CONTROL SEGURO DE UNA RED DE SENSORES/ACTUADORES DOMESTICOS

Maryori Del Carmen Sabalza Mejia
Director: Juan Carlos Martínez Santos

Universidad Tecnológica de Bolívar
Facultad de Ingenierías
Programa de Ingeniería Eléctronica
Cartagena

Junio de 2016

CONTROL SEGURO DE UNA RED DE SENSORES/ACTUADORES DOMESTICOS

Maryori Del Carmen Sabalza Mejia

Trabajo de grado para optar al título de

Ingeniera Electrónica

Director: Juan Carlos Martínez Santos

Universidad Tecnológica de Bolívar
Facultad de Ingenierías
Cartagena

Junio de 2016

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR

FACULTAD DE INGENIERÍAS

Título: Control Seguro de una red de Sensores/Actuadores domesticos

Autor: Maryori Del Carmen Sabalza Mejia

Jurado

Jurado

Director:

Cartagena, Junio de 2016

Resumen

El presente trabajo tiene como objetivo obtener un sistema hardware-software que permita controlar dispositivos eléctricos y/o electrónicos desde una aplicación web de una forma segura, que garantice niveles mínimos de autenticación, confidencialidad, integridad, disponibilidad, control de acceso, autenticación y el no repudio. Para el desarrollo de la aplicación se trabajo con plataformas abiertas con el fin de demostrar la implementación de un sistema de bajo costo seguro para controlar remotamente nuestros objetos conectados a la red. Los objetos o cosas conectados a internet se conocen como Internet de las cosas y por tanto, la seguridad debe estar al mismo nivel o mayor prioridad que aspectos tales como la velocidad, el tamaño y el consumo de energía. Como resultados preliminares de este trabajo, desarrollamos una aplicación con control de acceso que implementa la detección de movimiento, temperatura, consumo de potencia eléctrica y control de luces.

Palabras claves: Internet de las cosas, seguridad, integridad, confiabilidad, disponibilidad, autenticacion, control de acceso y no repudio.

Abstract

This project present the study and implementation of a secure network in order to control devices at home through the web. The basis of this research is that we are now in the era of the Internet of Things and therefore, security must be at the same level or higher priority than aspects such as speed, size, and power consumption. The objective of this work is to obtain a hardware/software system that allows controlling electronic devices from a web application in a secure way, which guarantees minimum levels of confidentiality, integrity, availability, access control, authentication, and non repudiation. For implementation, we rely on open platforms such as Arduino. As preliminary results for this work, we develop an application with access control that implements motion detection and light control.

Keywords: Security, Internet of Things, Low-cost, Hardware/Software, Open source, confidentiality, integrity, availability, access control, authentication, and non repudiation.

Agradecimientos

Agradezco a Dios, a mi madre Alicia, a mi prima Merlys y a mi Tita por preocuparse siempre por mi educación, por su inmenso apoyo y por recordarme que siempre se puede ser mejor. A mi hermano por preguntarme todos los días cuando me iba a graduar, a la pequeña Mayra por hacerme sonreír y demás familiares.

A mis compañeros y amigos, los que siempre me escuchaban cuando les hablaba de este proyecto.

Al Ingeniero Ivan Baños Delgado por su gran trabajo en el funcionamiento de la aplicacion web.

A todos mis profesores que a lo largo de estos años contribuyeron a mi crecimiento profesional.

A mi maestro y asesor Juan Carlos Martinez Santos por creer en mi, por su apoyo y paciencia a lo largo de este camino.

Índice general

Lista de Figuras

Lista de Tablas

Capítulo 1.
Introducción

Actualmente existen mas cosas que personas conectadas a internet debido a que las personas han optado por controlar su casa desde un ordenador o teléfono celular porque quieren encender/apagar las luces, el aire acondicionado, revisar el estado de las alarmas, entre otras cosas, en cualquier momento. sin embargo, tienen que estar seguros de que los usuarios no deseados no manipulan los objetos alrededor de su casa.

Debemos tener en cuenta que en cada segundo transcurrido viajan miles y miles de bytes en Internet los cuales contienen cualquier tipo de información, como por ejemplo nuestro nombre, dirección, intereses, paginas frecuentadas entre otras, que puede ser usurpada por personas maliciosas. Por esta razón el uso de un antivirus es necesario, pero no siempre esta medida es suficiente. Existen diferentes formas de seguridad que se pueden implementar para proteger nuestros equipos ya sea con el firewall, no ingresar a enlaces de bancos nuestros datos personales o no descargar programas de páginas no confiables, sin embargo, que podemos hacer cuando somos nosotros mismos los que nos abrimos al mundo a través de los estos dispositivos conectados a internet todo el tiempo.

1.0.1. Planteamiento del problema

En el " boom "de la Internet de las cosas (IoT), los desarrolladores han diseñado e implementado multiples plataformas que nos permiten controlar nuestra casa a traves de internet, usando un micro-controlador. Entre estas plataformas vale la pena mencionar a las compañías como DeviceHub.net[1] y Nearbus[2] las cuales se encargan de la prestación de servicios que permiten a los usuarios conectarse y controlar sus objetos de la casa a través de sus plataformas a cambio de una cuota mensual. Sin embargo, todavía hay cuestiones sobre la seguridad de la información almacenada en sus servidores y lo que pueden hacer para protegerlo de la explotada por un intruso.

Como se presenta en el trabajo de Hummen[45] " Un atacante tiene acceso a la red después de participar en la estructura de enrutamiento. a partir de ahí, él o ella puede enviar mensajes a cualquier nodo de red según el lugar donde el atacante es con ese tipo de acceso que puede retrasar, reordenar o eliminar paquetes legítimos."

Por otra parte una entrevista realizada en el 2014 un hacker comento lo siguiente: "Cada atacante tiene sus propias razones para robar la información. Muchas veces encontramos desde los esposos o esposas celosas que quieren entrar a los perfiles de sus parejas, también por venganza de un usuario hacia otro, personas que juegan cierto tipo de juegos online y quieren robar acceso a cuentas y poder robar todo su contenido, hasta personas que sólo lo hacen por ocio."[47].

[1]DeviceHub.net es una plataforma integrada para el desarrollo de proyectos de IO.

[2]Nearbus es un conector en la nube que permite a los usuarios integrar plenamente en las diferentes plataformas en la nube como MCU Arduino, OpenPicus, etc.

Por esta razón, el problema gira en torno a este trabajo es cómo asegurar que un sistema de bajo costo de hardware/software cumple con los niveles de seguridad aceptables? ¿Podría un sistema basado en microcontrolador soportar los niveles de encriptación necesarios para enviar información a través de una red?

Basándonos en las razones anteriores optamos por desarrollar una aplicación web que permita controlar dispositivos remotamente de una manera segura.

1.0.2. Justificación

Plataformas al alcance de todos como lo es Arduino uno ha permitido que mas personas investiguen y desarrollen aplicaciones en esta plataforma. Teniendo conocimientos básicos de lenguajes de programación web, protocolo Ethernet y desarrollo de aplicaciones web cientos de personas podrían controlar sus dispositivos desde internet, sin embargo que les garantiza a estas personas que nadie entrara a su sistema a manipular dichos dispositivos.

Para el desarrollo de la aplicación se trabajo con plataformas abiertas con el fin de demostrar en un sistema de bajo costo la implementación de un sistema seguro para controlar remotamente nuestros objetos conectados a la red. Los objetos o cosas conectados a internet se conocen como Internet de las cosas y por ende la seguridad debe ser una prioridad al momento de trabajar con este concepto.

1.0.3. Objetivos

El principal objetivo de este trabajo es obtener un sistema hardware-software que permita controlar dispositivos electrónicos desde una aplicación web de una forma segura, que garantice niveles mínimos de autenticación, confidencialidad, integridad, disponibilidad, control de acceso, autenticación y el no repudio.

Para cumplir este objetivo principal se proponen en especifico los siguientes objetivos:

- Diseñar una aplicación web que permita acceso seguro al sistema a través de una Id/Password de los usuarios autorizados.
- Obtener una red de dispositivos (sensores-actuadores) que permita su monitoreo y control remoto.
- Implementar un protocolo de comunicaciones que permita la interacción de forma segura entre sensores y actuadores con la aplicación web.

Capítulo 2.
Marco teórico

El concepto del Internet de las cosas (IoT) fue propuesto por Kevin Ashton en el Auto-ID Center del MIT en 1999 y se refiere a la interconexión digital de objetos cotidianos con Internet [5]. Alternativamente, Internet de las cosas es el punto en el tiempo en el que se conectarían a Internet más "cosas u objetos" que personas. Una de la principales ventajas que presenta esta tecnología es la capacidad de poder ser aplicada a numerosos sectores y áreas de diferente índole como el sector sanitario, militar, empresas productivas y de servicio, agricultura y ganadería, gobiernos, centros de educación y formación, etc. Sectores de todo el planeta se beneficiará de las múltiples ventajas que presenta el uso del internet de las cosas. Es valido mencionar que el control remoto de múltiples aparatos, presentados como una solución de tecnología aplicada al hogar, se conoce como Domótica [27] y va de la mano del concepto del internet de las cosas sin embargo no debe confundirse, pues las áreas donde es aplicable el IoT va mas allá del hogar.

La introducción de internet de las cosas está siendo una realidad, las tecnologías de software y hardware en las que se apoya están totalmente desarrolladas y listas para implantarse alrededor del planeta, tecnologías como: Big data, Business Intelligence,

Analytics, Cloud computing, dispositivos Wearables, etiquetas RFID, fibra óptica, comunicaciones Wireless, Smart cities, etcétera. Lo mencionado anteriormente son ejemplos de tecnologías disponibles y que forman parte del internet de las cosas.

2.1. Estándares de seguridad

Hay muchos elementos, tales como sensores, actuadores y micro-controladores, que se pueden utilizar para controlar aparatos de nuestro hogar a través de Internet, y múltiples micro-controladores están ahí para ponerlo en práctica. Sin embargo, lo que nunca debe faltar en estos sistemas, es la seguridad. Por lo que debe tenerse en cuenta que existen normas y requisitos de estos sistemas de seguridad con el fin de garantizar que sólo los usuarios autorizados pueden supervisar y controlar sus sistemas domésticos. Las dos normas siguientes se utilizan como referencia en nuestro diseño final.

2.1.1. FIPS 199

Es un estándar establecido por el gobierno de los Estados Unidos, emitido por el Instituto Nacional de Estándares y Tecnología (NIST) [40]. Esta norma establece las categorías de seguridad para los sistemas de información, que son: la confidencialidad, integridad y disponibilidad.

El FIPS 199 contiene tres niveles de potencial impacto en las organizaciones o individuos cuando existe una pérdida de las tres categorías de seguridad, estos niveles son: bajo, moderado y alto los cuales podrían variar según el contexto de la

organización o el interés del país.

- Impacto bajo: Aunque la organización es capaz de seguir realizando sus funciones primarias la eficacia de estas se puede ver reducida, también existirán menores pérdidas económicas.

- Impacto moderado: si la pérdida de confidencialidad, integridad o disponibilidad es moderada se puede esperar una reducción eficaz en las funciones realizadas por la organización o individuo, existirán daños significativos e importantes pérdidas financieras.

- Impacto Alto: esta impacto muestra la mayor pérdida de confidencialidad, integridad o disponibilidad por la cual la organización no es capaz de realizar sus funciones primarias provocando daños graves y económicos. En los individuos se puede tratar de perdida de la vida.

2.1.2. ISO 7498-2

Es un estándar que define un servicio de seguridad con el fin de satisfacer la comunicación entre sistemas abiertos o transferencias de datos en este tipo de sistemas [52]. Las categorías de esta norma son: autenticación, confidencialidad, control de acceso, integridad y no repudio. Las categorías de este estándar son:

- Autenticación: se garantiza que los usuarios son quienes dicen ser

- Control de acceso: se evita el uso no autorizado.

- Confidencialidad: garantiza que la información no va a ser revelada, ni estará disponible para personas fuera del sistema.
- Integridad: se garantiza que la información no será duplicada, divulgada, modificada por terceros.
- No repudio: se asegura que la información fue enviada y recibida por quien se esperaba.

2.2. DHCP

"Dynamic Host Configuration Protocol"(en español "protocolo de configuración dinámica de host") es un protocolo de red que permite a los clientes de una red IP obtener sus parámetros de configuración automáticamente. La librería de Arduino contiene un ejemplo Dhcp.ino [30] que le permite al usuario obtener información acerca de cuál es la IP asignada para Arduino y el modulo, puede conocer su dirección estática y dinámica.

2.3. PHP

"HyperText Pre-processor"Fue creado originalmente por Rasmus Lerdorf en 1995 [29]. Es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que

procese los datos. PHP se considera uno de los lenguajes más flexibles, potentes y de alto rendimiento conocidos hasta el día de hoy Puede ser usado en la mayoría de los servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún costo.

2.4. HTML

"HyperText Markup Language" Es el lenguaje que se emplea para el desarrollo de páginas de internet, está compuesto por una seríe de etiquetas que el navegador interpreta y da forma en la pantalla. HTML dispone de etiquetas para imágenes, hipervínculos que nos permiten dirigirnos a otras páginas, saltos de línea, listas, tablas, etc. Al ser un estándar, HTML busca ser un lenguaje que permita que cualquier página web escrita en una determinada versión, pueda ser interpretada de la misma forma (estándar) por cualquier navegador web actualizado [8].

2.5. Protocolo I2C

"Inter-Integrated Circuit" en español Inter-Circuitos Integrados, es un bus de comunicaciones en serie con la gran ventaja de utilizar solo dos lineas para transmitir la información y una para la referencia (GND). Las dos conexiones para el bus I2C se llaman reloj (SCL, Serial Clock) y línea de datos (SDA, Serial Data) [31].

En el Arduino uno los pines para utilizar este protocolo de comunicación se ubican despues del pin AREF.

En la figura 2.1 se muestra el protocolo I2C con Arduino uno que funciona cuando

el dispositivo maestro envía una secuencia de inicio, lo que hace que los esclavos estén alertas activándose en modo de espera de una transacción. El maestro especifica con cual desea compartir información envíando la dirección de dispositivo. El dispositivo esclavo que posee esa dirección continuará con la transacción,mientras que los otros esclavos esperaran la próxima secuencia de inicio.

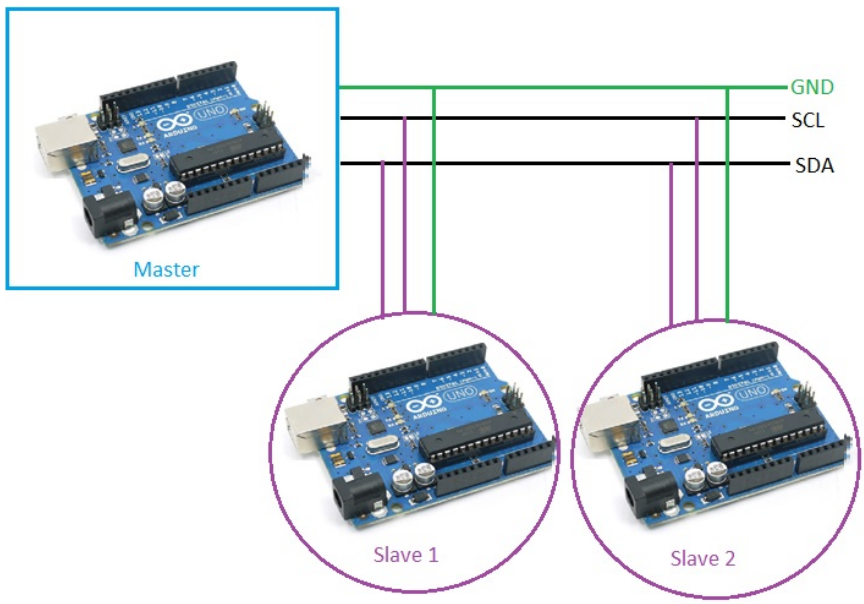

Figura 2.1: Protocolo de comunicación I2C con Arduino

2.6. Criptografía

Desde el principio de los tiempos el hombre ha tenido la necesidad de comunicarse de manera secreta, ya sea por motivos de guerra o porque simplemente un mensaje debía estar oculto para ciertas personas. La palabra criptografia significa: Escritura oculta, según el griego “criptos” y “grafé”, y desde A.C se desarrollaron diferentes formas de enviar los mensajes ocultos.

Aunque el objetivo original de la criptografía era mantener en secreto un mensaje, en la actualidad no se persigue únicamente la privacidad o confidencialidad de los datos, sino que se busca además garantizar la autenticación de los mismos (el emisor del mensaje es quien dice ser, y no otro), su integridad (el mensaje que leemos es el mismo que nos enviaron) y su no repudio (el emisor no puede negar el haber enviado el mensaje) [18]

La Criptografía se divide en dos grandes ramas:

- Criptografía simétrica: También conocida como de llave secreta es aquella que utiliza algún método matemático llamado sistema de cifrado para cifrar y descifrar un mensaje utilizando únicamente una llave secreta [41]. Ejemplos de esta criptografia son: El Data Encryption Standard (DES) y Advanced Encryption Standard (AES).

- Criptografía asimétrica: También conocida como de de llave pública, utiliza una llave publica y una llave privada, esta ultima solo es conocida por una persona. Ejemplos de esta criptografia son: El metodo de Rivest, Shamir y Adleman (RSA), El Gamal y Curvas Elípticas.

2.7. SOAP

"Simple Object Access Protocol" es un protocolo para el intercambio de mensajes entre dos objetos diferentes pueden comunicarse usando XML (eXtensible Markup Language) [13], este ultimo es un lenguaje de marcas que es utilizado para almacenar

datos en forma legible.

2.8. HTTP Request

"Hypertext Transfer Protocol" es un protocolo utilizado para transferir archivos en formato HTML entre un navegador y un servidor web que se localiza mediante una cadena de caracteres denominada dirección URL.

Este protocolo se basa en un modelo de solicitud/respuesta de modo que el navegador abre una conexión a un servidor y realiza una solicitud. El servidor procesa la solicitud del cliente y devuelve una respuesta [35].

Capítulo 3.
Estado del arte

A medida que el concepto de internet de las cosas se ha integrado en la sociedad, muchas personas en el mundo han optado por diseñar sistemas que les permitan controlar su hogar desde internet ya sea remotamente con computadores y/o aplicaciones móviles. A continuación se presentan diversos proyectos como punto de partida de este trabajo de grado.

