DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DOMÓTICO PARA LA SEGURIDAD DEL HOGAR CONTROLADO VÍA CENTRAL ASTERISK E INTERFAZ DE HARDWARE ARDUINO

David Alejandro Rueda Mosquera
Sandra Patricia Vargas Rueda

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS
FACULTAD TECNOLÓGICA

Bogotá D.C.
Agosto del 2015
DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DOMÓTICO PARA LA SEGURIDAD DEL HOGAR CONTROLADO VÍA CENTRAL ASTERISK E INTERFAZ DE HARDWARE ARDUINO

Código del Proyecto 201501273069

David Alejandro Rueda Mosquera - Código: 20132373055
Sandra Patricia Vargas Rueda - Código: 20131273009

Monografía para optar por el título de:

Ingeniero En Telecomunicaciones

Director:
Frank Nixon Giraldo Ramos

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

FACULTAD TECNOLÓGICA

Bogotá D.C.
Agosto del 2015
DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DOMÓTICO PARA LA SEGURIDAD
DEL HOGAR CONTROLADO VÍA CENTRAL ASTERISK E INTERFAZ DE
HARDWARE ARDUINO

PAGINA DE APROBACIÓN

Observaciones

________________________________________________________________________
________________________________________________________________________
________________________________________________________________________

Ingeniero Frank Nixon Giraldo Ramos
Director del Proyecto

________________________________________________________________________
Jurado 1

________________________________________________________________________
Jurado 2

Fecha de Presentación  Agosto  del 2015.
AGRADECIMIENTOS

Los autores expresan sus agradecimientos a:

La familia Vargas Rueda, y la familia Rueda Mosquera por estar siempre en esos momentos difíciles que la vida y una carrera profesional exigen, al amor, la comprensión y el incansable apoyo. A los compañeros, profesores y directivos que luchan día a día por llevar con orgullo el nombre de la Facultad Tecnológica de la Universidad Distrital.
# TABLA DE CONTENIDO

