conexi=$conex;
}

}
$cono= new conexion();
$cono->conecta();
$c=$cono->conexi;
$select=mysql_select_db("tesis",$c);
?>
```

Anexo E

Archivo de conexión Almacenado en el Servidor.

E.1. conexion-arduino.php

```
<?php
        class conexion{
                function recuperarDatos(){
                        $host = "localhost";
                        $user = "root";
                        $pw = "andres";
                        $db = "tesis";

                        $con = mysql_connect($host, $user, $pw) or die(
                            "No se pudo conectar a la base de datos ");
                        mysql_select_db($db, $con) or die ("No se
                            encontro la base de datos. ");
                        $query = "SELECT * FROM datos";
                        $resultado = mysql_query($query);



                        while ($fila = mysql_fetch_array($resultado)) {
                        echo " <tr>";
                        echo "ID=<td> $fila[id]</td> | SENSOR 1 = <td>
                            $fila[sensor1]</td> ºC | SENSOR 2 = <td>
                            $fila[sensor2]</td> ºC | SENSOR 3 = <td>
                            $fila[sensor3]</td> ºC <br>";
                                 echo " </tr> ";
                        }

                }
        }
?>
```

Anexo F

Archivo de conexión Almacenado en el Servidor.

F.1. datos.php

```
<?php

include($_SERVER['DOCUMENT_ROOT'] ."/joomla/datos/clase_conexion.php");
//$id_tiempo=date('YmdHis');//extraemos la fecha del servidor
$type=$_GET['sensor1'];
$type1=$_GET['sensor2'];
$type2=$_GET['sensor3'];


$consulta="insert into datos values (NULL ,'".$type."','".$type1."','".
    $type2."')";
$inserta_foto=mysql_query($consulta,$c);
?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://
    www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>DATOS FINALES ARDUINO</title>
<style type="text/css">
#apDiv1 {
        position: absolute;
        left: 200px;
        top: 110px;
        width: 475px;
        height: 233px;
        z-index: 1;
}
</style>
```

```
</head>

<body>
<div id="apDiv1">
  <table width="457" height="270" border="1">
    <tr>
      <td width="86">Sensor 1</td>
      <td width="355"><label for="sensor1"><?php echo $_GET['sensor1'
          ];?> </label></td>
    </tr>
    <tr>
      <td>Sensor 2</td>
      <td><label for="sensor2"><?php echo $_GET['sensor2'];?></label></
          td>
    </tr>
    <tr>
      <td>Sensor 3</td>
      <td><label for="sensor3"><?php echo $_GET['sensor3'];?></label></
          td>
    </tr>
    <tr>
      <td> </td>
      <td> </td>
    </tr>
  </table>
</div>
</body>
</html>
```