Hribernik en su articulo [23] plantea la en la co-creación que constituye en el desarrollo a partir de objetos inteligentes y el internet de las cosas, con la plataforma Arduino uno contribuyen a los que ellos llaman "co-creación"de productos inteligentes, para esto trabajaron con RFID y sensores de luz, humedad y servo motores. Para transmitir los datos de dichos sensores usaron la tecnología de transmisión inalámbrica del modulo XBee [14] que mediante el protocolo de comunicación Zigbee [20] permite operar el Arduino uno quien recibe los datos de los sensores y los envía a un coordinador que está conectado al servidor (Interfaz web PHP) de hogar que los almacena en una base de datos MySQL. También desarrollaron una solución de bajo costo para un montacargas inteligente con lectores RFID, el monitoreo de este montacargas se hace en línea usando las plataformas Pachube y Twitter con los

cuales permiten compartir datos de los sensores en tiempo real.

David Mendez de la Universidad Tecnológica de Mixteca en su trabajo de grado [7], diseña e implementa un sistema de comunicaciones que permita la interacción entre un sistema micro-controlado y una interfaz de usuario basada en la web para el monitoreo remoto dentro de la universidad tecnológica de Mixteca para controlar dispositivos basados en MCU de propósito general que se comunica con la computadora mediante un RS232-Ethernet, este sistema uso el protocolo Ethernet y el protocolo TCP/IP. Desde cualquier lugar dentro de la universidad conectado a dichos protocolo se podrá mandar y recibir información, la cual llega a un gestor de nodos quien traduce las ordenes generadas y establece la comunicación TCP/IP entre estos.

Martha Carvajal Aguilar en su trabajo de grado [33], presentan un sistema de control familiar para la automatización en áreas especificas de una casa, este sistema esta denominado "SECOFA" que le permita a usuario activar/desactivar sensores u actuadores como de humo, movimiento, luz, electroválvulas, entre otros. También le permite al usuario encender/apagar las luces de su vivienda. Todo este sistema se desarrollo con un micro-controlador PIC18F4520 y una pantalla táctil GLCD, se uso el lenguaje de programación Mikrobasic y para la realización de gráficos en el GLCD se uso Fastled. Al ingresar al hogar se encontrara con una cerradura eléctrica a la cual se le debe ingresar una contraseña de 5 dígitos, dicha clave permitirá que el micro-controlador envié la señal para que se abra la puerta. La contraseña se almacena gracias a la programación de la memoria EEPROM. El siguiente paso es que en la

pantalla se muestran las pasividades: Manual, automático y seguridad, en donde se activaran:

- Manual: Activar/desactivar sensores Análogos.
- Automático: Activar/desactivar sensores Digitales
- Seguridad: Activar/desactivar sensores de presencia y de humo.

Aunque los sensores de temperatura, luz, presencia y humo funcionan simultáneamente nunca interactuarán los cuatros juntos.

Alexis Roque Capel en su proyecto [9] tenia como objetivo mejorar el nivel de vida de personas con cierta discapacidad a partir del sistema domótico X10 debido a que es de bajo costo, programación sencilla, fácil manejo, se integra a la vivienda, contribuye al ahorro de energía, etc. Este sistema a parte de poseer la ejecución de actividades como encendido/apagado de luces, sensores de humo, presencia, humedad, electroválvulas, detector de incendio, entre otros. También cuenta con un sistema de detección de emergencias sanitarias o accidentes domésticos imprevistos que se activa solo con un pulsador cuya señal será enviada a una central donde se realiza el monitoreo de los hogares de este tipo, esta entidad sabrá la dirección y lugar en la casa donde se está realizando la llamada de emergencia. El sistema también cuenta con sensores de presencia que capta movimiento a temperaturas cercanas a los 36° que encenderán as luces dentro del hogar y serán alarmas de detección de intrusos. El proyecto completamente implementado y el sistema funcionando se presento con

un costo total de 10927*euros* sin contar el pago de la mano de obra.

Por otra parte Mario Rodríguez Cerezo en su trabajo [10] tenia el objetivo de diseñar, construir y probar un sistema domótico para el control de una red de sensores y actuadores con el fin de recolectar datos como temperatura, humedad y luminosidad y activar/desactivar elementos como la calefacción, riego del jardín, persianas y luces. Usaron la tecnología ZigBee [20] ya que no necesitaron grandes anchos de banda y se garantizaron bajos consumos, la unidad central está conformada por un módulo de radiofrecuencia XBee [14] y un dispositivo de control como unidad central: BeagleBone [12] para permitir que el usuario interactué con el sistema a través de computadores o tabletas. La plataforma BeagleBone funciona como maestro y los módulos XBee como esclavos quienes se comunican a través de radiofrecuencia, el esclavo envía los datos solicitados por el maestro y este último se encarga de enviarlos al servidor web, donde el usuario podrá visualizarlo a través de la dirección IP fija proporcionada por la BeagleBone.

Mario Sáenz Espinoza presento su trabajo [17], que consiste en el diseño e implementación de un sistema de bajo costo para el control de viviendas, los módulos seleccionados de la vivienda fueron: el modulo de detección de fuga de gas, control de puerta y control de luces. Utilizaron el micro-controlador ATmega16 [6] debido a sus especificaciones eléctricas y físicas y posee un convertidor análogo-digital. El sistema cuenta con una aplicación web que le avisara a usuario cuando tenga visita y desee ingresar a su vivienda para que abra la puerta del parqueadero, el modulo

de luces se activara cuando se entre o salga de las habitaciones, por medio de sensores ultrasónicos se determinara la cantidad de personas dentro de esta, el sistema es suficientemente autónomo para determinar as horas del encendido de iluminación. Después de realizar pruebas también se probo reemplazando los sensores por otros micro-controladores ATmega16, en el caso de sensor de gas puede ser un buen sustituto PIC16F873 [42]. Para la comunicación entre el sistema y el usuario se uso el programa Cutecom.

Daniel Oña Rodríguez presento un proyecto [48] que se baso en el diseño e implementación de un sistema de sensado inteligente como el fin de integrar todos los sistemas mecánicos y electrónicos para determinar su proceso de automatización en el edificio shalom en Quito-Ecuador. Se basaron el protocolo de comunicación ZigBee para controlar los dispositivos implementados en dicho sistema, cabe resaltar:

- Puerta de acceso vehicular estacionamiento.
- Cerradura de acceso al edificio, al departamento y oficinas, Para este sistema de seguridad usaron el HPA-2601.
- Para controlar dispositivos ON/OFF usaron los equipos de control HPA- 2130 Y HPA-2530.
- Se estimo un ahorro energético alrededor del 24 % cambiando las bombillas del edificio a bombillas con control de intensidad (DIMMER) HPA-2140.

El control de sistema se da a partir de una plataforma web en la cual los usuarios

pueden alertar sobre emergencias medicas, para el acceso a la plataforma el usuario debe contar con la IP determinada y con un usuario y contraseña.

Vale la pena mencionar los sistemas automatizados para el hogar como el desarrollado por Miguel A. Zamora en [38] donde se presenta DOMOSEC este sistema de automatización para el hogar, utiliza tecnologías existentes. Proponen un sistema de comunicación basado en IP a través de UDP, entre un controlador principal que es el módulo de automatización del hogar (HAM) y el restos de equipos los cuales utilizan tecnologías domoticas que permite su conexión a proveedores de seguridad y los dueños de las casas, la pueden visualizar a través de internet. El HAM centraliza la casa de "Inteligencia", porque contiene la configuración utiliza para controlar todos los instalados dispositivos e incluye una interfaz hombre-máquina (HMI).

Con respecto a la seguridad, utilizan distintos tipos de alarma. Los eventos de alarma Se notifica simultáneamente a través de todos disponible proxies de alarma, y para "mantenerse aliviado" se emplea para recibir mensajes periódicos del HAM. Este mecanismo evita que un atacante bloquee el canal de seguridad sin ser detectado.

El proyecto presentado en este trabajo a diferencia de los referenciados se basa en la plataforma open source Arduino uno, se probo con 3 diferentes módulos de Ethernet para este micro-controlador y se comprobó su fácil acceso a la red ademas de ser capaz de soportar el lenguaje de programación web: **PHP**. Lo que permitió el desarrollo de una aplicación web, se creó un login con ayuda de una base de datos en **MySQL** [21]. Para que personas puedan acceder a la página se les otorgara un usuario

y contraseña, permitiendo tener un control de los usuarios que manipulen los objetos entrelazados con la web. Por medio de esta aplicación se controlara remotamente dispositivos como luces, sensores de movimiento, entre otros, se uso el protocolo de comunicación I2C con la ventaja de poder expandir el proyecto a futuro.

Con respecto a la seguridad y a los estándares mencionados en este trabajo se presentan algunos trabajos que usaron diferentes tipos de criptografia para proteger sus aplicaciones. La razón principal es que actualmente con la llegada de Internet, fue indispensable el uso de herramientas automatizadas para la protección de archivos y otro tipo de información almacenada en la computadora, algunas de estas herramientas son los cortafuegos, los sistemas detectores de intrusos y el uso de sistemas criptográficos [41].

A continuación se describen los trabajos que sirvieron como punto de partida para desarrollar el modelo de seguridad para nuestra aplicacion.

Jan Henrik Ziegeldorf presento el articulo [26] donde se resalta las amenazas y desafíos que se tienen en el concepto de internet de las cosas, las cosas inteligentes permiten una recolección de datos sin embargo son muchas las amenazas a la privacidad, por esto se debe garantizar:

- Conciencia de los riesgos de privacidad impuestos por las cosas inteligentes y servicios que rodean el tema de los datos.
- Control individual sobre la recolección y tratamiento de información personal por las cosas inteligentes.

- Conciencia y control del uso y divulgación de información personal por aquellas entidades a ninguna entidad fuera del circulo de control.

Martin Henze en su trabajo [34] presento un enfoque para una aplicacion de privacidad para los servicios de IoT basados en la nube cuyos requisitos acordados son: Notificación, consentimiento, autodeterminación, seguridad adecuada y el uso internacional. La aplicacion lleva el nombre de UPECSI. los puntos de aplicacion de privacidad (Privacy Enforcement Points-PEP) se encargan de cifrar los datos con una clave de protección simétrica antes de subirlo a la nube por consiguiente en el servicio dela nube el usuario puede descifrar los datos a los que se encuentre autorizado. En esta aplicacion se introduce el desarrollo del lenguaje de privacidad (PDL) para que los desarrolladores del servicio de Cloud puedan describir sus consideraciones de privacidad.

Se resalta el trabajo de René Hummen, quien presenta un mecanismo detallado de fragmentación 6LoWPAN [45], este mecanismo es una capa de adaptación del IPV6 con el fin de viabilizar la conectividad IP en dispositivos de recursos limitados. En este trabajo describen un ataque en el cual el atacante primero consigue acceso a la red luego participan en la estructura de enrutamiento y desde ahí pueden enviar mensajes a cualquier nodo en la red, dependiendo en donde esté situado el atacante podría retrasar reordenar o dejar caer paquetes legítimos. Para contrarrestar los ataques proponen dos mecanismos de defensa que son: “El esquema de encadenamiento de contenido” y “El enfoque de buffer Split”

Ahmed Lounis propone en su trabajo [4] un sistema seguro que garantice la confidencialidad, integridad, así como el control de acceso de datos, basándose en los desafíos que enfrentan los sensores en aplicaciones medicas. El sistema combina varios esquemas criptográficos como lo es el uso de criptografía simétrica y ABE para cifrar los datos, generando la clave simétrica (RSK) y cifrar la RSK con CP-ABE, El archivo cifrado y el cifrado RSK se envían a la nube para almacenamiento de información que sera compartida con los usuarios autorizados. De hecho, si un usuario tiene una clave secreta cumple con la Directiva de acceso CP-ABE, que será capaz de descifrar la RSK y por lo tanto, para descifrar el archivo.

Mihai Christodorescu en su articulo [11] sostiene que la seguridad de los datos debe ir mas allá de contratos legales o sociales y sobre todo si se quieren manejar datos que se envían a servidores web, por esta razón en este trabajo se propone una arquitectura que garantice la seguridad de los datos de los usuarios dentro de aplicaciones web, la cual funciona cifrando los datos desde el momento que son enviados y son descifrados cuando llegan a la maquina del usuario permitiendo que todos los datos que se queden en el equipo local estén encriptados. Se propone generar una clave única para cada usuario con el fin de autenticar la aplicacion web, por ello la generación de la clave de encriptacion se hace a partir de la ecuación:

$$K = PRFpassword(dom) \tag{3.1}$$

Esto se hace para el nombre de dominio “dom” de una aplicación web determinada. Además se cambia la contraseña de la aplicación web para que el nombre de dominio

sea cifrado a partir de la ecuación:

$$Web - apppassword = Ek(dom) \tag{3.2}$$

Si el usuario tiene que cambiar su contraseña, los datos almacenados en el proveedor tienen que volverse a cifrar bajo la nueva clave. El único caso en el que aparentemente esto no es posible, es cuando los datos privados son de sólo lectura después de ser subidos a la aplicacion.

Capítulo 4.
Descripción del sistema

El sistema se basa en el co-diseño hardware/software, que permite controlar y recibir información de sensores-actuadores conectados a una red de bajo costo desde cualquier parte del mundo, en la figura 4.1 se observa como esta conformado este sistema.

Este sistema se diseño con el fin de cumplir los siguientes requerimientos estipulados en los estándares de seguridad:

Confidencialidad: Para mantener la confidencialidad de las comunicaciones internas y externas, que deben estar encriptados.

Integridad: Con el fin de detectar la corrupción de datos, técnicas de encriptación serán utilizadas durante las transmisiones para validar la integridad de la información.

Disponibilidad: Para garantizar que el servidor es siempre hacia arriba, dependemos de empresas externas que prestan servicios en la nube. Su infraestructura (firewall, proxy, cachés, despido, etc.) nos ayudará a mitigar los ataques de Denegación de Servicio-y temas relacionados.

Autenticación: Debemos poner en práctica un sistema de autenticación que permite establecer una identificación y una contraseña para cada usuario y que conectado

a nuestro sistema, así como la base de datos.

Control de acceso: En conjunción con la autenticación, tenemos que configurar diferentes niveles de capacidades. Por ejemplo, los administradores pueden añadir y eliminar usuarios. Los usuarios pueden añadir y eliminar sus propias cosas. Las cosas sólo pueden enviar datos a sus propietarios.

No repudio: Los datos serán almacenados encriptados utilizando una clave asimétrica. Esto puede no evita la modificación, pero podemos estar seguros de que los datos pertenecen a nosotros.

A continuación se describe el hardware y el software utilizado en este trabajo.

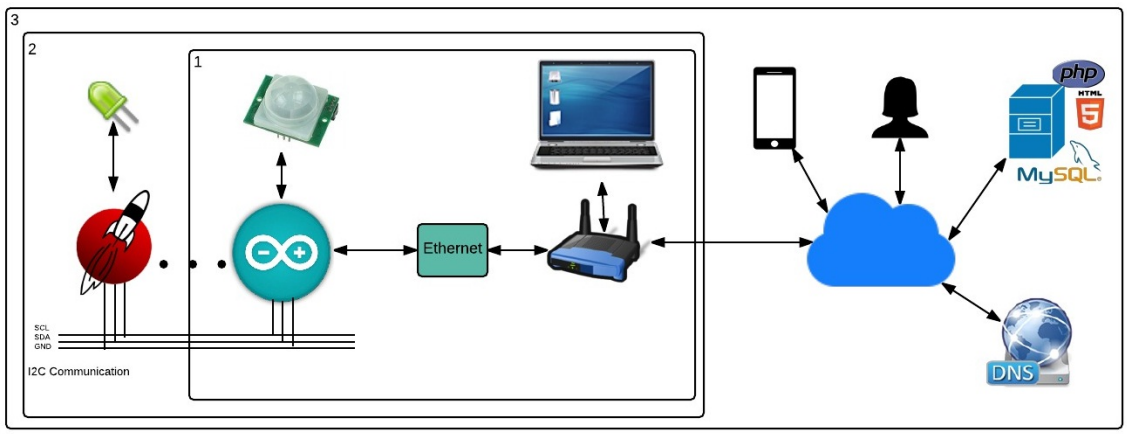

Figura 4.1: Diagrama del sistema

4.1. Hardware

Para la implementación del sistema fueron necesarios los dispositivos que se detallaran a continuación.

4.1.1. Arduino uno

Es una plataforma de hardware libre, basada en una placa electrónica basada con un micro-controlador ATmega328 y un entorno de desarrollo, diseñada para facilitar el uso de la electrónica en proyectos multidisciplinares [2]. Arduino puede tomar información del entorno a través de sus entradas analógicas y digitales, puede controlar luces, motores y otros actuadores. El micro-controlador en la placa Arduino se programa mediante el lenguaje de programación Arduino (basado en Wiring) y el entorno de desarrollo Arduino (basado en Processing). Cuenta con 14 pines digitales de entrada/salida (de los cuales 6 se pueden utilizar como salidas PWM), 6 entradas analógicas, un 16 MHz resonador cerámico, una conexión USB, un conector de alimentación, un header ICSP (In Circuit Serial Programming), y un botón de reinicio. Contiene todo lo necesario para apoyar el micro-controlador; simplemente conectarlo a un ordenador con un cable USB o el poder con un adaptador de CA o la batería a CC para empezar. Ver figura 4.2

Figura 4.2: Arduino uno R3

4.1.2. Intel Galileo

Galileo es una placa electrónica basada en el procesador Intel Quark SoC procesador X1000 aplicación, un sistema de clase Pentium Intel de 32 bits en un chip (ficha técnica) [44]. Es la primera de mesa basado en la arquitectura Intel diseñado para ser compatible con pin de hardware y software con escudos diseñados para el Arduino Uno R3. Pines digitales 0 a 13 (y la AREF adyacente y pines GND), entradas analógicas de 0 a 5, el encabezado de poder, header ICSP, y los pines del puerto UART (0 y 1), están todos en los mismos lugares que en la Arduino Uno R3. Esto también se conoce como el 1,0 pinout Arduino. La ventaja de esta tarjeta es que tiene incluido la shield de Ethernet permitiendo que con la conexión de un cable RJ45 nos

conectemos a internet.