AGRADECIMIENTOS ............................................................................................................ 4
LISTA DE FIGURAS .................................................................................................................. 7
LISTA DE TABLAS ................................................................................................................... 8
LISTA DE ANEXOS ................................................................................................................... 8
GLOSARIO ............................................................................................................................. 9
RESUMEN ................................................................................................................................ 12
ABSTRACT ............................................................................................................................. 13
INTRODUCCIÓN .................................................................................................................... 14
CAPITULO 1 ............................................................................................................................. 15
  1.1 Descripción del problema .............................................................................................. 15
  1.2 Justificación .................................................................................................................... 15
    1.2.1 Impacto Social .......................................................................................................... 16
    1.2.2 Impacto Económico ................................................................................................. 16
    1.2.3 Impacto Tecnológico ............................................................................................... 17
  1.3 Metodología Usada ......................................................................................................... 17
  1.4 Alcance del Proyecto ...................................................................................................... 19
CAPITULO 2 ............................................................................................................................. 20
  2.1 Objetivo General ............................................................................................................. 20
  2.2 Objetivos Específicos .................................................................................................... 20
CAPITULO 3 ............................................................................................................................. 21
  3.1 Asterisk ........................................................................................................................... 21
    3.1.1 Historia del Asterisk ................................................................................................. 21
    3.1.2 AsteriskNOW. Definiciones y descripción funcional .................................................. 22
    3.1.3 Protocolos de VOIP ............................................................................................... 23
    3.1.4 AGI Asterisk Gateway Interface ............................................................................. 26
  3.2 Arduino y Ethernet Shield ............................................................................................... 28
    3.2.1 Estructura General de Un sketch ........................................................................... 29
  3.3 Softphone ......................................................................................................................... 30
<table>
<thead>
<tr>
<th>3.4 Estado del arte</th>
<th>31</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.4.1 Proyecto 1:</td>
<td>31</td>
</tr>
<tr>
<td>3.4.2 Proyecto 2:</td>
<td>32</td>
</tr>
<tr>
<td>3.4.3 Proyecto 3:</td>
<td>33</td>
</tr>
<tr>
<td>3.4.4 Proyecto 4:</td>
<td>33</td>
</tr>
<tr>
<td>3.4.5 Proyecto 5:</td>
<td>34</td>
</tr>
<tr>
<td>4.1 Hardware del Sistema</td>
<td>35</td>
</tr>
<tr>
<td>4.1.1 Sensor infrarrojo</td>
<td>35</td>
</tr>
<tr>
<td>4.1.2 Sensor magnético</td>
<td>36</td>
</tr>
<tr>
<td>4.1.3 Electroimán 12VDC</td>
<td>37</td>
</tr>
<tr>
<td>4.1.4 Arduino UNO y tarjeta Ethernet Shield</td>
<td>38</td>
</tr>
<tr>
<td>4.1.5 Telefono Android</td>
<td>40</td>
</tr>
<tr>
<td>4.1.6 Modem</td>
<td>40</td>
</tr>
<tr>
<td>4.1.7 PC, Servidor PBX Asterisk</td>
<td>40</td>
</tr>
<tr>
<td>4.2 Software del Sistema</td>
<td>41</td>
</tr>
<tr>
<td>4.2.1 Servidor PBX AsteriskNOW</td>
<td>41</td>
</tr>
<tr>
<td>4.2.2 Software Arduino</td>
<td>43</td>
</tr>
<tr>
<td>4.2.3 Software Zoiper</td>
<td>44</td>
</tr>
<tr>
<td>4.3 Descripción Funcional del Proyecto</td>
<td>45</td>
</tr>
<tr>
<td>4.4 Instalación de Asterisk</td>
<td>47</td>
</tr>
<tr>
<td>4.5 Configuración de Archivos Asterisk</td>
<td>55</td>
</tr>
<tr>
<td>4.5.1 Asignación de IP Estatica</td>
<td>56</td>
</tr>
<tr>
<td>4.5.2 Extensions_custom.conf</td>
<td>58</td>
</tr>
<tr>
<td>4.5.3 Implementación de Scripts 1. Conexión_Arduino.php</td>
<td>59</td>
</tr>
<tr>
<td>4.5.4 Implementación de Scripts 2. Arduino_call.php</td>
<td>60</td>
</tr>
<tr>
<td>5.1 Pruebas y Resultados del Sistema</td>
<td>62</td>
</tr>
<tr>
<td>5.1.1 Primera Prueba. Comunicación LAN</td>
<td>62</td>
</tr>
<tr>