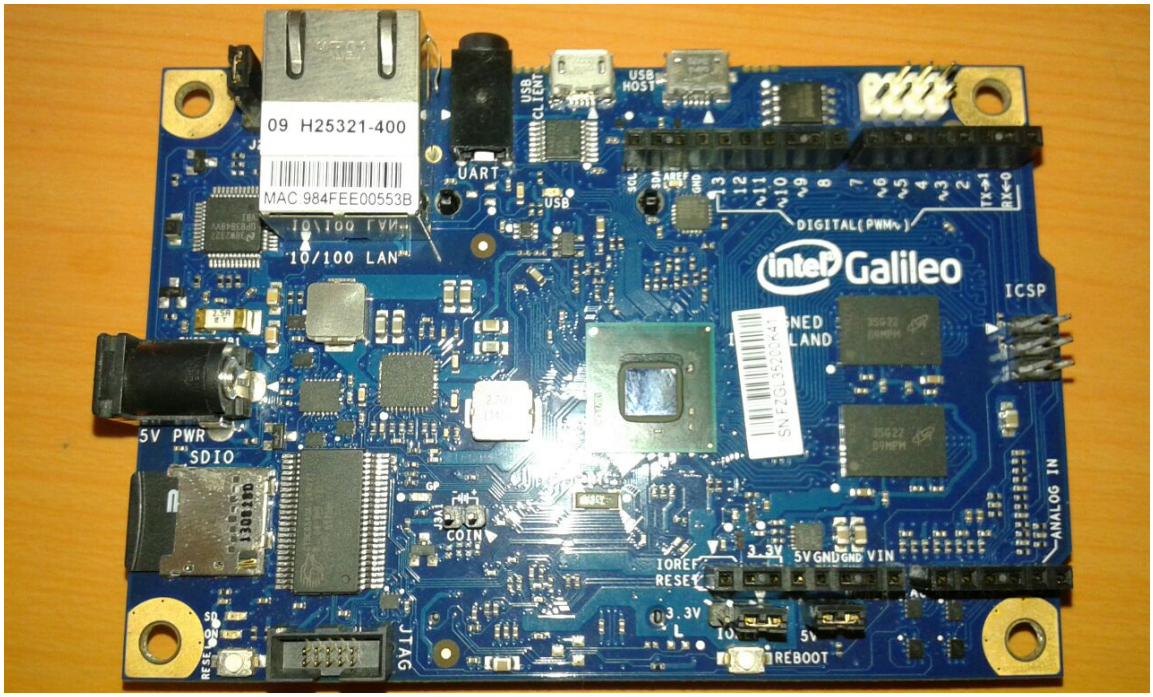

Figura 4.3: Intel Arduino Galileo

4.1.3. Tiva C Launchpad Serie TM4C123G

Es una placa micro-controlada de bajo costo fabricada por Texas Instruments, cuenta con botones programables por el usuario, 40 pines que permiten agregar soporte para displays, interfaces inalámbricas, sensores, un LED RGB para aplicaciones personalizadas también ofrecen compatibilidad básica con plataformas de lanzamiento MSP430 y C2000 existentes y otras capacidades a sus proyectos con mucha facilidad [25].

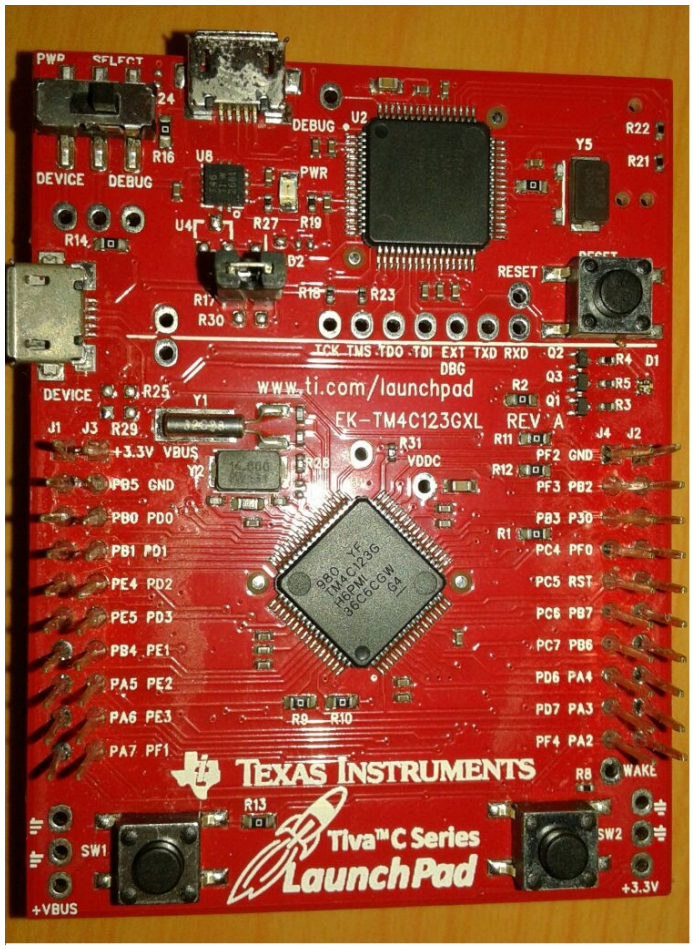

Figura 4.4: Tiva C Series TM4C123G LaunchPad Evaluation Board

4.1.4. Modulo ENC28J60

Este módulo trabaja con un protocolo serial síncrono que se utiliza para comunicar un micro-controlador con otro y con periféricos a distancias cortas, más conocido como SPI. Para hacer una conexión SPI siempre habrá un dispositivo maestro (usualmente un micro-controlador) que controlará uno o varios periféricos (esclavos), sin embargo en este trabajo utilizamos el protocolo de Los pines de suma importancia en este módulo son:

Arduino	**ENC28J60**
8	CS
11	SI
12	SO
13	SCK

Cuadro 4.1: Conexión ENC28J60 y Arduino

- Cs o select: Se usa por el máster para habilitar o deshabilitar un determinado periférico.
- SCK: pulsos de reloj para sincronizar la comunicación.
- SI (Master out/ Slave in): Datos del maestro al esclavo.
- SO (Master in/ Slave out): Datos del esclavo al maestro.

La conexión entre Arduino y este módulo se puede apreciar en la tabla 4.1

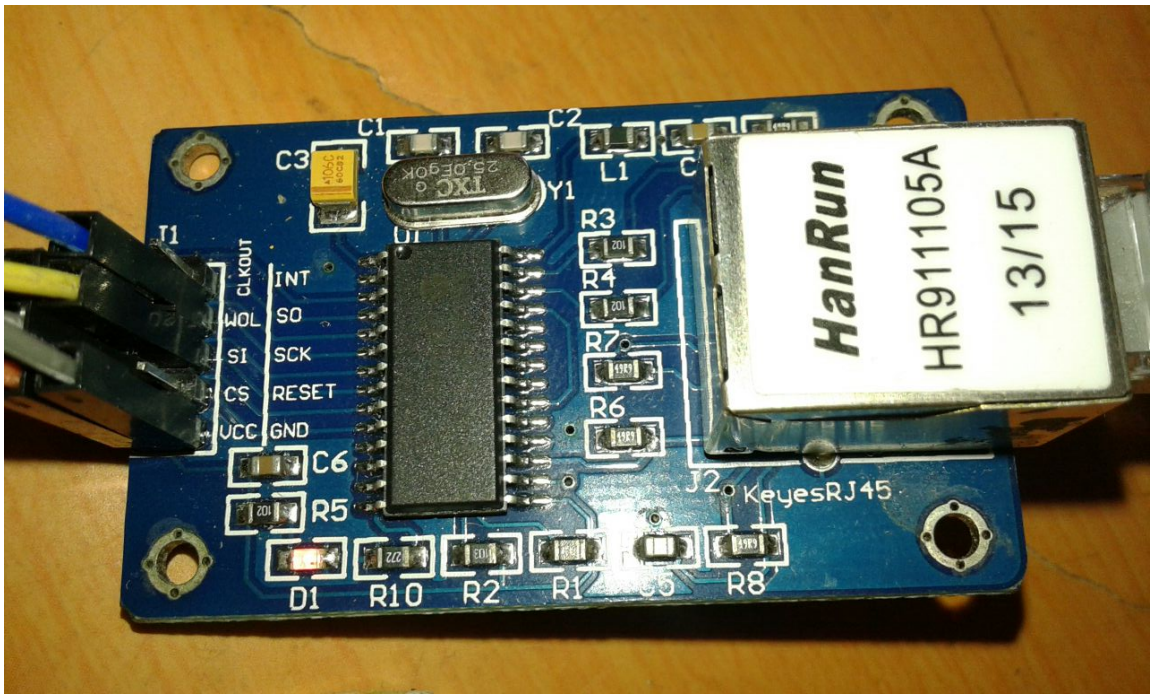

Figura 4.5: Modulo Ethernet Lan ENC28J60

4.1.5. Ethernet shield

La Shield Ethernet Arduino conecta tu Arduino a Internet en cuestión de minutos. Se tiene que conectar este módulo en su placa Arduino, conectarlo a su red con un RJ45 cable y seguir algunas instrucciones sencillas para empezar a controlar su mundo a través de internet.

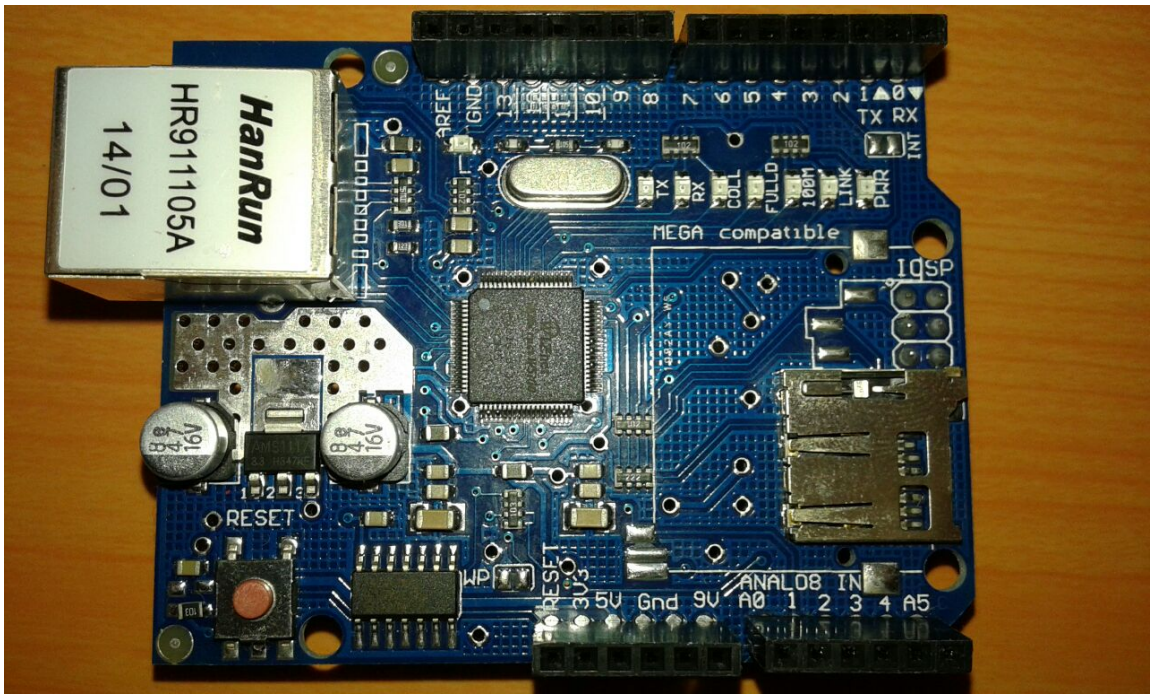

Figura 4.6: Ethernet shield Arduino

4.1.6. PIR-Sensor de movimiento

"Pasive Infra Red", es un dispositivo piroeléctrico detector y medidor de calor, este sensor detecta movimiento mediante un promedio de calor irradiado en el tiempo. Es muy utilizado en aplicaciones electrónicas, robóticas e iluminación debido a su bajo costo y fácil programación y solo necesita 5V de alimentación.

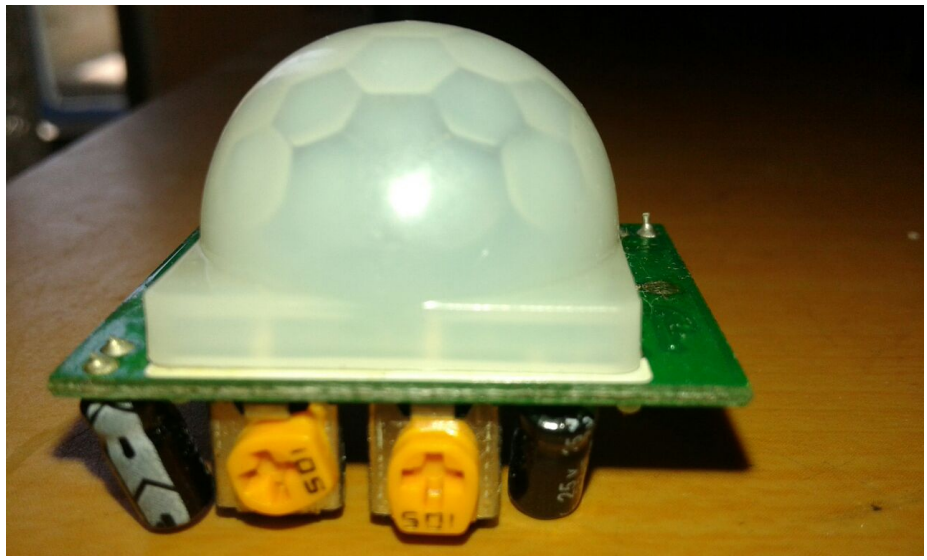

Figura 4.7: PIR sensor de movimiento

4.1.7. LM35-Sensor de Temperatura

LM35 es un sensor de temperatura que contiene circuitos integrados de precisión, cuya tensión de salida es linealmente proporcional a la temperatura Celsius (centígrados). Dentro de las ventajas de este sensor esta que no requiere calibración externa, puede usar con fuentes de alimentación individuales, o con más y menos suministros, ya que llama solamente consume 60mA desde su suministro y tiene muy escaso calentamiento [24].

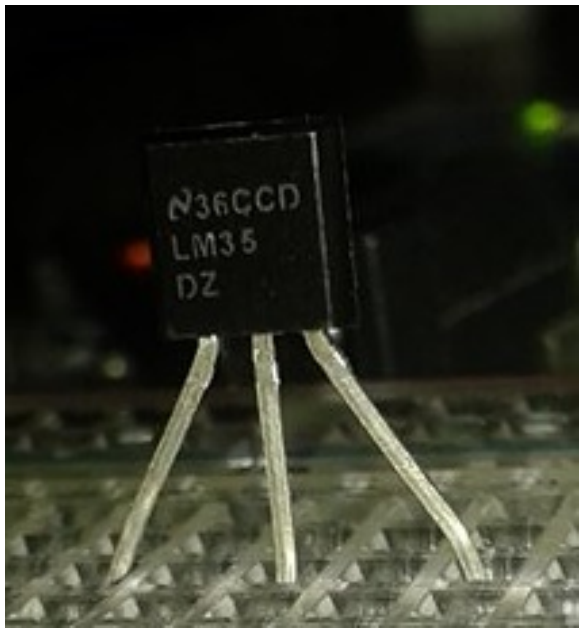

Figura 4.8: LM35 Sensor de Temperatura

4.1.8. Sensor de Corriente

El Sensor No-Invasivo ECS1030-L72 es capaz de medir una carga de hasta 30 Amperios. El cual puede ser usado para la construcción de un dispositivo de protección contra sobrecargas de corriente.

Figura 4.9: Sensor de corriente ECS1030-L72

Teniendo en cuenta las características del sensor ECS1030-L72 [22], se hace necesario colocar una resistencia de carga a la salida para convertir la corriente inducida de la bobina a un voltaje, este sensor necesita un pequeño circuito para que los niveles de tensión generados puedan ser interpretados seguramente por el converso análogo-digital del micro-controlador [49]. En la Figura 4.10 se observa el circuito usado.

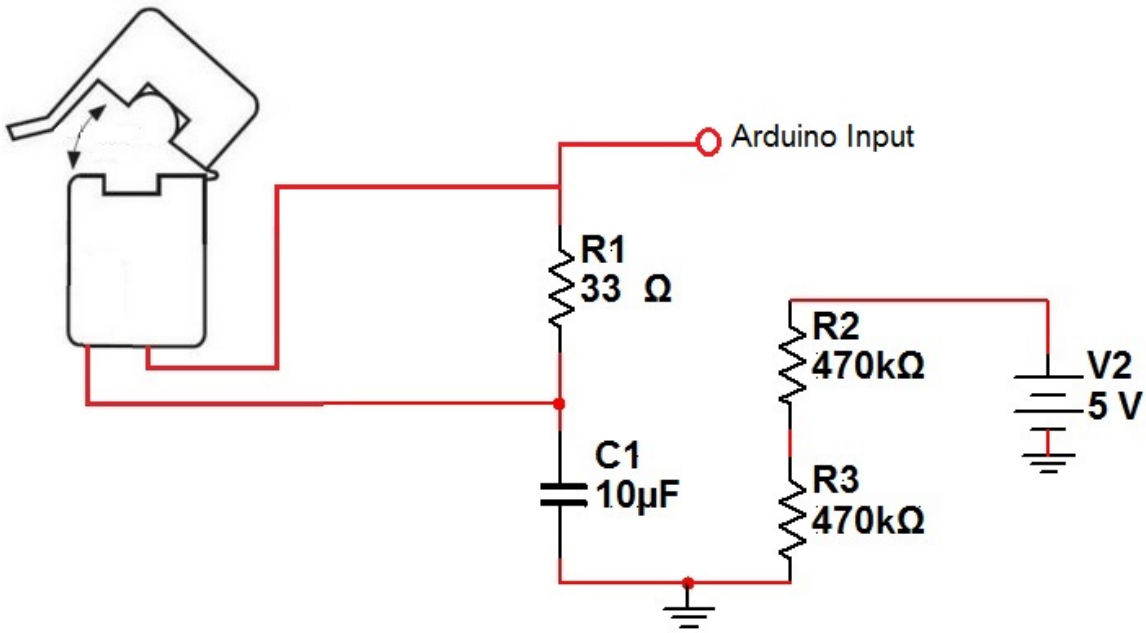

Figura 4.10: Circuito para el sensor de corriente (Tomada del articulo de Sabalza [49])

Para la medición de voltaje se utiliza un transformador AC/AC de 110V en el primario y 12V en el secundario, este por si solo no nos permite adaptar el voltaje de la toma en el rango de 0 y 5V. Para adecuar el valor de voltaje a este rango, se implemento un circuito divisor de tensión, la especificación de los elementos pasivos se realizo en un trabajo anterior [49]. En la Figura 4.11 se observa el circuito usado.

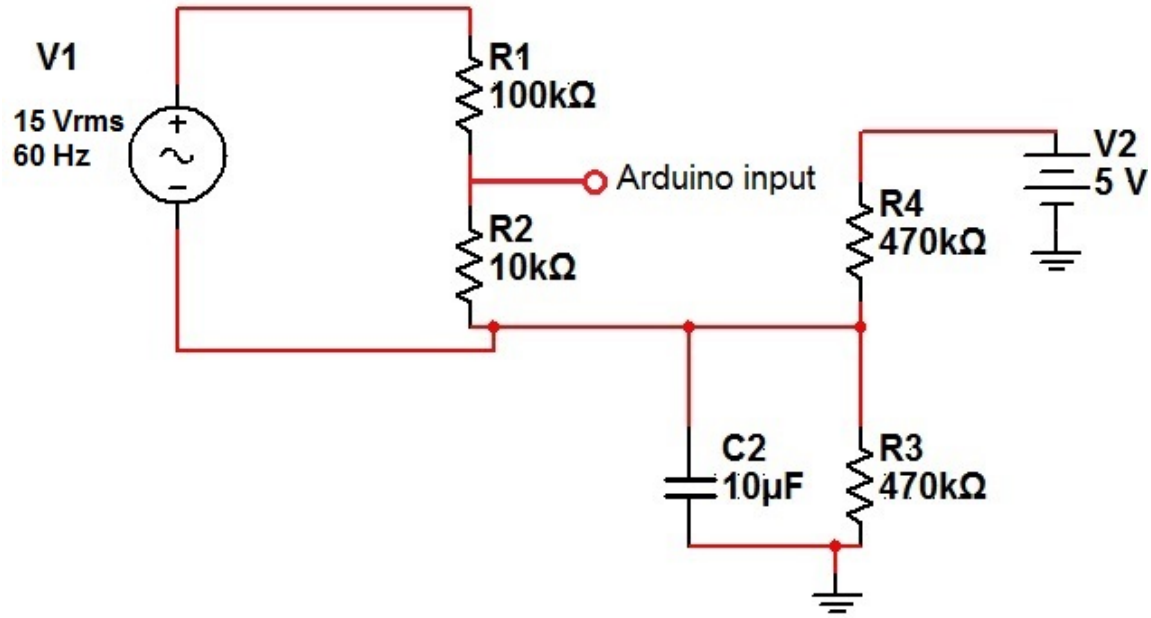

Figura 4.11: Circuito divisor de tensión (Tomada del articulo de Sabalza [49])

4.1.9. Keypad

Teclado numérico con una matriz de 4x4 ideal para el desarrollo de proyectos con micro-controladores, características principales: Diseño ultra-delgado y respaldo adhesivo proporciona una fácil integración a cualquier proyecto, excelente relación calidad-precio, Fácil comunicación con cualquier micro-controlador. Ideas de aplicación: sistemas de seguridad, selección de menú, calculadoras, etc.

Figura 4.12: Keypad

4.1.10. Router Inalambrico

El 300Mbps Wireless N Router TL-WR841N es un dispositivo combinado con cable/inalámbrico de conexión de red diseñado específicamente para las necesidades de pequeñas empresas y de redes de oficina en casa. Contiene un botón WPS en el exterior elegante y de moda asegura WPA2, la prevención de la red de intrusiones externas. Cumpliendo con el estándar IEEE 802.11n, el TL-WR841N puede establecer una red inalámbrica y obtener hasta 15 veces la velocidad y 5 veces el alcance de los

productos convencionales 11g. También, con velocidades de transmisión de hasta 300 Mbps [53].

Figura 4.13: Router TP-LINK

4.2. Software

A continuación, se describirán las distintas herramientas de software con las cuales se realizo la programación de los micro-controladores para luego describir el trabajo realizado.

4.2.1. Arduino

El código abierto Arduino Software (IDE) hace que sea fácil de escribir código y subirlo a la junta. Se ejecuta en Windows, Mac OS X y Linux. El entorno está escrito en Java y basadas en el procesamiento y otro software de código abierto. Este

software se puede utilizar con cualquier placa Arduino, para conocer las ventajas y todo lo que brinda esta plataforma se recomienda leer el Arduino CookBook [32]. Para este proyecto se uso la versión 1.0.5-r2 para Windows.

4.2.2. Energia

El software de programación Energia [3], esta es una plataforma de prototipos electrónicos de código abierto iniciado por Robert Wessels en enero de 2012 con el objetivo de llevar el cableado y el marco de Arduino para el LaunchPad basada Texas Instruments MSP430. La Energia IDE es multiplataforma y apoyado en Mac OS, Windows y Linux. Energia utiliza el compilador mspgcc por Peter Bigot y se basa en el cableado y el marco Arduino. Para este proyecto se uso la versión de Energia-0101E0012 para Windows.

Capítulo 5.
Metodología

En el desarrollo de esta investigación el primer sistema diseñado e implementado fue el de la figura 5.1 donde se muestra el diagrama básico nuestro primer prototipo implementado está conformado por Arduino, modulo Ethernet, un router, computador y sensores.

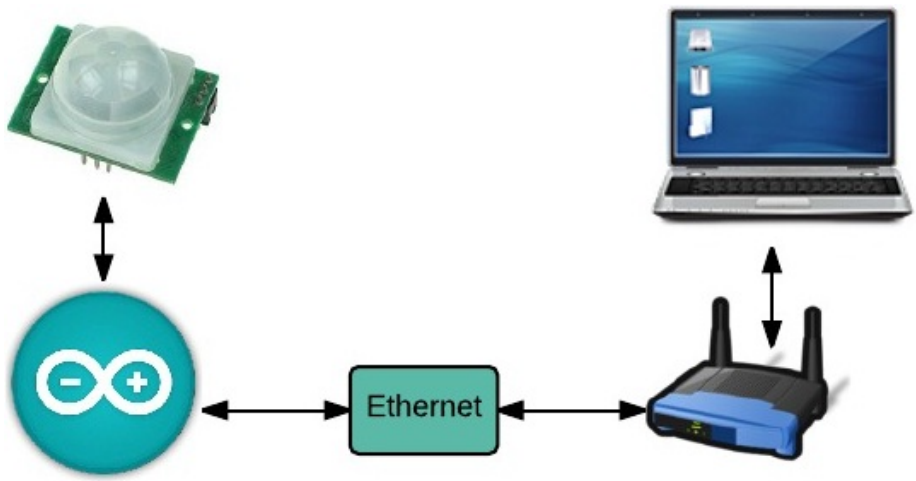

Figura 5.1: Diagrama Básico

La interconexión entre el modulo Ethernet y Arduino se da a través de las librerías diseñadas para cada modulo Ethernet de Arduino Cada una de estas trae un test de DHCP que le permite al usuario conocer información acerca de cuál es la IP asignada para Arduino (Ver Anexos 8). En la figura 5.2 se observa la salida en el monitor serial.

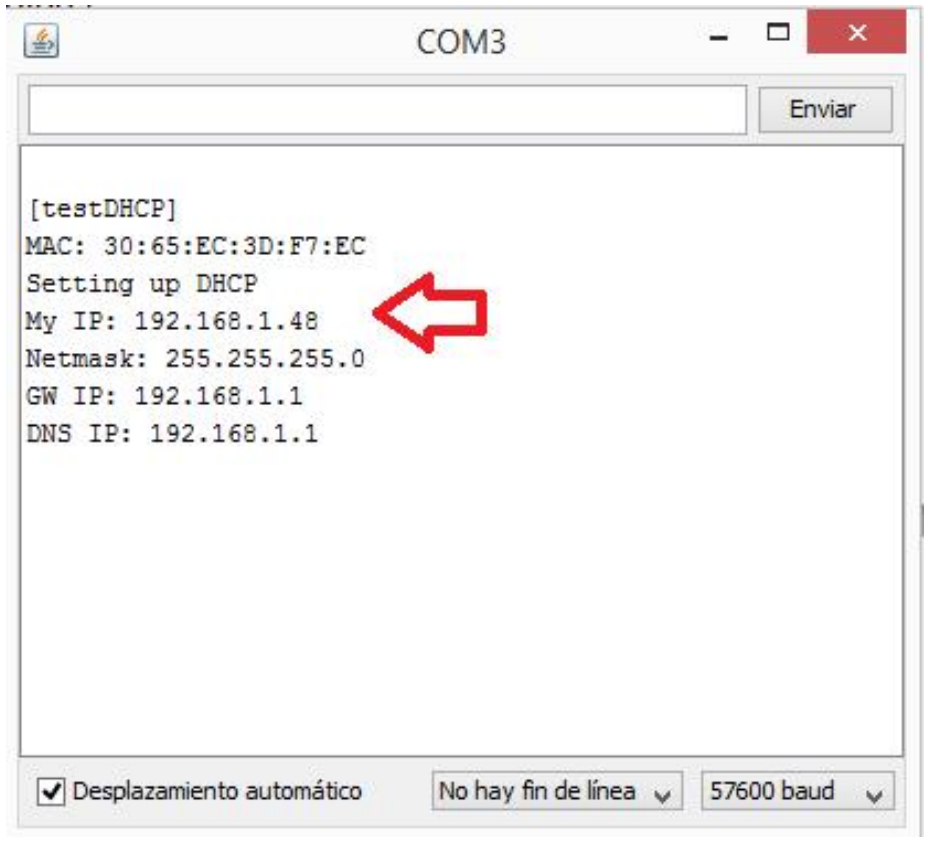

Figura 5.2: Monitor serial del Test DHCP

Cuando se conoció la IP asignada al Arduino se realizo el "Hola Mundo"desde dicha IP (Ver código en Anexos 8) y el resultado en la IP asignada se observa en la figura 5.3

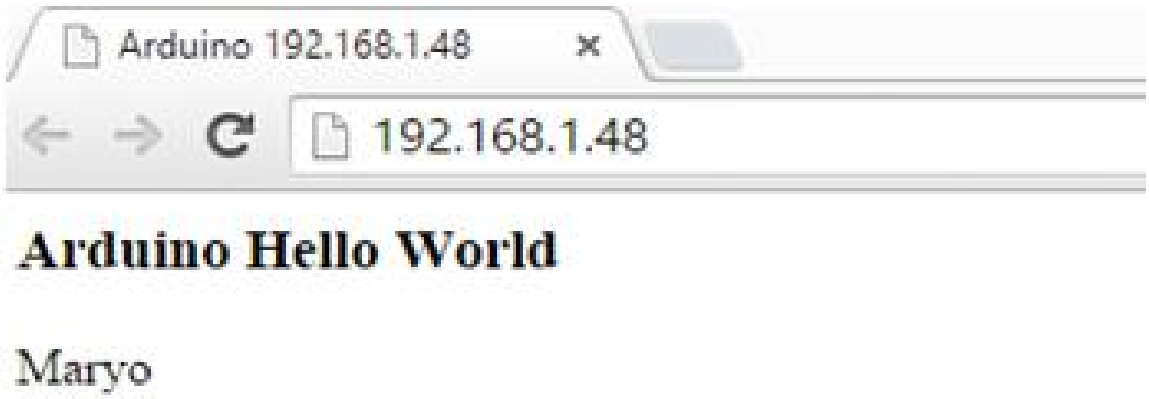

Figura 5.3: Hello Word Arduino con la IP asignada

Después de avance se procedió a realizar pruebas de control ON/OFF con un LED. Para enviar la orden de apagar/encender desde la IP se usa el comando "GET"que se encarga de leer un valor de un caracter de un archivo y lo de devuelve como valor de retorno, en este caso servirá para especificarle al Arduino que el estado del led ya sea ON/OFF (Codigo en Anexos 8). En la figura 5.4 se aprecia el resultado de la implementación de este código, para control ON/OFF de un led desde una IP asignada.

Figura 5.4: Controlando el estado de un Led desde la dirección IP

A continuación se probo conectado un sensor análogo en este caso el sensor de movimiento PIR, quien muestra en la IP si se detecta movimiento '1' y '0' de lo

contrario, también da la opción de encender un led si detecta movimiento o puede registrarlo sin encenderlo, ver figura 5.5. El código implementado en Anexos 8.

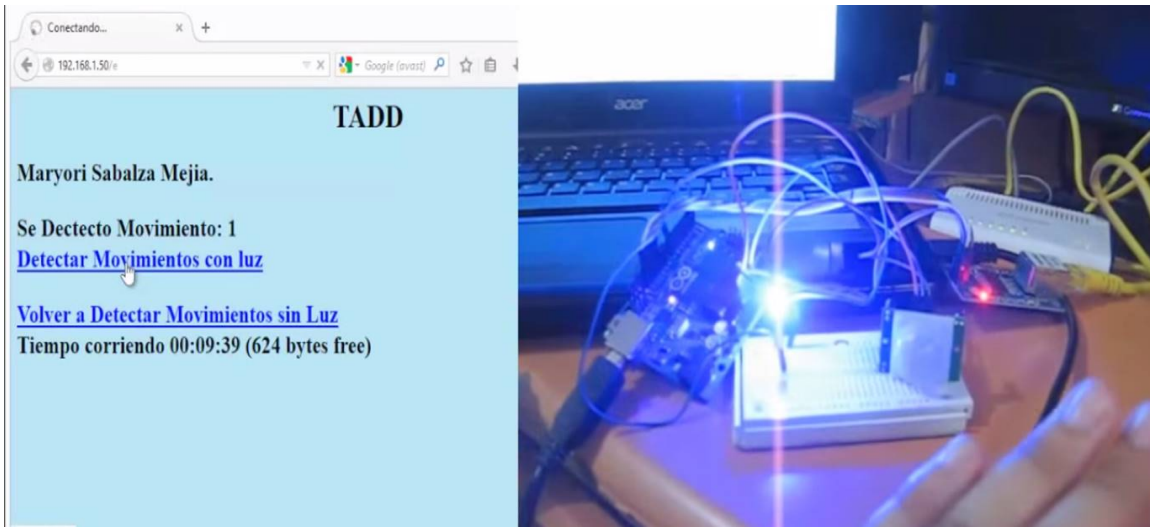

Figura 5.5: Control de un Sensor de movimiento desde la IP

5.1. Protocolo de comunicación I2C

Luego de comprobar el control desde una dirección IP, nos enfocamos en la comunicación ya que por términos de seguridad se opto que los sensores digitales manden información a un arduino esclavo quien se encargara de enviar esta información a un maestro quien se encarga de subir dicha información al servidor web para que sea visible por el usuario cuando lo solicite.

Para implementar la comunicación entre estas plataformas se dio usa a la librería que Wire.h [55] de Arduino la cual le permite comunicarse con I2C/TWI dispositivos. En las placas Arduino con el diseño R3 (1.0 pinout), la SDA (línea de datos) y SCL (línea de reloj) están en los conectores macho cerca del pin AREF. El Arduino

Debido tiene dos I2C/TWI interfaces de SDA1 y SCL1 están cerca del pin AREF y el adicional es en los pines 20 y 21.

La primera prueba para la comunicación fueron: Master Writer/Slave en donde el maestro le puede enviar ordenes o pedir información al esclavo y Receiver y Master Reader/Slave Sender donde el esclavo envía al maestro la información que le solicito. Para esto se conecto un Arduino uno como maestro [54] y la Tiva C Launchpad y otro Arduino uno como esclavos como se observa en la figura 5.6

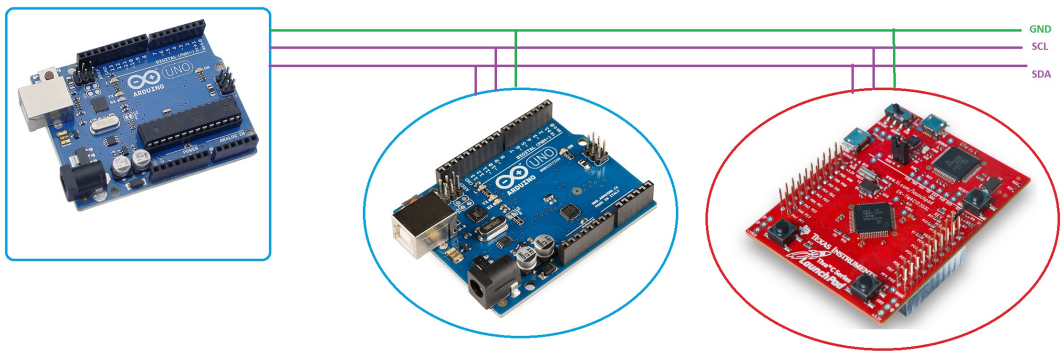

Figura 5.6: Maestro y esclavos

la siguiente prueba fue agregando a la comunicación establecida e implementada un teclado Keypad con el fin de que el usuario pudiese elegir por medio del Maestro de que dispositivo esclavo desea controlar y obtener información. Para lograr esto se implemento un código para el maestro donde se controla el teclado y establece una transmisión con los dispositivo, mientras que para los esclavos se conecto un led a cada uno y se le asignaron 2 teclas especificas (diferentes para cada esclavo) para apagarlo o encenderlo. Los códigos implementados en Anexos 8.

5.2. Seguridad del sistema

Para garantizar que en nuestro sistema seremos los únicos en controlar y monitorear los dispositivos, se opto por encriptar los datos recojidos por los sensores y solo podrán ser desencriptados por el maestro. Por otra parte es necesario aplicarle a la pagina web un protocolo de seguridad para que los espías no puedan acceder. A continuación se describe los métodos utilizados.

5.2.1. Protocolo SSL

Secure Sockets Layer (SSL), un protocolo de seguridad que proporciona privacidad a través de Internet. El protocolo permite que las aplicaciones cliente/servidor se comuniquen de una manera que no puede ser espiado. La ventaja de este protocolo es que puede negociar un algoritmo de cifrado y clave de sesión, así como autenticar a un servidor antes de que el protocolo de aplicación transmita o reciba su primer byte de datos [16].

5.2.2. Encriptacion entre el protocolo I2C

Debido a la implementación de una red de sensores segura, se hizo necesario cifrar la información que viaja entre los esclavos y el maestro, para ellos se uso la librería Data Encryption Standard-DES [46]. La clave DES consta de 64 dígitos binarios (“0”ó “1”s) de los cuales 56 bits son generados al azar y utilizados directamente por el algoritmo. Los otros 8 bits, que no son utilizados por el algoritmo, se pueden utilizar para la detección de errores. Sin embargo las claves DES se han roto en menos

de 24 horas, por ello se desarrollo el triple cifrado DES-TDES, se refiere a un llave que consta de tres claves DES, que también se conoce como un paquete de clave y ha hecho seguro el método DES. [43].

A partir del ejemplo "DESexample"que trae la librería para Arduino que implementar el cifrado del TDES, entre el maestro y los esclavos se realizaron una serie de modificaciones con el fin de encriptar los datos que recogerán los sensores. Los códigos implementados en Anexos 8.

5.3. Configuración del sistema

Para poner en marcha nuestro sistema se hicieron pruebas en un servidor local con el objetivo de garantizar los objetivos propuestos y con el fin de emigrar a un servidor remoto para que dicho sistema se encuentre disponible en cualquier lugar del mundo.