<td>5.1.2 Segunda Prueba, Conexión desde Asterisk a Arduino</td>
<td>65</td>
</tr>
<tr>
<td>5.1.3 Tercer Prueba, Enlace desde Arduino a Asterisk</td>
<td>67</td>
</tr>
<tr>
<td>5.1.4 Cuarta Prueba, Conexión Arduino – Asterisk – Arduino remoto WAN</td>
<td>68</td>
</tr>
</tbody>
</table>
CONCLUSIONES .................................................................................................................. 71
BIBLIOGRAFÍA ....................................................................................................................... 72
Anexo A. Código Arduino ........................................................................................................ 74

LISTA DE FIGURAS

Ilustración 1. Arquitectura de protocolos SIP ........................................................................ 24
Ilustración 2 Softphone Zoiper .......................................................................................... 30
Ilustración 3 Sensor Infrarrojo PIR IS2560T ....................................................................... 36
Ilustración 4 Sensor Magnético 7939WG-WH .................................................................. 36
Ilustración 5 Electroimán 12VDC ...................................................................................... 37
Ilustración 6 Arduino Uno (Izquierda), Ethernet Shield (Derecha). ................................. 39
Ilustración 7 Circuito usado para la conexión y adaptación de los sensores al Arduino. .... 39
Ilustración 8 Hardware del Sistema .................................................................................. 41
Ilustración 9 Software Asterisk ........................................................................................ 43
Ilustración 10 Diagrama de Flujo Programa Arduino ....................................................... 44
Ilustración 11 Configuración Zoiper Comunicación local ................................................. 45
Ilustración 12 Descripción Funcional del Sistema .............................................................. 47
Ilustración 13 Instalación Asterisk A .................................................................................. 48
Ilustración 14 Instalación Asterisk B .................................................................................. 49
Ilustración 15 Instalación Asterisk C .................................................................................. 50
Ilustración 16 Instalación Asterisk D .................................................................................. 51
Ilustración 17 Imagen Punto ISO ....................................................................................... 52
Ilustración 18 Restaurar por defecto la configuración de Red ........................................... 52
Ilustración 19 Configuración Asterisk ............................................................................. 53
Ilustración 20 Parámetros básicos. .................................................................................... 53
Ilustración 21 Asignación de Claves. ................................................................................ 54
Ilustración 22 Interfaz de Usuario AsteriskNow ............................................................... 54
Ilustración 23 Directorios y subdirectorios usados por Asterisk ........................................ 55
Ilustración 24 Ifconfig, Configuración de Red en Asterisk ............................................. 56
Ilustración 25 Asignación de IP estática a nuestro servidor PBX Asterisk ....................... 57
Ilustración 26 Service Network Restart ............................................................................. 57
Ilustración 27 Extensions_custom.conf .......................................................................... 58
Ilustración 28 AGI: Conexión_Arduino.php .................................................................... 59
Ilustración 29 Configuración para la recepción de una llamada hecha por Arduino a Asterisk. Arduino_call.php ..... 60
Ilustración 30 Implementación Scripts 1 y 2 .................................................................... 61
Ilustración 31 Administración FreePBX. ............................................................................ 62
Ilustración 32 Interfaz de línea de Comando CLI ................................................................. 63
Ilustración 33 Estadísticas FreePBX ..................................................................................... 64
Ilustración 34 Interfaz de línea de Comandos durante la llamada ....................................... 64
Ilustración 35 Ping a la placa Arduino .................................................................................. 65
Ilustración 36 Carpeta Agi-bin ............................................................................................ 66
Ilustración 37 Comunicación del Arduino al Asterisk, Activación Sensor magnético ........ 67
Ilustración 38 Configuración NO-IP ................................................................................... 68
Ilustración 39 Configuración NAT para tener una conexión WAN y no solo LAN .................. 69
Ilustración 40 Configuración Zoiper Android ....................................................................... 69
Ilustración 41 Puertos RTP ................................................................................................. 70