5.3.1. Versión 1: Local

La primeras pruebas de control del sistema se hicieron locamente gracias al servidor de aplicaciones XAMPP [1] debido a que es el entorno más popular de desarrollo con PHP, es una distribución de Apache completamente gratuita y fácil de instalar que contiene MySQL [21], PHP y Perl. El paquete de instalación de XAMPP ha sido diseñado para ser increíblemente fácil de instalar y usar. Se implemento el control ON/OFF de un led por medio del servidor XAMPP a partir del comando GET, dicho código se menciona antes.

Fue necesario modificar el código HTML para que el Arduino uno reciba las ordenes generadas a partir del servidor local se realiza por medio de una IP. Ver Anexos 8.

De las ventajas de XAMPP es que permite crear bases de datos con MYSQL a partir de esto se creó una base de datos para asignar el ID/PASSWORD a el usuario. Con diferentes archivos PHP que permitieron, el registro, login, perfil, logout, entre otros. Sin embargo XAMPP solo se puede utilizar en una red local, lo que implica

que los usuarios solo pueden controlar y visualizar la información de sus dispositivos solo en su hogar, por esto se fue necesario migrar la aplicacion a un servidor remoto donde se encuentre disponible desde cualquier parte del mundo.

5.3.2. Versión 2: Remoto

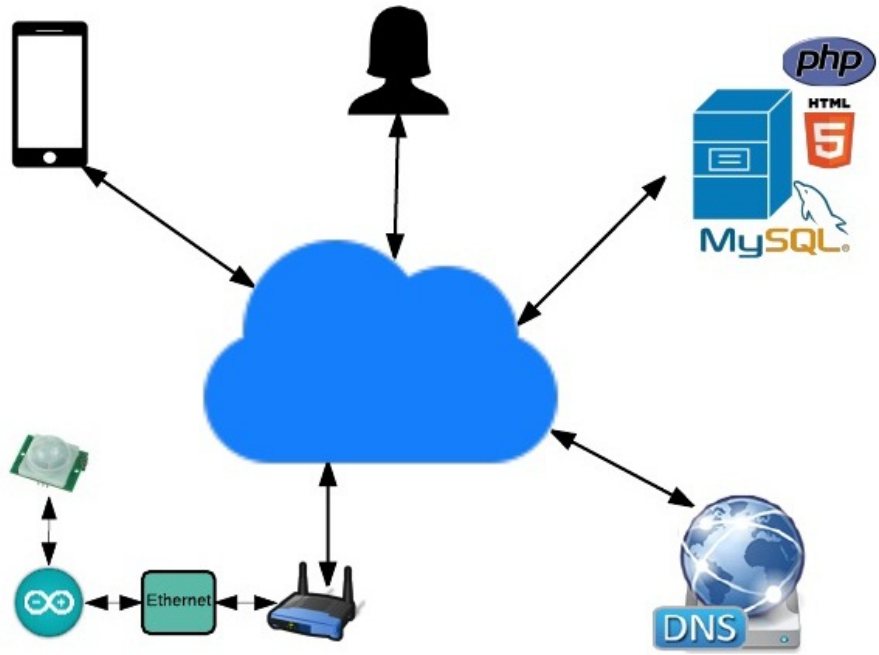

Figura 5.7: Servidor en la Nube

Las pruebas con el servidor remoto se hicieron en la nube gracias a Ormuco[15] donde creamos una máquina virtual con 1 VCPU, 1 GB de RAM, 50 GB de almacenamiento usando Windows Server 2012 R2[1] y contamos con una dirección IP física que lo pone a disposición de cualquier parte, cabe resaltar que este servidor también es compatible con la base de datos MySQL, dicho modelo que se utilizó se observa

[1]Microsoft Cloud Platform vision. http://www.microsoft.com/en-us/server-cloud/products/windows-server-2012-r2/

en la figura 5.8 la pagina web se acomoda a cualquier pantalla. Para el desarrollo de la pagina web se uso Boostrap [50].

Los detalles de la configuración de la pagina web en el servidor, se describen detalladamente en los Anexos 8.

Figura 5.8: Modelo de Base de datos

Debido a el poco desarrollo y algunas restricciones con que cuenta la librería Ethercard [51] del modulo Lan ENC28J60, se opto por utilizar la Ethernet shield que usa la librería Ethernet [36], ya que es la estándar de Arduino. A partir de ejemplo WebClient.ino [37] se hicieron modificaciones del código se hicieron con el fin de hacer que el micro controlador pregunte constantemente el estado de un led a un servidor.

Con esto se valido el protocolo de comunicación a partir del servidor web, las modificaciones realizadas al WebClient.ino se agregaron al código del micro-controlador Maestro, quien hace la petición al servidor para saber el estado del led del esclavo.

Sin embargo se necesitaba que la comunicación entre el servidor y los dispositivos fuese bidireccional, con el fin de garantizarle al usuario que realmente se esta ejecutando la orden que mande desde la pagina web. Los códigos usados en Anexos 8.

Con el fin de incorporar los sensores en nuestra red, el siguiente paso fue la implementación de los sensores de temperatura, de movimiento y de corriente, los cuales enviaran los datos recogidos al maestro y este ultimo a la aplicacion web, quien cada 5 minutos refrescara los datos obtenidos. Para el dispositivo maestro se agregaron ciertas lineas de código, las cuales se encargan de solicitar los datos de los sensores a los esclavos (Ver Anexos 8).

Los códigos que se implementaron en los esclavos a los cuales se conectaron los sensores de movimiento y temperatura, se pueden ver en los Anexos 8.

Para el sensor de corriente, basándonos en la librería EmonLib [28], se hicieron

modificaciones en los códigos de acuerdo a el tipo de transformador cuya relación es de 8:1 y por medio de la formula para determinar el consumo que se observa a continuación, el código el esclavo enviara los datos del consumo en KW/h. Las modificaciones en el código en Anexos 8.

$$Consumo = Dias\ x\ \frac{Watts}{1000}\ x\ Horas$$

Capítulo 6.
Resultados y Discusión

Hemos demostrado que es posible utilizar plataformas de hardware abierto como Arduino y Tiva C para conectar dispositivos a Internet de forma segura. Se demuestra que es posible garantizar condiciones de seguridad, tales como: confidencialidad, integridad, disponibilidad, autenticación, control de acceso y no repudio.

La aplicación web se encuentra disponible en http://192.207.60.50

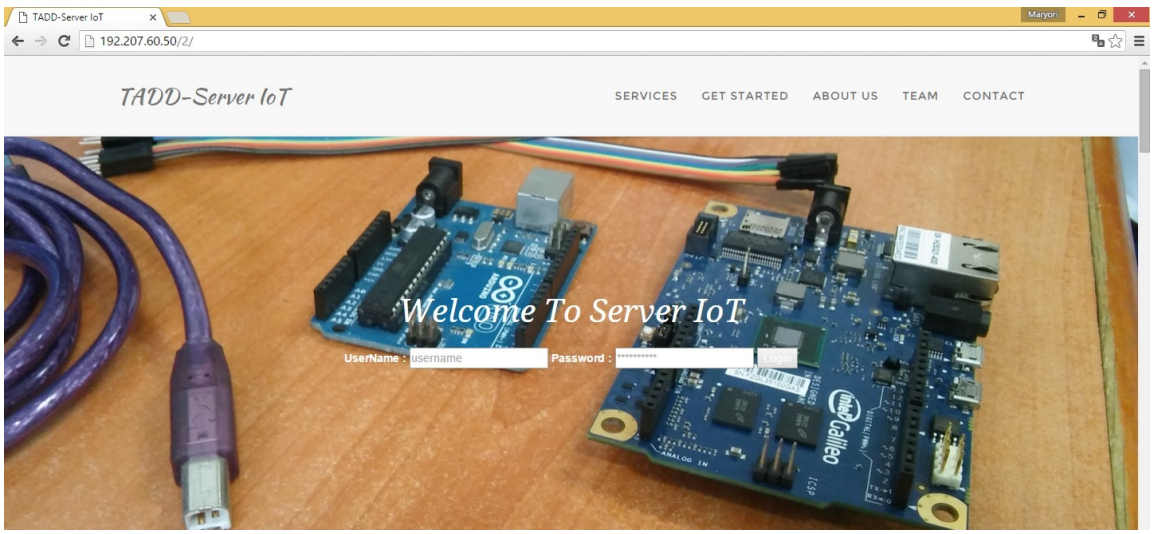

Figura 6.1: Pagina web del sistema

Hemos seguido la recomendación principal para proteger nuestra aplicación web contra muchos tipos de ataque que los usuarios no deseados pueden utilizar para

obtener el control de cuentas de otros usuarios, eliminar cuentas y/o cambiar los datos de [39]. Este enfoque se mezcla el filtrado de datos, cifrado y otros métodos para hacer más difícil para cualquier ataque. Tratamos de defenderse: Inyecciones SQL, Secuestro de sesión, la red de escuchas ilegales, Cross Site Scripting, los ataques de fuerza bruta, ataques de canal de sincronización encubiertas. Hasta ahora, hemos tratado de cubrir todas las vulnerabilidades conocidas.

Para asegurar las "cosas", hemos puesto en marcha una comunicación cifrada entre ellos. Se utilizo el método de encriptacion TDES (Triple Data Encryption Standard) presentado por el estándar FIPS en 1977 y confirmado en 1998 [19].

Por último, para asegurar los datos, codificamos la base de datos utilizando un enfoque de clave pública. De esta manera, podemos asegurar la integridad, así como añadir características de no repudio.

Luego de realizar la autenticación el usuario podrá acceder en el perfil donde visualizara y seleccionara los lugares que tiene registrado, como por ejemplo: Casa1/cuarto. Previamente observara las luces que puede controlar desde la aplicacion y los datos recogidos por los sensores en tiempo real. En la figura 6.3 se visualiza este perfil del usuario registrado.

En la figuras 6.4 y 6.5 resultan de presionar el botón "SEE DETAILS", la gráfica se refrescara cada 10 datos tomados por los sensores. La Figura 6.4 muestra las mediciones de temperatura en un cuarto. La Figura 6.5 muestra el consumo de potencia del Router inalambrico, debido a que se conecto el sensor de corriente y se tuvo en

cuenta por la hoja de datos del router [53], se sabe que trabaja con:

- Input 100-240V 50/60Hz 0.3A
- Output 9V/0.6A

Teóricamente el consumo de potencia eléctrica mensualmente es:

$$9V\ x\ 0{,}6A = 5{,}4W$$

$$Consumo = Dias\ x\ \frac{Watts}{1000}\ x\ Horas$$

Reemplazando en la Formula, obtenemos:

$$Consumo = 30\ x\ \frac{5{,}4}{1000}\ x\ 24 = 3{,}88$$

En la figura 6.2 se observan los valores experimentales obtenidos con nuestro sensor de corriente, cuyos valores promedios son:

- Voltaje= 126.7V
- Corriente= 0.51A
- Potencia= 64.5W
- Consumo= 4 Kw/h

El error porcentual entre el valor esperado y el medido es:

$$Error\,\% = |\frac{3{,}88 - 4}{3{,}88}|\ x\ 100 = 3{,}1$$

En la figura 6.5 se observa el diagrama de consumo de potencia en la aplicacion web.

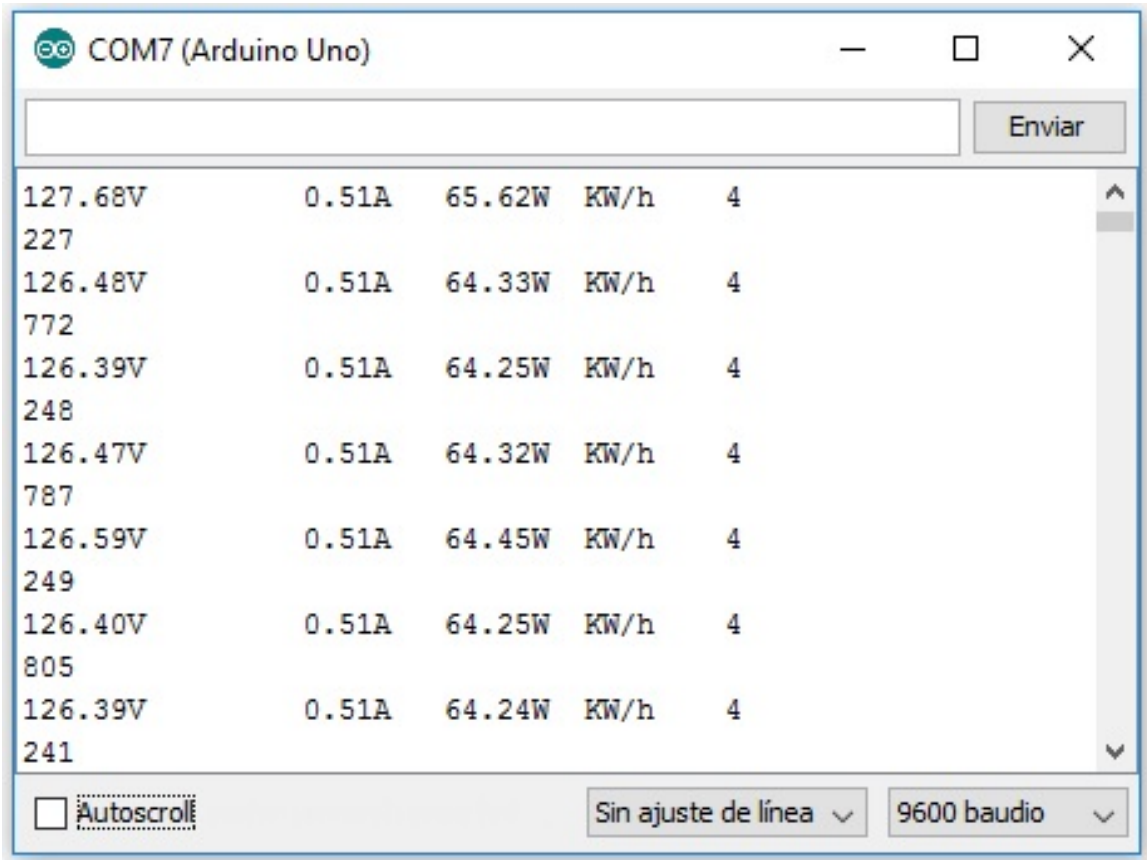

Figura 6.2: Medidas del sensor de corriente

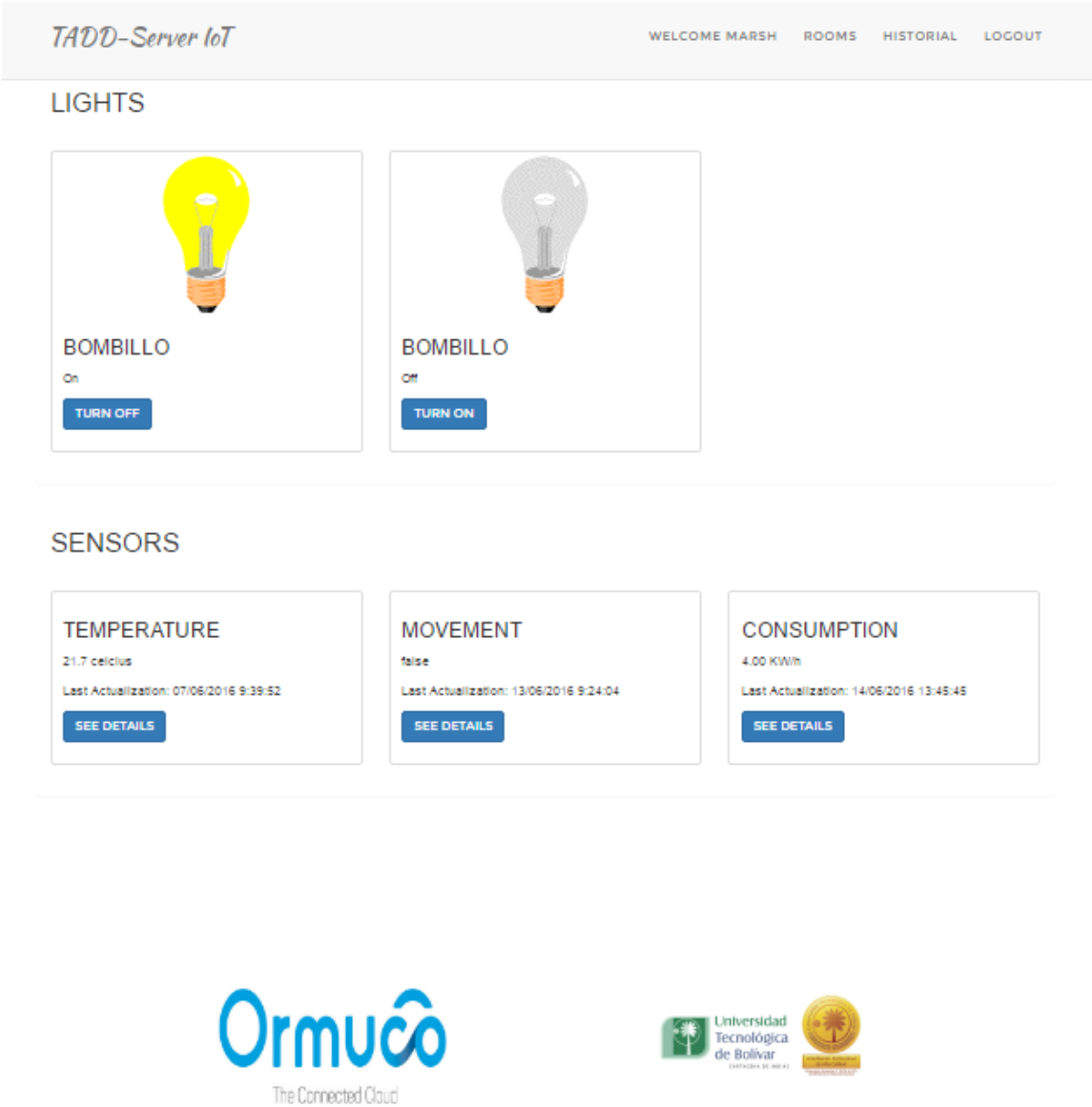

Figura 6.3: Perfil de usuario en el sistema

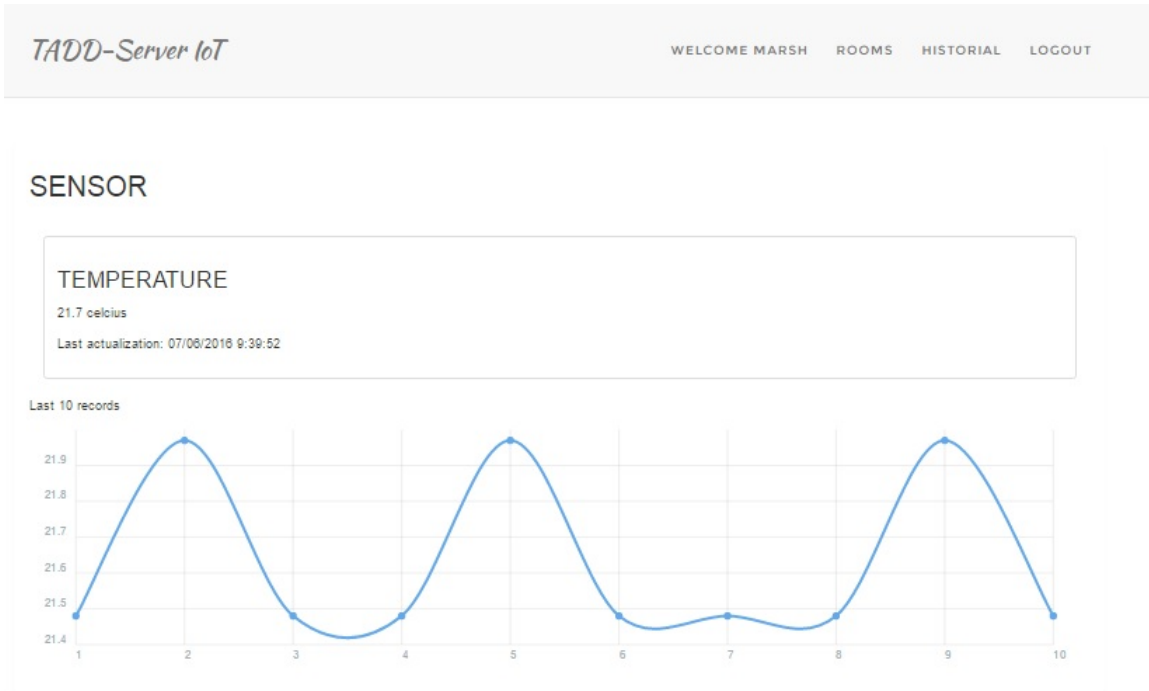

Figura 6.4: Medidas de la Temperatura

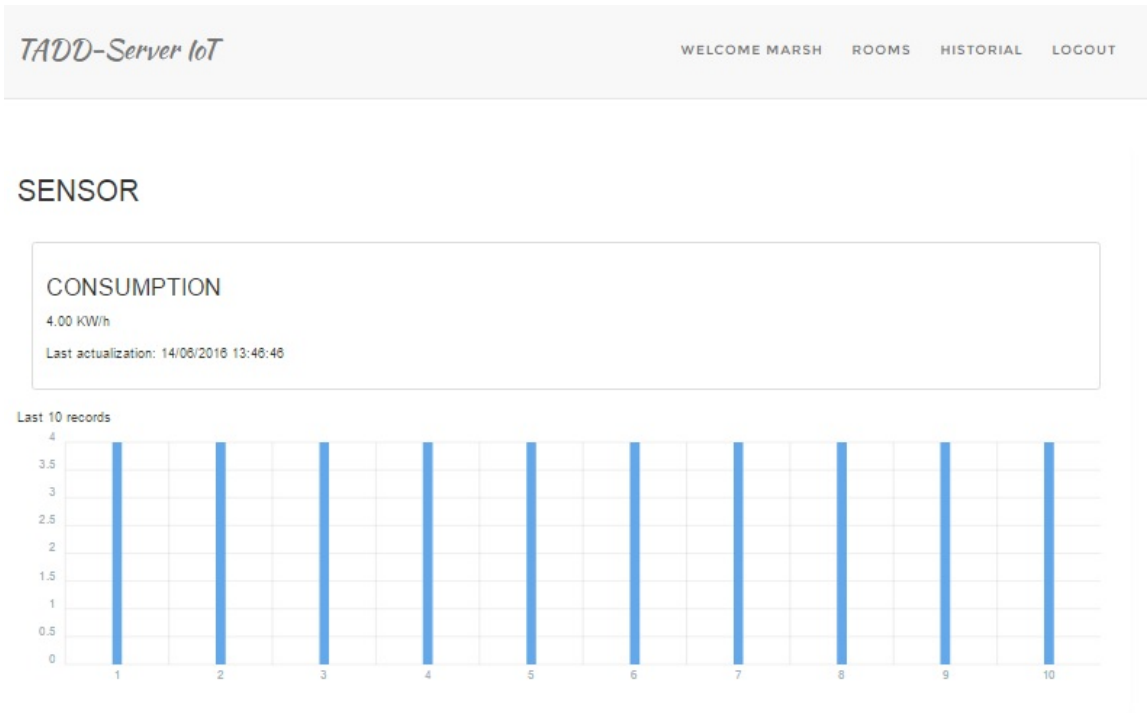

Figura 6.5: Medidas del consumo de potencia

Capítulo 7.
Conclusiones y Recomendaciones

Cuando hablamos de la Internet de las cosas (IoT), no nos referimos sólo a la conexión de las cosas inteligentes, también tenemos que pensar en la seguridad de la información. En este trabajo se describe un sistema de bajo costo de hardware/software que se ajuste a los estándares FIPS 199 e ISO 7498-2. Por confidencialidad, se utiliza el cifrado de clave simétrica y pública. El cifrado también nos ayuda a garantizar la integridad y el no repudio de los datos almacenados en la base de datos como las transmisiones entre los actores del sistema. Para garantizar la disponibilidad, nos basamos en las buenas prácticas de codificación en conjunto con servicios de alojamiento que proporcionan la infraestructura tales como cortafuegos, proxies, cachés, redundancia, etc., que nos ayudan a mitigar los ataques denegado-de-servicio y ataques relacionados. Un sistema de autenticación permite establecer una identificación y una contraseña para cada usuario y que conectado a nuestro sistema, así como la base de datos. También hemos creado diferentes funciones que permiten manejar nuestro sistema. Hemos demostrado que podemos tener un sistema seguro de IoT en nuestros hogares.

Se desarrollo una aplicacion web capaz de mantener un registro de control y

monitoreo de las actividades realizadas por cada usuario registrado. Para los sensores se proporciono una interfaz gráfica que permite la visualización día a día de los datos recogidos, para el caso del consumo eléctrico se despliega un diagrama del consumo del mes en Kw/h.

Recomendación: La utilización de la shield de Ethernet sobre el arduino, debido a la magnitud de procesamiento que realiza este dispositivo que funciona como maestro, se advierte sobre la baja memoria en el serial después de compilar. Por esto se recomienda ampliarle la memoria al Arduino, si se desea agregar mas funciones al sistema.

El estudio e implementación del sensor de corriente con el fin de brindar a el usuario información del voltaje, corriente, potencia real, potencia aparente y factor de potencia, se llevo a cabo en una investigación anterior a este trabajo y los resultados obtenidos se presentaron en el XI Encuentro Departamental de Semilleros de Investigación (EDESI 2014) Nodo Bolívar, el cual obtuvo el aval para el XVII Encuentro Nacional Y XI Internacional de Semilleros de Investigación 2014 realizado en la ciudad de Tunja. También en el CIIMA 2014 (Congreso Internacional de Ingeniera Mecatronica y Automatización) donde se obtuvo una publicación de este trabajo (Ver Anexos). Y por ultimo se presento en Congreso Regional en Instrumentación Avanzada-CRIA 2014 realizado en Costa Rica. A partir de los resultados obtenidos en este articulo se contribuyó al progreso de esta investigación, debido a que se vio la necesidad de llevar estos resultados a una aplicacion web con el fin de que el usua-

rio pudiese visualizar el consumo de potencia eléctrica de algunos dispositivos en su hogar.

Los Resultados preliminares de esta investigación se presentaron en el XII Encuentro departamental de semilleros de investigación (EDESI 2015), el cual obtuvo el aval para el XVIII Encuentro Nacional Y XII Internacional de Semilleros de Investigación (ENISI 2015) Realizado en la ciudad de Cali.

Bibliografía

[1] Xampp apache + mysql + php + perl. urlhttps://www.apachefriends.org/es/index.html, 2015.

[2] Arduino - Home, February 2016.

[3] Energia, February 2016.

[4] A. B. Ahmed Lounis, Abdelkrim Hadjidj and Y. Challal. Secure and scalable cloud-based architecture for e-health wireless sensor networks. *International Conference on Computer Communication Networks (ICCCN),Munich-Germany*, 1, 2012.

[5] K. Ashton. That 'internet of things' thing. *RFiD Journal*, 22(7):97–114, 2009.

[6] Atmel. Atmega 16, 2015.

[7] D. M. bautista. Sistema de comunicaciones basado en ethernet para el control de sistemas empotrados. *Universidad Tecnológica de Mixteca*, 1:25–80, 2009.

[8] T. Bernes. Html: Hypertext markup language. *http://www.hipertexto.info/documentos/html.htm*, 1991.

[9] A. R. Capel. Diseño y desarrollo parcial de un sistema domótico para facilitar la movilidad de los minusválido. *Universidad politécnica de Cataluña*, 1, 2005.

[10] M. R. Cerezo. Sistema de control remoto para aplicaciones domóticas a través de internet. *Universidad Autónoma de Madrid-Escuela Politécnica Superior*, 1, 2014.

[11] M. Christodorescu. Private use of untrusted web servers via opportunistic encryption. *IEEE*, 1, 2008.

[12] G. Coley. Beaglebone black system reference manual. *Texas Instruments, Dallas*, 2013.

[13] desarrolloweb.com. Soap. url http://www.desarrolloweb.com/articulos/1557.php, 2016.

[14] DIGI. Xbee. urlhttp://www.digi.com/products/xbee-rf-solutions, 2015.

[15] B. B. Dominic Larose, Orlando Bayter. Ormuco: The connected cloud. urlhttps://http://www.ormuco.com/, 2015.

[16] L. ECHUN Electronic Co. The ssl protocol, 2012.

[17] M. S. Espinoza. Diseño de un sistema modular de control residencial remoto a través de internet. *Universidad de Costa Rica*, 1, 2009.

[18] S. Fernández. La criptografia clásica. *SIGMA*, 2004.

[19] P. FIPS. 46-3: Data encryption standard (des). *National Institute of Standards and Technology*, 25(10):1–22, 1999.

[20] D. Gislason. Zigbee wireless networking. *Zigbee alliance*, 1, 2008.

[21] J. Greenspan and B. Bulger. *MySQL/PHP database applications*. John Wiley & Sons, Inc., 2001.

[22] K. E. Hickman. Split core current transformer ecs1030-l72, 1995.

[23] E. a. Hribernik, Karl A. Co-creating the internet of things—first experiences in the participatory design of intelligent products with arduino. *Concurrent Enterprising (ICE), 17th International Conference on*, 1, 2011.

[24] T. Instrument. *LM35 Precision Centigrade Temperature Sensors*. "Texas Instrument", 1999-2016.

[25] T. Instruments. Tiva™ c series tm4c123g launchpad evaluation board, 2013. Google Patents.

[26] O. G. M. Jan Henrik Ziegeldorf and K. Wehrle. Privacy in the internet of things:threats and challenges. *Communication and Distributed Systems, RWTH Aachen University, Germany*, 7, 2013.

[27] C. L. Jimeno. La domótica como solución de futuro. *fundación de la energía de la comunidad de Madrid*, 1, 2007.

[28] T. Lea. Arduino energy monitoring library, 2010.

[29] R. Lerdorf. Php: Hypertext preprocessor. *URL http://php. net*, 1995.

[30] A. Lindsay. Ethernet.localip(), 2011. Google Patents.

[31] M. Margolis. *Arduino Cookbook*. O'REILLY, 2011.

[32] M. Margolis. *Arduino cookbook*. .º'Reilly Media, Inc.", 2011.

[33] A. R. I. Martha Carvajal Aguilar. Diseño e implementación de un sistema de control y seguridad en tiempo real de una vivienda inteligente. *Escuela superior politécnica de Chimborazo*, 1, 2014.

[34] D. K. R. H. B. R. K. W. Martin Henze, Lars Hermerschmidtz. User-driven privacy enforcement for cloud-based services in the internet of things. *Communication and Distributed Systems, RWTH Aachen University, Germany*, 1, 2014.

[35] I. MB. Http request/response, 2009.

[36] D. A. Mellis. Ethernet library, 2009.

[37] D. A. Mellis. Webclient, 2009.

[38] J. S. Miguel A. Zamora-Izquierdo and A. F. Gómez-Skarmeta. An integral and networked home automation solution for indoor ambient intelligence. *Pervasive Computing, IEEE*, 9(4), 2010.

[39] R. Nixon. *Learning PHP, MySQL, JavaScript, and CSS: A step-by-step guide to creating dynamic websites*. .º'Reilly Media, Inc.", 2012.

[40] N. I. of Standards and T. (NIST). Standards for security categorization of federal information and information systems. *FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION*, 1, 2014.

[41] G. G. paredes. Introducción a la criptografía. *Revista digital universitaria*, 1, 2006.

[42] PIC. El microcontrolador pic16f873, 2015.

[43] F. PUB. Data encryption standard (des), 1999. Google Patents.

[44] M. C. Ramon. *Intel galileo and intel galileo gen 2*. Springer, 2014.

[45] H. W. M. H. H. S. K. W. René Hummen, Jens Hiller. 6lowpan fragmentation attacks and mitigation mechanisms. *Communication and Distributed Systems, RWTH Aachen University, Germany*, 1, 2013.

[46] T. Riemann. Des.h library, 2013.

[47] W. Rincón. Un hacker explica qué tan fácil es robar una contraseña. *Revista Semana*, 1, 2014.

[48] D. A. O. Rodríguez. Diseño y construcción de un sistema inteligente de sensado y control para aplicaciones residenciales. caso: edificio shalom. *universidad internacional del ecuador*, 1, 2014.

[49] M. M. Sabalza, J. D. B. O., and J. C. M. Santos. Design and construction of a power meter to optimize usage of the electric power. In *Engineering Mechatronics and Automation (CIIMA), 2014 III International Congress of*, pages 1–5, Oct 2014.

[50] S. Santo. Best way to customize bootstrap css style. *Bootstrap framework*, 2014.

[51] G. Socher and P. Stang. Ethercard library, 2014.

[52] I. p. s. Technical Committee ISO/TC 97. Iso7498-2:information processing system, open system interconnection, basic reference mode.part 2: Security architecture. *ISO*, 1, 1989.

[53] L. TP-LINK TECHNOLOGIES CO. Tp-link user guide tl-wr841nd 300mbps wireless n router, 2014. Google Patents.

[54] N. Zambetti. Master reader/slave sender. urlhttps://www.arduino.cc/en/Tutorial/MasterReader, 2006.

[55] N. Zambetti. Wire.h library, 2006.