LISTA DE TABLAS

Tabla 1 Características del Protocolo SIP .............................................................................. 24
Tabla 2 Potencias requeridas por los elementos .................................................................... 37
Tabla 3 Máquinas Virtuales .................................................................................................. 42
Tabla 3 Descripción de directorios usados por Asterisk ....................................................... 56
Tabla 5 Configuración de puertos en el Router ..................................................................... 70

LISTA DE ANEXOS

Anexo A. Código Arduino
GLOSARIO

AGI  Interface de Entrada de Asterisk
ADSL  Línea de Abonado Digital Asimétrica.
AMI  Interface de Administración de Asterisk
AMPS  Sistema Telefónico Movil Avanzado.
AP  Punto de acceso inalámbrico.
ARP  Protocolo de Resolucion de Direcciones.
AU  Unidad de Acceso.
Backhaul  Red retorno.
Broadcast  Difusion.
BU  Unidad Base.
BW  Ancho de Banda.
CDR  Registro de llamadas
CEL  Registro de eventos de Canal.
CPE  Equipo Local del Cliente.
CRT  Comité de Regulación de Telecomunicaciones.
DSCP  Punto de Código para Servicios Diferenciados.
ESSID  Extended Service Ser IDentifier.
ETD  Equipo terminal de datos.
FCC  Comité de Comunicaciones Federal.
FEC  Correcion de Errores hacia Adelante.
FXO  Oficina de intercambio externo
FXS  Estación de intercambio externo
GPS  Sistema de Posición Global.
GSM  Sistema Global para comunicaciones Moviles.
GW  Puerta de Enlace.
<table>
<thead>
<tr>
<th>Abreviatura</th>
<th>Definición</th>
</tr>
</thead>
<tbody>
<tr>
<td>IAX</td>
<td>Intercambio entre servidores Asterisk</td>
</tr>
<tr>
<td>IAP</td>
<td>Punto de Acceso Inteligente.</td>
</tr>
<tr>
<td>IEEE</td>
<td>Instituto de Ingeniería de Eléctricos y Electrónicos.</td>
</tr>
<tr>
<td>IPS</td>
<td>Sistema de Proteccion contra Intrusiones.</td>
</tr>
<tr>
<td>IS-136</td>
<td>Estadar intermedio de comunicaciones mobiles.</td>
</tr>
<tr>
<td>ISDN</td>
<td>Red digital de servicios Integrados.</td>
</tr>
<tr>
<td>IVR</td>
<td>Respuesta de voz interactiva</td>
</tr>
<tr>
<td>LED</td>
<td>Diodo Emisor de Luz.</td>
</tr>
<tr>
<td>LOS</td>
<td>Linea de vista.</td>
</tr>
<tr>
<td>MAC</td>
<td>Control de Acceso al Medio.</td>
</tr>
<tr>
<td>Mbps</td>
<td>Megabit por segundo.</td>
</tr>
<tr>
<td>MESH</td>
<td>Malla.</td>
</tr>
<tr>
<td>MIMO</td>
<td>Multiple Entrada – Multipla Salida.</td>
</tr>
<tr>
<td>MHz</td>
<td>Megahertz.</td>
</tr>
<tr>
<td>ms</td>
<td>Milisegundo.</td>
</tr>
<tr>
<td>MWR</td>
<td>Router Inalambico para redes Mesh</td>
</tr>
<tr>
<td>NAT</td>
<td>Traduccion de direcciones de Red</td>
</tr>
<tr>
<td>NLOS</td>
<td>Sin linea de vista.</td>
</tr>
<tr>
<td>OFDMA</td>
<td>Acceso Multiple por Division de Frecuencia Ortogonal.</td>
</tr>
<tr>
<td>OFDM</td>
<td>Multipleacion por Division de Frecuencias Ortogonales.</td>
</tr>
<tr>
<td>OS</td>
<td>Sistema Operativo.</td>
</tr>
<tr>
<td>OSS</td>
<td>Sistema de sonido Abierto.</td>
</tr>
<tr>
<td>PBX</td>
<td>Ramal privado de conmutación automática</td>
</tr>
<tr>
<td>PCI</td>
<td>Interconexion de Componentes Perifericos.</td>
</tr>
<tr>
<td>PCMCIA</td>
<td>Personal Computer Memory Card International Association.</td>
</tr>
<tr>
<td>PDA</td>
<td>Asistente Digital Personal.</td>
</tr>
<tr>
<td>PDC</td>
<td>Controlador de Dominio Primario.</td>
</tr>
<tr>
<td>Acrónimo</td>
<td>definición</td>
</tr>
<tr>
<td>----------</td>
<td>------------</td>
</tr>
<tr>
<td>PmP</td>
<td>Punto – multipunto.</td>
</tr>
<tr>
<td>POTS</td>
<td>Servicio Telefónico Tradicional</td>
</tr>
<tr>
<td>PSTN</td>
<td>Red Telefónica Pública conmutada</td>
</tr>
<tr>
<td>PtP</td>
<td>Punto a Punto.</td>
</tr>
<tr>
<td>QoS</td>
<td>Calidad de Servicio.</td>
</tr>
<tr>
<td>Reflexión</td>
<td>Cambio de dirección de una onda regresando al medio del que venía.</td>
</tr>
<tr>
<td>RB</td>
<td>Punto Remoto.</td>
</tr>
<tr>
<td>RF</td>
<td>Radio Frecuencia.</td>
</tr>
<tr>
<td>RTCP</td>
<td>Protocolo de control de Tiempo Real.</td>
</tr>
<tr>
<td>RTP</td>
<td>Protocolo de transporte de Tiempo Real.</td>
</tr>
<tr>
<td>RTPC</td>
<td>Red de telefonía Pública Conmutada</td>
</tr>
<tr>
<td>SIP</td>
<td>Protocolo de Inicio de Sesión.</td>
</tr>
<tr>
<td>SMS</td>
<td>Servicio de Mensajería Instantánea.</td>
</tr>
<tr>
<td>SSID</td>
<td>Servicio de Establecimiento de Identificador de Red.</td>
</tr>
<tr>
<td>SU</td>
<td>Unidad Suscriptora.</td>
</tr>
<tr>
<td>TCP</td>
<td>Protocolo de Control de Transmisión.</td>
</tr>
<tr>
<td>TDMA</td>
<td>Acceso Multiple por División de Tiempo.</td>
</tr>
<tr>
<td>TIC</td>
<td>Tecnología de Información y Comunicación.</td>
</tr>
<tr>
<td>ToS</td>
<td>Tipo de Servicio.</td>
</tr>
<tr>
<td>Tx</td>
<td>Transmisor.</td>
</tr>
<tr>
<td>USB</td>
<td>Bus Serial Universal.</td>
</tr>
<tr>
<td>VLAN</td>
<td>Red de Área Local Virtual</td>
</tr>
<tr>
<td>VoIP</td>
<td>Voz sobre IP.</td>
</tr>
<tr>
<td>WDS</td>
<td>Sistema de Distribución Inalámbrico.</td>
</tr>
<tr>
<td>WiFi</td>
<td>Red Inalámbrica a Corto Alcance.</td>
</tr>
<tr>
<td>WLAN</td>
<td>Red Inalámbrica de Área Local.</td>
</tr>
<tr>
<td>WMAN</td>
<td>Red Inalámbrica de Área Metropolitana.</td>
</tr>
</tbody>
</table>
RESUMEN

En este proyecto se realiza el diseño y la implementación de un sistema domótico apoyado en las tecnologías libres de Asterisk y Arduino, para el desarrollo de una alternativa en la seguridad del hogar, y que servirá como base para la búsqueda de soluciones tecnológicas a partir de plataformas de libre acceso y desarrollo; brindando alternativas para la comunidad académica.