Capítulo 8.
Anexos

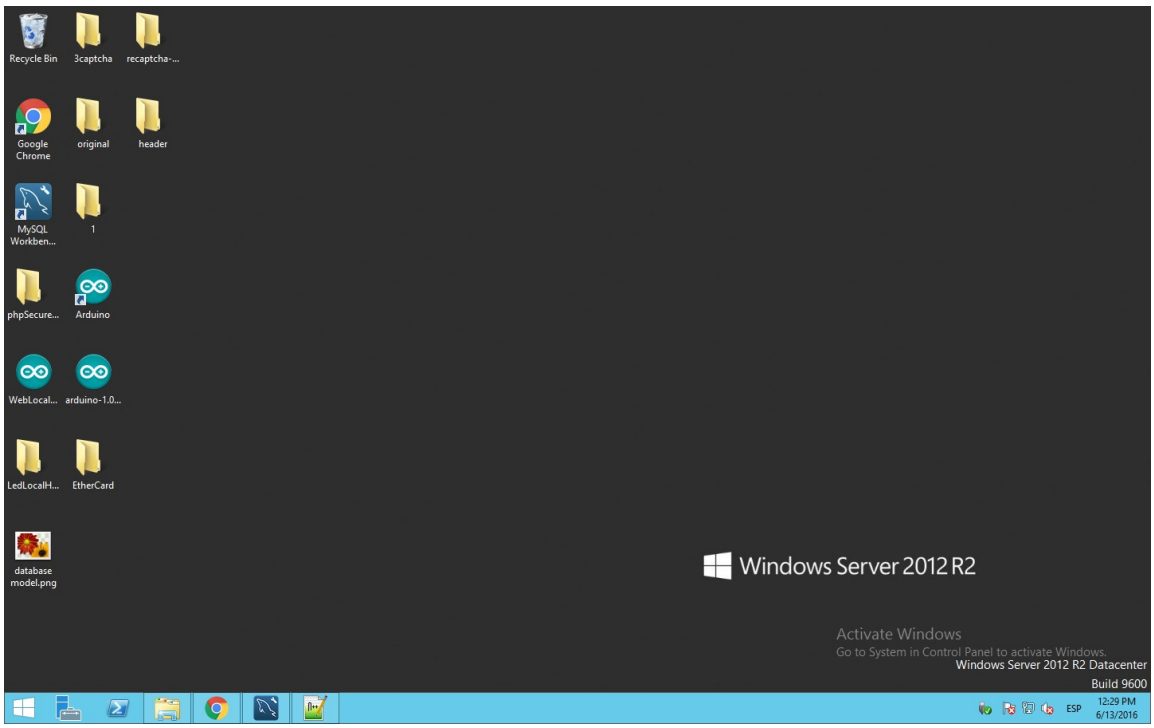

Figura 8.1: Maquina virtual

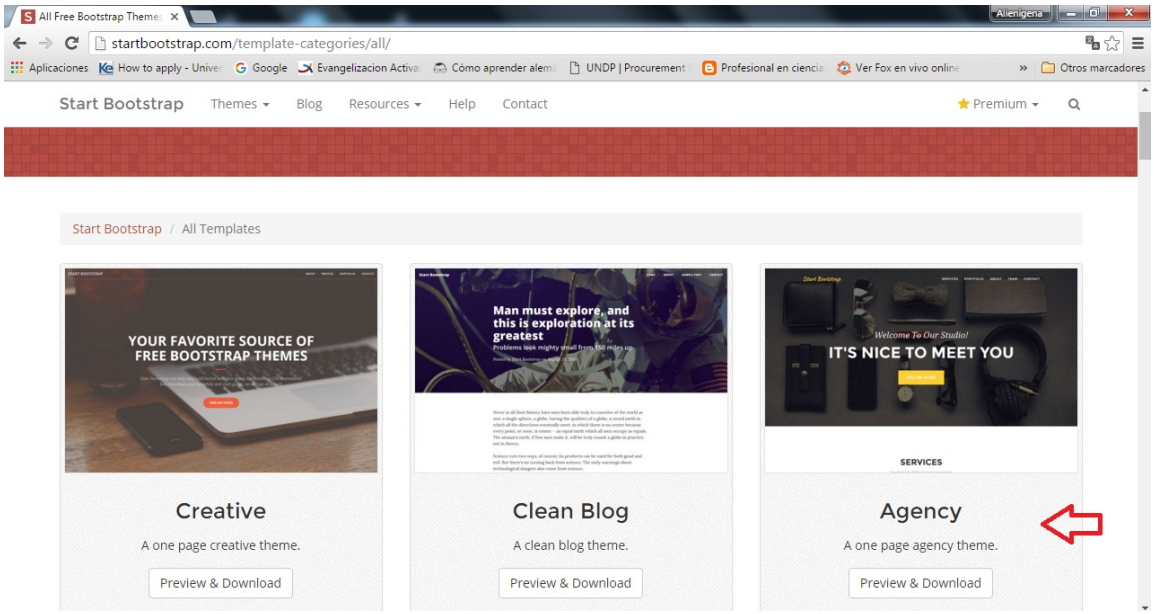

Figura 8.2: Plantilla de Boostrap

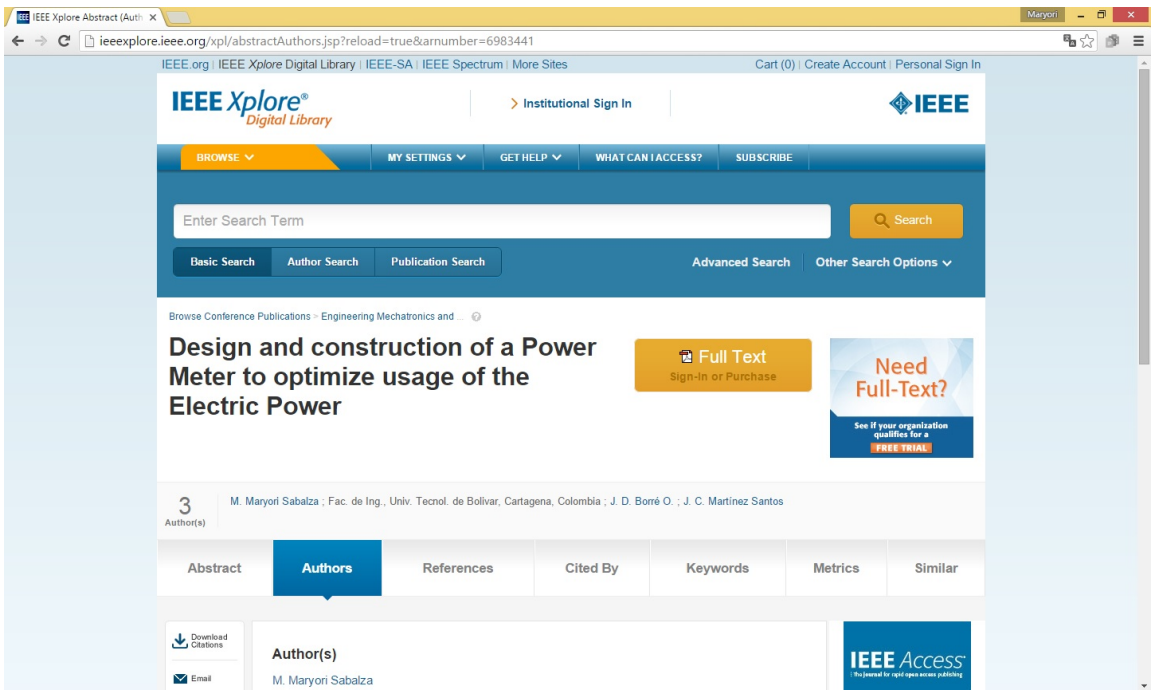

Figura 8.3: Publicación del Articulo

Figura 8.4: Presentación del trabajo en el CRIA 2014

EL SUSCRITO COORDINADOR EJECUTIVO DE LA RED COLOMBIANA DE SEMILLEROS DE INVESTIGACIÓN- REDCOLSI NODO BOLÍVAR

HACE CONSTAR:

Que el proyecto en ejecución "MEDICIÓN DE POTENCIA ELÉCTRICA PARA OPTIMIZAR EL USO DE ENERGÍA ELÉCTRICA" fue presentado en el marco XI Encuentro Departamental de Semilleros de Investigación - XI EDESI, por los ponentes Maryori del Carmen Sabalza Mejia y Jesus David Borre Ordosgoitia, estudiantes de pregrado pertenecientes al semillero TECNICAS AVANZADAS DE DISEÑO DIGITAL – TADD, en representación de la Universidad Tecnológica de Bolívar. El XI EDESI se realizó del 22 al 23 de Mayo del 2014 en las instalaciones del Centro multisectorial del Sena, ubicado en el Km 1 vía a Turbaco.

De igual forma se certifica, que el proyecto una vez sustentado en el Encuentro Departamental obtuvo un puntaje de 95.5 sobre cien (100) en el promedio de las dos evaluaciones efectuadas, obteniendo el aval al XVII Encuentro Nacional y XI Internacional de Semilleros de Investigación RedCOLSI.

Dado en Cartagena, a los 10 días del mes de Junio de 2014.

JUAN SEBASTIÁN HERNÁNDEZ COBOS
Coordinador Ejecutivo
RedCOLSI NODO BOLÍVAR
Visita: www.fundacionredcolsi.org

www.edesinodobolivar.com

Figura 8.5: Presentación del trabajo en EDESI 2014

Figura 8.6: Presentación del trabajo en EDESI 2015

Códigos en C++

Listing 8.1: Código testDHCP.ino

```
#include <EtherCard.h>

static byte mymac[] = { 0x74,0x69,0x69,0x2D,0x30,0x31 };

byte Ethernet::buffer[700];

void setup () {
  Serial.begin(57600);
  Serial.println(F("\n[testDHCP]"));

  Serial.print("MAC: ");
  for (byte i = 0; i < 6; ++i) {
    Serial.print(mymac[i], HEX);
    if (i < 5)
      Serial.print(':');
  }
  Serial.println();

  if (ether.begin(sizeof Ethernet::buffer, mymac) == 0)
    Serial.println(F("Failed to access Ethernet controller"));

  Serial.println(F("Setting up DHCP"));
  if (!ether.dhcpSetup())
    Serial.println(F("DHCP failed"));

  ether.printIp("My IP: ", ether.myip);
  ether.printIp("Netmask: ", ether.netmask);
  ether.printIp("GW IP: ", ether.gwip);
  ether.printIp("DNS IP: ", ether.dnsip);
}

void loop () {}
```

Listing 8.2: Código para "Hello Word Arduino"desde dirección IP

```

//Ethernet IP, default gateway and MAC addresses
static byte myip[] = { 192,168,1,48 };
static byte gwip[] = { 192,168,1,1 };
```

```
5  static byte mymac[] = { 0x5A, 0xEE, 0x65, 0x18, 0x60, 0xED };
6  //TCP/IP send and receive buffer
7  byte Ethernet::buffer[500];
8
9  char page[] PROGMEM =
10 "HTTP/1.0 503 Service Unavailable\r\n"
11 "Content-Type: text/html\r\n"
12 "Retry-After: 600\r\n"
13 "\r\n"
14 "<html>"
15   "<head><title>"
16     "Arduino 192.168.1.48"
17   "</title></head>"
18   "<body>"
19     "<h1>Hello World</h1>"
20     "<h3>Maryo</h3>"
21   "</body>"
22 "</html>";
23
24 void setup(){
25   Serial.begin(57600);
26   Serial.println("\n[Hello World]");
27
28   if (ether.begin(sizeof Ethernet::buffer, mymac) == 0)
29     Serial.println( "Failed to access Ethernet controller");
30   ether.staticSetup(myip, gwip);
31
32   ether.printIp("IP:  ", ether.myip);
33   ether.printIp("GW:  ", ether.gwip);
34   ether.printIp("DNS: ", ether.dnsip);
35 }
36
37 void loop(){
38 //wait for an incoming TCP packet, but ignore its contents
39   if (ether.packetLoop(ether.packetReceive())) {
40     memcpy_P(ether.tcpOffset(), page, sizeof page);
41     ether.httpServerReply(sizeof page - 1);
42   }
43 }
```

Listing 8.3: Comando GET

1

```
2  void loop() {
3
4    word len = ether.packetReceive();
5    word pos = ether.packetLoop(len);
6
7    if(pos) {
8
9     if(strstr((char *)Ethernet::buffer + pos, "GET /?status=ON") != 0) {
10       Serial.println("Received ON command");
11       ledStatus = true;
12     }
13
14    if(strstr((char *)Ethernet::buffer + pos, "GET /?status=OFF") != 0) {
15       Serial.println("Received OFF command");
16       ledStatus = false;
17     }
18
19     if(ledStatus) {
20       digitalWrite(ledPin, HIGH);
21       statusLabel = on;
22       buttonLabel = off;
23     } else {
24       digitalWrite(ledPin, LOW);
25       statusLabel = off;
26       buttonLabel = on;
27     }
```

Listing 8.4: Código para el control del sensor PIR por IP

```
1   /*
2       https://www.youtube.com/watch?v=pPIdDl1Cg1Y
3                 Maryori Sabalza Mejia
4                   Abril 3 2015
5 */
6
7 #include <EtherCard.h>
8
9 #define DEBUG   1
10 #define SERIAL  1
11
12 static char outCount = -1;
13
14 //Configuration, as stored in EEPROM
```

```
15 struct Config {
16     byte band;
17     byte group;
18     byte collect;
19     word refresh;
20     byte valid; // keep this as last byte
21 } config;
22
23 //Number of messages saved in history
24 #define NUM_MESSAGES  10
25 //Truncate message payload to reduce memory use
26 #define MESSAGE_TRUNC 15
27
28 //Used as cursor while filling the buffer
29 static BufferFiller bfill;
30
31 static byte history_rcvd[NUM_MESSAGES][MESSAGE_TRUNC+1];
32 static byte history_len[NUM_MESSAGES];
33 static byte next_msg;
34 static word msgs_rcvd;
35 byte ledstatus = 0;
36 //TCP/IP send and receive buffer
37 byte Ethernet::buffer[1000];
38
39 //Ethernet interface mac address
40 static byte mymac[] = { 0x74,0x69,0x69,0x2D,0x30,0x31 };
41 //Ethernet interface ip address
42 static byte myip[] = { 192,168,1,50 };
43 //Gateway ip address
44 static byte gwip[] = { 192,168,1,1 };
45
46 #if DEBUG
47 static int freeRam () {
48   extern int __heap_start, *__brkval;
49   int v;
50   return (int) &v - (__brkval == 0 ? (int) &__heap_start : (int) __brkva
51 }
52 #endif
53
54 /////////////////  PIR sensor de Movimiento ///////////////
55 const int buttonPin = 7;
56 int buttonState = 0;
```

```
57 int cont=0;
58
59 void setup(){
60
61    pinMode(2, OUTPUT);
62    pinMode(buttonPin, INPUT);
63
64 #if SERIAL
65     Serial.begin(57600);
66     Serial.println("\n[etherNode]");
67 #endif
68
69
70     if (ether.begin(sizeof Ethernet::buffer, mymac) == 0)
71       Serial.println( "Failed to access Ethernet controller");
72     ether.staticSetup(myip, gwip);
73
74 #if SERIAL
75     ether.printIp("IP: ", ether.myip);
76 #endif
77 }
78 char okHeader[] PROGMEM =
79     "HTTP/1.0 200 OK\r\n"
80     "Content-Type: text/html\r\n"
81     "Pragma: no-cache\r\n"
82 ;
83
84 static void homePage(BufferFiller& buf, byte led) {
85
86     buf.emit_p(PSTR("$F\r\n"
87         "<meta http-equiv='refresh' content='$D'/>"
88         "<title>Estado de un PIR por internet</title>"
89
90         "<body style=background-color:lightblue>"
91         "<body><h1><center>TADD<center></h1>"
92         "<body><p><h2>Maryori Sabalza Mejia.<h2></p>"
93
94         "<body><p>Se Dectecto Movimiento: $D</br>"
95        // "<body><p> Movimiento: content='$D' </br>"
96
97        "<a href='e'>Detectar Movimientos con luz</a><br></p>"
98        "<a href='a'>Volver a Detectar Movimientos sin Luz</a></br>"), ok
```

```
99
100     // Cronometrar el tiempo de ejecucion del programa
101     long t = millis() / 1000;
102     word h = t / 3600;
103     byte m = (t / 60) % 60;
104     byte s = t % 60;
105     buf.emit_p(PSTR(
106         "Tiempo corriendo $D$D:$D$D:$D$D</body>"), h/10, h%10, m/10, m%
107 #if DEBUG
108     buf.emit_p(PSTR(" ($D bytes free)"), freeRam());
109     // if $D=1 THEN Serial.println("Hay personas en la casa");
110 #endif
111 }
112 void loop(){
113     word len = ether.packetReceive();
114     word pos = ether.packetLoop(len);
115     // check if valid tcp data is received
116     if (pos) {
117         bfill = ether.tcpOffset();
118         char* data = (char *) Ethernet::buffer + pos;
119
120 /////////////// PIR ///////////////
121   buttonState = digitalRead(buttonPin);
122
123 #if SERIAL
124         Serial.println(data);
125 #endif
126
127
128         // Pagina sin solicitud
129         if (strncmp("GET / ", data, 6) == 0) {
130             homePage(bfill, buttonState);
131         }
132         else if (strncmp("GET /a", data, 6) == 0) {
133             // Recibimos el parametro a (apagar)
134             homePage(bfill, buttonState);
135             //configPage(data, bfill);
136             if(buttonState == LOW) {
137               digitalWrite(2, LOW);
138               buttonState = LOW;
139             }
140         }
```

```
141             else if (strncmp("GET /e", data, 6) == 0) {
142                 // Recibimos el parametro e (encender)
143                 homePage(bfill, buttonState);
144                 if(buttonState == HIGH) {
145                   digitalWrite(2, HIGH);
146                   buttonState = HIGH;
147                     delay(10);
148                   cont++;
149                   Serial.println(cont);
150                 }
151 
152           Serial.println(buttonState);
153 
154         delay(1000);
155           }
156           else
157                bfill.emit_p(PSTR(
158                    "HTTP/1.0 401 Invalido\r\n"
159                    "Content-Type: text/html\r\n"
160                    "\r\n"
161                    "<h1>401 Unauthorized</h1>"));
162           //Send web page data
163           ether.httpServerReply(bfill.position());
164       }
165 }
```

Listing 8.5: Código Para el Micro-controlador Maestro

```
1 
2 //Maryori Sabalza 05-08-2015
3 
4 #include <Wire.h>
5 
6 //Number of rows in the keypad
7 const int numRows = 4;
8 //Number of columns
9 const int numCols = 4;
10 //Number of milliseconds for switch to be stable
11 const int debounceTime = 20;
12 //keymap defines the character returned when
13 //the corresponding key is pressed
14 const char keymap[numRows][numCols] = {
15   { '1', '2', '3', 'A' } ,
```

```
16    { '4', '5', '6', 'B' } ,
17    { '7', '8', '9', 'C' } ,
18    { '*', '0', '#', 'D' }
19  };
20  // this array determines the pins used for rows and columns
21  //Rows 0 through 3
22  const int rowPins[numRows] = { 5, 4, 3, 2};
23  //Columns 0 through 2
24  const int colPins[numCols] = { 9, 8, 7, 6};
25
26
27  void setup()
28  {
29    Serial.begin (9600);
30    //join i2c bus (address optional for master)
31    Wire.begin();
32    for (int row = 0; row < numRows; row++)
33    {
34      //Set row pins as input
35      pinMode(rowPins[row], INPUT);
36      //Turn on Pull-ups
37      digitalWrite(rowPins[row], HIGH);
38    }
39    for (int column = 0; column < numCols; column++)
40    {
41     //Set column pins as outputs for writing
42     pinMode(colPins[column], OUTPUT);
43     // Make all columns inactive
44     digitalWrite(colPins[column], HIGH);
45    }
46  }
47
48
49
50  void loop()
51  {
52    char key = getKey();
53
54    //transmit to device #4
55    Wire.beginTransmission(4);
56    //sends one byte
57    Wire.write(key);
```

```
58    //stop transmitting
59    Wire.endTransmission();
60
61    //transmit to device #8
62    Wire.beginTransmission(8);
63    //sends one byte
64    Wire.write(key);
65    //stop transmitting
66    Wire.endTransmission();
67  }
68  // returns with the key pressed, or 0 if no key is pressed
69  char getKey()
70  {
71    // 0 indicates no key pressed
72    char key = 0;
73    for (int column = 0; column < numCols; column++)
74    {
75      //Activate the current column.
76      digitalWrite(colPins[column], LOW);
77      //Scan all rows for a key press.
78      for (int row = 0; row < numRows; row++)
79      {
80        // Is a key pressed?
81        if (digitalRead(rowPins[row]) == LOW)
82        {
83          // debounce
84          delay(debounceTime);
85          // wait for key to be released
86          while (digitalRead(rowPins[row]) == LOW);
87          // Remember which key was pressed.
88          key = keymap[row][column];
89        }
90      }
91      // De-activate the current column.
92      digitalWrite(colPins[column], HIGH);
93    }
94    // returns the key pressed or 0 if none
95    return key;
96  }
```

Para los esclavos se uso el siguiente código el cual para cada esclavo solo se modifica el numero del 'Case'.

Listing 8.6: Código Para los Micro-controladores Esclavos

```
1  //Maryori Sabalza 05-08-2015
2  //Modificado el 09-08-2015
3
4  #include <Wire.h>
5
6  //int LED= 9; //Asignamos el valor 10 a la variable led
7  #define LED RED_LED
8  void setup()
9  {
10   Wire.begin(4);
11
12   Wire.onReceive(receiveEvent);
13
14   Serial.begin (9600);
15    pinMode(LED, OUTPUT);
16 }
17
18 void loop()
19 {
20 }
21
22 void receiveEvent(int howMany) {
23   //Leemos el dato recibido
24   char key = Wire.read();
25   if(key) {
26   //select case de la variable x
27   switch (key) {
28   case '1':
29     Serial.println("Encendido");
30     //enciende el led
31     digitalWrite(LED, HIGH);
32     break;
33   }
34   }
35    if(key) {
36     switch (key){
37       case '2':
38      Serial.println("Apagado");
39      //apaga el led
40     digitalWrite(LED, LOW);
41     break;
```

```
  }
}
}
```

Listing 8.7: Código de encriptacion-Maestro

```
/*
                        Ing. Ivan Banos Delgado
                           Junio 10 2016
*/

#include <DES.h>
#include <Wire.h>

DES des;



void setup() {
  Serial.begin(9600);
  Wire.begin();
  Serial.println("Hello!");
  des.init("012345677654321001234567\0", (unsigned long long int)0);
}
int x = 0;
void loop() {

  des.iv_inc();
  String mesC = "x is ";
  mesC = mesC + x;
  byte message[mesC.length() + 1];
  mesC.getBytes(message, mesC.length() + 1);
  Serial.println((char*)message);
  des.calc_size_n_pad(sizeof(message));
  byte plaintext_p[des.get_size()];
  des.padPlaintext(message, plaintext_p);
  byte cyphertext[des.get_size()];
  des.tdesCbcEncipher(plaintext_p, cyphertext);
  des.calc_size_n_pad(sizeof(cyphertext));
  des.tdesCbcDecipher(cyphertext, plaintext_p);
  bool ok = des.CheckPad(plaintext_p, sizeof(plaintext_p));
  if (ok)
    printf("padding ok!");
```

```
38    else
39      printf("padding corrupted!");
40
41    unsigned long long int iv = des.get_IV_int();
42    Serial.println("Cypher");
43    Serial.println((char*)cyphertext);
44    Serial.println("Plain");
45    Serial.println((char*)plaintext_p);
46    Serial.println("Iv");
47    Serial.println((long)iv);
48    Serial.println("Size");
49    int sizeCT = sizeof(cyphertext);
50    Serial.println(sizeCT);
51    String ivS = String((long)iv);
52    byte ivC[ivS.length() + 1];
53    ivS.getBytes(ivC, ivS.length() + 1);
54    delay(2000);
55    char* Emessage = (char*)cyphertext;
56    Wire.beginTransmission(4); // transmit to device #4
57    Wire.write(Emessage);          // sends five bytes
58    Wire.write("IV:");          // sends five bytes
59    Wire.write((char*)ivC);          // sends five bytes
60    Wire.endTransmission();     // stop transmitting
61
62    x++;
63    delay(2000);
64  }
```

Listing 8.8: Código de encriptacion-Esclavo

```
1
2  /*
3                    Ing. Ivan Banos Delgado
4                         Junio 10 2016
5  */
6
7  #include <DES.h>
8  #include <Wire.h>
9
10 DES des;
11 String cypherText = "";
12 String iv = "";
13 long long int ivI;
```

```
14 int j = 0;
15 void setup() {
16   Serial.begin(9600);
17   Serial.println("Hello!");
18   Wire.begin(4);
19   Wire.onReceive(receiveEvent);
20   des.init("01234567765432100123456\0",(unsigned long long int)0);
21 }
22
23 void loop() {
24   delay(2000);
25 }
26
27 void receiveEvent(int howMany)
28 {
29   int i = 8;
30   while (Wire.available()) // loop through all but the last
31   {
32     char c = Wire.read(); // receive byte as a character
33     Serial.print(c);// print the character
34     if(i>0){
35       cypherText +=c;
36     } else {
37       if(j == 0){
38        if(c=='I'){
39         Serial.print("OK");
40         j=1;
41        }else {
42           j=0;
43         }
44       }
45       else
46       if(j == 1){
47         if(c=='V'){
48         Serial.print("OK");
49         j=2;
50        }else {
51           j=0;
52         }
53       }
54       else
55       if(j == 2){
```

```
56          if(c==':'){
57          Serial.print("OK");
58          j=3;
59         }else {
60            j=0;
61          }
62        }
63        else
64        if(j==3){
65          Serial.print("OK");
66          iv += c;
67        }
68      }
69      i--;
70    }
71    int x = Wire.read();     // receive byte as an integer
72    Serial.println("");          // print the integer
73    Serial.println("CypherText");          // print the integer
74    Serial.println(cypherText);          // print the integer
75    Serial.println("IV");          // print the integer
76    Serial.println(iv);          // print the integer
77
78    ivI = iv.toInt();
79    Serial.println((long)ivI);          // print the integer
80    des.init("012345677654321001234567\0",ivI);
81
82
83    byte message[cypherText.length()+1];
84    cypherText.getBytes(message, cypherText.length()+1);
85    Serial.println((char*)message);
86    des.calc_size_n_pad(sizeof(message));
87    byte plaintext_p[des.get_size()];
88    des.tdesCbcDecipher(message,plaintext_p);
89    bool ok = des.CheckPad(plaintext_p,sizeof(plaintext_p));
90    if (ok)
91      printf("padding ok!");
92    else
93      printf("padding corrupted!");
94     Serial.println("Plain");
95     Serial.println((char*)plaintext_p);
96    cypherText = "";
97    iv ="";
```

```
98    j=0;
99  }
```

Listing 8.9: Maestro comunicación bidireccional

```
1  //Ivan Banos Delgado Mayo 27 2016
2
3  #include <SPI.h>
4  #include <Ethernet.h>
5  #include <Wire.h>
6
7  //Variables to receive event
8  int pos = 0;
9  byte MAC[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
10 byte IP[] = { 172,16,9,203};
11 byte gateway[] = { 192, 168, 1, 1 };
12 byte subnet[] = { 255, 255, 255, 1 };
13 EthernetServer server(80);
14 String readString;
15
16 //Variable to respond event
17 char serverToRespond[] = "ledswitch.botsolucionador.com";
18 IPAddress ip(192,168,0,101);
19 bool actionCourse;
20 bool first;
21 String eventS;
22     EthernetClient client;
23
24
25 void setup() {
26  // Open serial communications and wait for port to open:
27   Serial.begin(9600);
28    while (!Serial) {
29     ;// wait for serial port to connect. Needed for Leonardo only
30   }
31
32   //start the Ethernet connection and the server:
33
34   actionCourse =true;
35   first = true;
36   Wire.begin();
37 }
38
```

```
39 void loop() {
40   if(actionCourse){
41     if(first){
42     Ethernet.begin(MAC, IP, gateway, subnet);
43   server.begin();
44   Serial.print("server is at ");
45   Serial.println(Ethernet.localIP());
46   first=false;
47 }
48   client = server.available();
49
50   if (client) {
51     while (client.connected()) {
52       if (client.available()) {
53         char c = client.read();
54
55 //read char by char HTTP request
56         if (readString.length() < 100) {
57 //store characters to string
58           readString += c;
59 //Serial.print(c);
60         }
61
62 //if HTTP request has ended
63         if (c == '\n') {
64           client.println("HTTP/1.1 200 OK"); //send new page
65           client.println("Content-Type: text/html");
66           client.println();
67           client.println("<HTML>");
68           client.println("<HEAD>");
69           client.println("<TITLE>Semillero TADD</TITLE>");
70           client.println("</HEAD>");
71           client.println("<BODY>");
72           client.println("<H1><h3>Semillero TADD 172.16.9.203</h3></H1>
73           client.println("<hr />");
74           client.println("<br />");
75           client.println("<p>Created by Maryori Sabalza Mejia.
76           https://www.youtube.com/user/ButterflyDeath730/videos
77           for more projects and tutorials!</p>");
78           client.println("<br />");
79           client.println("</BODY>");
80           client.println("</HTML>");
```

```
81
82              delay(1);
83              //stopping client
84              client.stop();
85              if((readString.indexOf("Pi") >0 && readString.indexOf("Ac")>0
86              readString.indexOf("Ev") >0)){
87              //Parsing the string
88              //finding the pin
89            int posPin = readString.indexOf("Pi");
90            int inicV = readString.indexOf("=",posPin);
91            int finV = readString.indexOf(",",posPin);
92            String pinS = readString.substring (inicV+1,finV);
93            //Finding the action
94            int posAction = readString.indexOf("Ac");
95            int inicA = readString.indexOf("=",posAction);
96            int finA = readString.indexOf(",",posAction);
97            String actionS = readString.substring (inicA+1,finA);
98            //Finding the event
99            int posEvent = readString.indexOf("Ev");
100           int inicE = readString.indexOf("=",posEvent);
101           int finE = readString.indexOf(",",posEvent);
102           eventS = readString.substring (inicE+1,finE);
103
104           char ActionA[8];
105           actionS.toCharArray(ActionA,8);
106           int pinI = pinS.toInt();
107           Wire.beginTransmission(pinI); //transmit to slave
108           Wire.write(ActionA); //sends one action
109           Wire.endTransmission();
110           actionCourse = false;
111               }
112           //clearing string for next read
113               readString="";
114
115            }
116          }
117      }
118  }
119    }else{
120      if (Ethernet.begin(MAC) == 0) {
121      Serial.println("Failed to configure Ethernet using DHCP");
122      // no point in carrying on, so do nothing forevermore:
```

```
123     // try to congifure using IP address instead of DHCP:
124     Ethernet.begin(MAC, IP);
125   }
126   if (client.connect(serverToRespond, 80)) {
127     Serial.println("connected");
128     readString =  "Content-Length: 372";
129     String event =  "<idEvento>"+ eventS+"</idEvento>";
130     Serial.println(readString);
131     Serial.println(event);
132     client.println("POST /Services/DataHandler.asmx HTTP/1.1");
133     client.println("Host: ledswitch.botsolucionador.com");
134     client.println("Content-Type: application/soap+xml; charset=utf-8");
135     client.println(readString);
136     client.println();
137     client.println("<?xml version=\"1.0\" encoding=\"utf-8\"?>");
138     client.println("<soap12:Envelope xmlns:xsi=\"http://www.w3.org/2001/
139     client.println("<soap12:Body>");
140     client.println("<setEstadoEvento xmlns=\"http://tadd.org/\">");
141     client.println(event);
142     client.println("<estado>ok</estado>");
143     client.println("</setEstadoEvento>");
144     client.println("</soap12:Body>");
145     client.println("</soap12:Envelope>");
146     client.println();
147     readString = "";
148     while (client.connected()) {
149       if (client.available()) {
150         char c = client.read();
151 
152       if (readString.length() < 100) {
153           //store characters to string
154           readString += c;
155           Serial.print(c);
156          }
157          if (c == '\n') {
158 
159     readString = "Closing";
160     client.stop();
161      }
162       }
163     }
164     readString = "";
```

```
165       client.stop();
166     }
167     else {
168     //if you didn't get a connection to the server:
169       Serial.println("connection failed");
170     }
171     actionCourse = true;
172     first = true;
173     }
174 }
```

Listing 8.10: Código Esclavo

```
1  //Maryori Sabalza Mejia-Mayo 20 2016
2
3  #include <Wire.h>
4  int LED= 6;
5  void setup()
6  {
7    Wire.begin(6);                 //join i2c bus with address #4
8    Wire.onReceive(receiveEvent); //register event
9    Serial.begin(9600);           //start serial for output
10   pinMode(LED, OUTPUT);
11 }
12
13 void loop()
14 {
15   delay(100);
16 }
17
18 //function that executes whenever data is received from master
19 //this function is registered as an event, see setup()
20 void receiveEvent(int howMany)
21 {
22   while(1 < Wire.available()) // loop through all but the last
23   {
24     char c = Wire.read(); // receive byte as a character
25     Serial.print(c);         // print the character
26   }
27   char x = Wire.read();    // receive byte as an integer
28   Serial.println(x);         // print the integer
29   if(x) {
30   switch (x) { //select case de la variable x
```

```
31    case '1': //si es 1
32      Serial.println("Encendido");
33      digitalWrite(LED, HIGH); //enciende el led
34      break;
35    case '0':
36       Serial.println("Apagado");
37      digitalWrite(LED, LOW); //apaga el led
38      break;
39    }
40  }
41  }
```

Listing 8.11: Maestro solicita datos

```
1   //Ivan Banos Delgado junio 3 2016
2
3     Wire.requestFrom(2, 6, 28); // Sensor de Temperatura
4     String toma = "";
5     while (Wire.available())    //slave may send less than requested
6     {
7       char c = Wire.read();    //receive a byte as character
8       toma += c;
9     }
10    sendData("1", toma, "368");
11
12
13    Wire.requestFrom(3, 6, 28); // Sensor de movimiento
14    toma = "";
15    while (Wire.available())    //slave may send less than requested
16    {
17      char c = Wire.read();    //receive a byte as character
18      toma += c;
19    }
20    sendData("3", toma, "368");
21
22
23    Wire.requestFrom(4, 6, 28); //Sensor de Corriente
24    toma = "";
25    while (Wire.available())    //slave may send less than requested
26    {
27      char c = Wire.read();    //receive a byte as character
28      toma += c;
29    }
```

```
30    sendData("5", toma, "368");
31
32    Serial.println("");
33
34    delay(50000); // Cada 5 minutos se actualizan los datos
```

Listing 8.12: Esclavo: Sensor de Movimiento

```
1
2  // by Maryori sabalza 2 junio 2016
3
4  #include <Wire.h>
5
6  #define pinPIR 7
7  #define pinLED 12
8  #define MILISEGUNDOS 1000
9  int iCalibrar = 10;
10 boolean bMovimiento = true;
11
12 void LED(int iONOFF)
13 {
14   switch (iONOFF) {
15     case 0:
16       digitalWrite(pinLED, LOW);
17       delay(MILISEGUNDOS);
18       break;
19     case 1:
20       digitalWrite(pinLED, HIGH);
21       delay(MILISEGUNDOS);
22       break;
23   }
24 }
25
26 void setup()
27 {
28   Wire.begin(3);                // join i2c bus with address #4
29   Wire.onRequest(requestEvent); // register event
30
31   Serial.begin(9600);
32   pinMode(pinPIR, INPUT);
33   pinMode(pinLED, OUTPUT);
34   LED(0);//apagar
35   Serial.print(" Esperando al sensor ");
```

```
36    for (int i = 0; i < iCalibrar; i++) {
37      Serial.print(".");
38      delay(MILISEGUNDOS);
39    }
40    Serial.println("INICIADO");
41    delay(50);
42  }
43
44  void loop()
45  {
46    if (digitalRead(pinPIR) == HIGH) {
47      LED(1);//encender
48      bMovimiento = true;
49    }
50    else {
51      LED(0);//apagar
52      bMovimiento = false;
53    }
54    delay(500);
55
56  }
57  // function that executes whenever data is received from master
58  // this function is registered as an event, see setup()
59  void requestEvent()
60  {
61    Serial.println(bMovimiento);
62    if (bMovimiento) {
63      Wire.write("true  ");
64
65    } else {
66      Wire.write("false ");
67    }
68    delay(1000); // delay before loop
69  }
```

Listing 8.13: Esclavo: Sensor de Temperatura

```
1
2  // by Maryori sabalza 2 junio 2016
3
4  #include <Wire.h>
5
6  const unsigned int TEMP_SENSOR_PIN = 0;    // We mean analog 0
```

```
7  const float SUPPLY_VOLTAGE = 5.0;
   // If you use 3.3V, write it here.
8  const unsigned int BAUD_RATE = 9600;
9  float tempc=0;
10
11 const float get_temperature() {
12   const int sensor_voltage = analogRead(TEMP_SENSOR_PIN);
13   const float voltage = sensor_voltage * SUPPLY_VOLTAGE / 1024;
14   return (voltage * 100);
15 }
16
17 const char* floatToChar(float val){
18 char buff[10];
19   char valueString[100] = "";
20   //val = 0.0123;
21     dtostrf(val, 4, 2, buff);//4 is mininum width, 6 is precision
22     strcat(valueString, buff);
23     strcat(valueString, " ");
24
25   return valueString;
26 }
27
28 void setup()
29 {
30   Serial.begin(BAUD_RATE);
31   Wire.begin(2);                 // join i2c bus with address #4
32   Wire.onRequest(requestEvent); // register event
33
34 }
35
36 void loop()
37 {
38   tempc = get_temperature();
39   delay(500);
40
41 }
42 // function that executes whenever data is received from master
43 // this function is registered as an event, see setup()
44 void requestEvent()
45 {
46
47 const char *buf = floatToChar(tempc);//sprintf(buf, "%f", tempc);
```

```
48    Wire.write(buf);
49    Serial.println(buf);
50    tempc = 0;
51
52  delay(1000); // delay before loop
53  }
```

Listing 8.14: Modificacion en en Emonlib.cpp

```
1
2   int EnergyMonitor::getConsumo()
3   {
4       Serial.print(Vrms);
5       Serial.print("V");
6       Serial.print(' ');
7       Serial.print('\t');
8
9       Serial.print(Irms);
10      Serial.print("A");
11      Serial.print(' ');
12      Serial.print('\t');
13
14
15      Serial.print(apparentPower);
16      Serial.print("W");
17      Serial.print(' ');
18      Serial.print('\t');
19
20  //Consumo:(dias*((watts/1000))*horas) -- modificado el 06/05/2016
21      int consumo=0;
22      consumo=(30*((apparentPower/1000))*24);
23      Serial.print(consumo);
24      Serial.print("KW/h ");
25      Serial.print(' ');
26      Serial.print('\t');
27
28      delay(100);
29    return consumo;
```

Listing 8.15: Modificacion en en Emonlib.ino

```
1
2   float cons=0;//variable que envia dato al master
```

```
3
4  int inpin = A2;
5  int inpinc = A3;
6  int val = 0;
7
8  void setup()
9  {
10   Wire.begin(4);
11   Wire.onRequest(requestEvent);
12
13   Serial.println("Medicion de Potencia Electrica");
14 }
15 void loop()
16
17 {
18   int consumo = emon1.getConsumo();
19   Serial.println(consumo);
20   val = analogRead(inpin);
21   Serial.println(val);
22   cons=consumo;
23   delay(500);
24 }
25 ////////////////////////////////////////
26 void requestEvent()
27 {
28  const char *buf = floatToChar(cons);
29 //sprintf(buf, "%f", tempc);
30   Wire.write(buf);
31   Serial.println(buf);
32
33 cons= 0;
34
35 delay(1000); // delay before loop
36 }
```

Códigos en HTML

Listing 8.16: Código HTML implementado en XAMPP

```
1
2  <!--Start of the HTML-->
```

```
<html>
<head>
<title>Semillero TADD</title>
</head>
<body bgcolor="#ADD8E6">
<?php

// Check of LED2 is set.  If it is use it
if (isset($_POST["LED2"]))
{
$LED2= $_POST["LED2"];
//echo "<b>$LED2</b>";
}
else
{
$LED2 ="";
}
if ($LED2 == "ON")
{
// Set led2 ON by calling the Arduino using fopen
//ini_set("allow_url_fopen On", true);
$h = @fopen("http://192.168.1.177/?LED2=ON", "rb");
}
else if ($LED2 == "OFF")
{
// Set led2 OFF by calling the Arduino using fopen
//ini_set("allow_url_fopen On", true);
$h= @fopen("http://192.168.1.177/?LED2=OFF", "rb");
}


?>
<!-- LED2 FORM -->
<table>
<tr><td colspan="2"><font size="4" color="black">LED2</font></H4></td></
<tr><td>
<form action="led2.php" method="post">
<input type="hidden" name="LED2" value="ON">
<input type="submit" name="submit" value="ON">
</form>
</td><td>
<form action="led2.php" method="post">
```

```
45 <input type="hidden" name="LED2" value="OFF">
46 <input type="submit" name="submit" value="OFF">
47 </form>
48 </td></tr>
49 </table>
50 </body>
51 </html>
```