Se buscan optimizar de esta forma las variables de tiempo, costo, y beneficio que pueden brindar a la comunidad en general la implementación de los sistemas domóticos, aprovechando los importantes recursos que brindan las redes de datos y las telecomunicaciones en general para realizar acciones de forma remota sobre las diferentes tecnologías utilizadas en la era actual.

Para ello se contextualiza el desarrollo del proyecto en la capacidad y fundamentación de los sistemas de VoIP, y aplicando los conceptos adquiridos durante los últimos años en protocolos de comunicaciones, programación de microcontroladores y diseños electrónicos con actuadores y sensores. Punto de partida teórico práctico, para brindar una solución de bajo costo a la seguridad del hogar de forma remota y que genera acciones en pro de la conservación de bienes de la comunidad.
ABSTRACT

In this project the design and implementation of automation system based on free technologies is made: Asterisk and Arduino, for the development of an alternative home security, and will serve as a basis for finding technological solutions from Free access platforms and development; providing alternatives to the academic community.

They thus seek to optimize the variables of time, cost, and benefit the community can provide overall implementation of automation systems, using significant resources that provide data networks and telecommunications in general to perform actions remote on the different technologies used in the present age.

To do the project on the basis of capacity and VoIP systems is contextualized, and applying the knowledge acquired over the years in communications protocols, programming of microcontrollers and electronic designs with actuators and sensors. Theoretical and practical starting point, to provide a low-cost home security remotely and generating actions for the conservation of community assets.
INTRODUCCIÓN

Tener el control de las funciones de sistemas externos de forma remota, genera una gran cantidad de ideas, estrategias, soluciones y procesos para el desarrollo de la sociedad, por otro lado Colombia es lastimosamente un País en el cual los índices de robo a viviendas incrementa de forma sustancial año tras año, y es un hecho que la tecnología puede servir como acción complementaria para garantizar que los hogares cuenten con zonas seguras.

La seguridad en el hogar es primordial para el ciudadano que pasa gran parte de las horas diarias por fuera de su vivienda. Por lo tanto se hace necesario la creación de un sistema, que permita monitorear remotamente el hogar, que no sólo actúe de forma pasiva, sino que implemente acciones en pro de la seguridad, y por último que aporte en el desarrollo de integraciones entre plataformas de hardware y software libres como Asterisk y Arduino.

Este proyecto se dirige a personas sin mucho poder adquisitivo ya que está basado sobre la plataforma Asterisk, la cual funciona con licencias de código abierto, además de marcar un punto de partida en los sistemas de seguridad de bajo costo, ya que en general son simplemente sistemas de percepción e información efectiva, y que no realizan acción sobre los hechos de inseguridad que siempre han sido un mal de la sociedad a través de la historia.
1.1 Descripción del problema

Este proyecto busca la creación de un puente entre las personas que desean proteger su patrimonio en el hogar y los costos de los sistemas electrónicos de vigilancia remota. En el último año, más de 600 casas fueron víctimas de hurto cada día en Colombia. En la mayoría de casos, los delincuentes ingresaron a las residencias forzando las cerraduras, la puerta o las ventanas, cualquiera de estas acciones, generan movimientos físicos del delincuente. Como se mencionó anteriormente los propietarios de las viviendas pasan gran cantidad de tiempo fuera del hogar, y no poseen herramientas tecnológicas en donde puedan monitorearlo, muchas veces deben acudir a costosos sistemas de vigilancia que generan mensualidades altas para el bolsillo de una persona promedio en Colombia. La comunicación debe ser efectiva al momento de informar estos eventos, por ello es necesario que se cree no sólo un medio de comunicación efectivo sino también una plataforma de interfaz con el usuario que sea rápida y amigable.

1.2 Justificación

Las plataformas libres de Arduino y Asterisk poseen un potencial muy grande cuando se combinan, ya que permiten brindar soluciones tecnológicas eficientes, así que para la aplicación específica de la seguridad en el hogar funciona correctamente, ya que logra enviar información en tiempo real del estado de dos sensores al Softphone que funciona bajo el servidor virtual PBX, permitiendo al usuario realizar una acción sobre un actuador en pro de la seguridad del hogar.
El entorno de desarrollo de Arduino por su parte, permite crear una lógica sencilla sobre los sensores y actuadores, y no solo recibir instrucciones de parte del servidor de Asterisk, sino entregarle información a este para que el usuario interactúe, eliminando la brecha que muchas veces existe en los sistemas de seguridad electrónica convencionales en los que el usuario simplemente recibe información sobre el estado de los sensores, y debe actuar de forma presencial para realizar reacciones oportunas.

1.2.1 Impacto Social

Los hogares que implementen el sistema domótico con Arduino y Asterisk, tendrán la oportunidad de monitorear en tiempo real acciones como la apertura de una puerta, y realizar la activación de un bloqueo o electroimán que impide la huida del posible delincuente. Teniendo acceso a esta tecnología por medio de un software (Softphone) donde recibe información del estado de la puerta(s) del lugar y la posibilidad de enviar alertas.

1.2.2 Impacto Económico

Como se mencionó anteriormente, los sistemas de seguridad electrónica en general son de altos costos, y el monitoreo de alarmas es un servicio que los usuarios pagan mensualmente, con la implementación del sistema domótico con Arduino y asterisk, estos costos serán reducidos en gran medida, debido a que se desarrollan sobre plataformas libres y es el usuario final quien mantiene el monitoreo sobre su propio sistema.
1.2.3 Impacto Tecnológico

En Colombia varias organizaciones desarrollan sistemas sobre Asterisk, aunque específicamente, no se han presentado innovaciones de estos sistemas integrados a sistemas domóticos controlados con Arduino, que se presenten formalmente como desarrollos prácticos.

La implementación del sistema domótico para la seguridad con Arduino y Asterisk, va a marcar un punto de partida para el incentivo del desarrollo domótico controlado desde Softphones, y otras soluciones que puedan generar eficiencia combinando las telecomunicaciones y los sistemas de Control, desde la plataforma de Asterisk y de Arduino.

1.3 Metodología Usada

A continuación se describe la metodología usada, la cual se divide en 6 partes:

- En primer lugar, se realizó la búsqueda de una tecnología que permitiera la convergencia entre un software y un hardware libres, y que permitan al usuario final realizar procesos, brindar soluciones, o ejecutar acciones físicas de forma remota sobre un sistema convencional. Es allí donde se encuentra que Arduino y Asterisk son esos dos pilares que se logran complementar para cumplir con lo anteriormente mencionado.

Ahora la búsqueda de la acción a realizar no es muy compleja, debido a que en Colombia se evidencia la falta de seguridad dentro de los hogares, la gran cantidad de hurtos a viviendas en horas en que los propietarios están por fuera, y el poder adquisitivo que se debe tener para adquirir un sistema de seguridad monitoreado el cual no garantiza finalmente que no ocurran hurtos a las viviendas. Por lo tanto la acción que realiza el sistema es monitorear dos sensores ubicados dentro del hogar que supervisan el ingreso de un delincuente y brindarle a un usuario
ubicado remotamente la posibilidad de accionar un actuador que también estará ubicado en el hogar, implementando una lógica sencilla.

- Luego se tiene la instalación, configuración y ejecución de Asterisk, teniendo en cuenta los requisitos mínimos para su buen funcionamiento, puesto que se debe tener cuidado con el tipo de sistema operativo donde va a correr el sistema, y la versión del software. Así mismo se realiza una consulta efectiva sobre el funcionamiento de un Softphone (Aplicativo para el usuario final), protocolos que se manejan, y configuración básica necesaria.

- Se realiza la configuración, implementación y enlace entre el Softphone y el servidor virtual PBX, para garantizar una efectiva comunicación desde este punto en adelante, realizando direccionamiento IP dentro del PBX y brindando pautas como el protocolo SIP, y con esto realizando pruebas pertinentes para establecer las bases de la comunicación.

- Implementación del hardware Arduino, realizando el diseño del sistema electrónico, y con ello la lógica del microcontrolador para la configuración de puertos de salidas y entradas, así como también buscando la mejor solución para la alimentación del hardware. Además de ello, se cargan las correspondientes librerías sobre el compilador de Arduino para realizar una comunicación por medio de TCP/IP con Asterisk.

- Con la lógica y la estructura de Arduino funcionando localmente, se realiza la conexión en red de los dos pilares Asterisk y Arduino, que convergen finalmente en el servidor PBX para enviar y recibir instrucciones en una comunicación bidireccional.

- Luego se tiene la realización de pruebas en red, en un entorno real, y con ello la presentación de resultados y de conclusiones del proyecto, el cual también consta de una interfaz al usuario.
con el Softphone, y el correcto funcionamiento de este no solo desde un ordenador convencional, sino la conexión del sistema a una aplicación móvil para Android.

1.4 Alcance del Proyecto

- Obtener un sistema domótico que integre las plataformas de hardware y software libres de Arduino y Asterisk, logrando una comunicación bidireccional en una red local y a través de Internet.

- Establecer una lógica funcional para el comportamiento de los sensores y el actuador, el Softphone y el servidor, para lograr una buena percepción y confiabilidad del usuario final.

- Generar nuevas expectativas de desarrollo de Arduino y Asterisk dentro de la comunidad académica, para brindar soluciones a la sociedad Colombiana.

- Brindar a las personas que tienen la necesidad de implementar sistemas de seguridad electrónica de bajo costo una alternativa a los sistemas convencionales tanto en la parte electrónica como en la comunicación y reporte del sistema integrado.
CAPITULO 2
OBJETIVOS

2.1 Objetivo General

Diseñar e implementar un sistema domótico para la seguridad del hogar controlado vía central Asterisk e interfaz de hardware Arduino. Presentando una lógica efectiva en la parte electrónica y realizando una comunicación bidireccional entre las dos plataformas.

2.2 Objetivos Específicos

- Implementar un servidor virtual para una central Asterisk en un PC de control del sistema.

- Desarrollar un script en Asterisk y el firmware en Arduino para intercambio de datos con la central Asterisk.

- Implementar el sistema domótico y la interfaz de hardware, sensores y actuadores requeridos para la plataforma Arduino.

- Configurar la central Asterisk para el control del sistema domótico de seguridad desde Softphones.
CAPITULO 3  
MARCO TEÓRICO Y ESTADO DEL ARTE

3.1 Asterisk

3.1.1 Historia del Asterisk

Las primeras centrales telefónicas se crearon para interconectar un conjunto de líneas telefónicas de un punto hacia otro y la conmutación de una línea a otra se realizaba manualmente. Con el desarrollo de la tecnología y la telefonía se consiguió automatizar este sistema, de manera que la conmutación se realizara automáticamente y no se necesitaba tener un gran número de operadores realizando este trabajo. Hoy en día, tenemos la red de telefonía pública conmutada o RTPC que es un conjunto de centrales telefónicas analógicas conectadas entre sí y a través de la cual la conmutación de circuitos permite interconectar las líneas telefónicas de los usuarios. Esto facilitó el crecimiento de la red.
A medida que pasaban los años, el teléfono se convirtió en algo muy necesario y pasó a ser considerado como un servicio básico. Entonces se crearon las centrales telefónicas secundarias automáticas o PBXs. Mediante estas centrales las empresas se conectaban a la PSTN y lograban tener una mejor administración y control de sus teléfonos internos asignando un número de n dígitos para cada teléfono de la empresa denominado extensión. Empresas como Panasonic, Samsung, Nortel, entre otras, empezaron a fabricar y vender este tipo de centrales.

Luego de varios años, con el desarrollo de las redes de datos surgió la transmisión de voz sobre el protocolo IP. Esto trajo consigo el desarrollo de centralitas telefónicas para VoIP que en ciertos casos han pasado a remplazar a las centrales analógicas debido a que actualmente las empresas utilizan enlaces de datos para comunicarse entre sucursales y estos enlaces pueden ser aprovechados no solo para transmisión de datos sino también para voz o video. Actualmente existen varios protocolos de VoIP como SIP, IAX, H.323, MGCP o SCCP.
Asterisk fue desarrollada por Mark Spencer utilizando lenguaje C debido a la necesidad de adquirir una central telefónica de bajo costo para su empresa inicial. Spencer en ese entonces era estudiante de ingeniería y había desarrollado previamente proyectos como el cliente de chat GAIM que actualmente se lo conoce como PIDGIN y además de este, otros proyectos de software libre. Este joven emprendedor inició lo que hoy se conoce como Asterisk. (Long, J & Chaffin Larry, 2007).

Spencer junto con otros programadores que contribuyeron a la corrección de errores y adición de ciertas funciones desarrollaron la PBX Asterisk inicialmente para el sistema operativo GNU/Linux. Sin embargo las versiones actuales son compatibles con los sistemas operativos BSD, MAC OS X, Solaris, Microsoft Windows. Aunque para la plataforma Linux que es la nativa existe un mejor soporte que para el resto de plataformas. La compañía que desarrolla hardware y software para Asterisk se llama Digium.

3.1.2 AsteriskNOW. Definiciones y descripción funcional.

Asterisk es un software open source, convirtiéndose en una plataforma para crear una cantidad infinita de aplicaciones basadas no solamente en voz, sino en la unión con datos y/o cualquier otro sistema de cómputo que se necesite que interactúe con un teléfono.

AsteriskNOW, es la distribución oficial de Digium, está diseñado para desarrolladores de aplicaciones, integradores de sistemas, estudiantes, hackers y todo aquel que desee crear soluciones a medida con Asterisk. Algo que la caracteriza es que esta, es la distribución más ligera de todas con lo que no se instalan extras como HUd en Trixbx, vtiger e Hylafax en Elastix. Este software funciona como una PBX, un ramal privado de conmutación automática. En otras palabras, se puede describir a Asterisk como una central telefónica secundaria privada que se conecta a la red de telefonía pública través de líneas troncales para gestionar llamadas internas, llamadas entrantes y salientes con autonomía sobre cualquier otra central telefónica. (Madsen, Van & Bryant, 2011). Está basado en el protocolo IP permitiendo instalar extensiones en sitios remotos fuera de la red interna.
donde alberga el servidor PBX-IP, logrando un intercambio de información entre estas que por medio de un canal de comunicación desde y hacia la red IP donde se encuentra instalado el servidor Asterisk, así como desde y hacia las diferentes localidades donde estén instalados los teléfonos IP, para que se puedan comunicar entre ellos. Esto convierte a software de código abierto en una PBX muy versátil y con muchas ventajas sobre otra central telefónica.

3.1.3 Protocolos de VOIP

Es importante mencionar que intervienen una serie de reglas o protocolos para el establecimiento de una llamada entre dos o más teléfonos IP, de manera nativa se utilizan los protocolos SIP e IAX2. Estos a su vez viajan encapsulados en paquetes TCP/UDP sobre redes IP.

Protocolo SIP.

SIP o Session Initiation Protocol, en español protocolo de inicio de sesiones, el cual define la iniciación, modificación y finalización de sesiones interactivas de usuario donde intervienen elementos multimedia como el video, voz, mensajería instantánea, entre otros. Este protocolo se encarga de la señalización de llamada, Se complementa con SDP (Session Description Protocol) y RTP (Real Time Protocol), SDP para él enviado de los detalles del contenido multimedia de la sesión, como por ejemplo direcciones IP, puertos, y codec que se usaran durante la comunicación, y RTP para la transmisión de los datos, ya sean voz sobre IP, video u otros multimedia, entre los participantes de la comunicación, que previamente se estableció por SIP, siendo actualmente este protocolo el más utilizado para control de transporte de media. A nivel de capa 4, SIP utiliza por defecto el puerto 5060 en TCP y UDP. (Gómez, J; Gil F, 2009).

El protocolo SIP posee cuatro características que lo hacen muy recomendable para cumplir esta función, estas se pueden visualizar en la Tabla 1 que se muestra a continuación:
<table>
<thead>
<tr>
<th>Característica</th>
<th>Descripción.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Localización del usuario</td>
<td>SIP posee la capacidad de poder conocer en todo momento la localización de los usuarios. De esta manera no importa en qué lugar se encuentre un determinado usuario. En definitiva la movilidad de los usuarios no se ve limitada.</td>
</tr>
<tr>
<td>Negociación de los parámetros</td>
<td>Posibilidad de negociar los parámetros necesarios para la comunicación: puertos para el tráfico SIP, así como el tráfico Media, direcciones IP para el tráfico Media, códdec, etc.</td>
</tr>
<tr>
<td>Disponibilidad del usuario</td>
<td>SIP permite determinar si un determinado usuario está disponible o no para establecer una comunicación.</td>
</tr>
<tr>
<td>Gestión de la comunicación.</td>
<td>Permite la modificación, transferencia, finalización de la sesión activa. Además informa del estado de la comunicación que se encuentra en progreso.</td>
</tr>
</tbody>
</table>

Tabla 1 Características del Protocolo SIP

La ilustración 2, muestra la arquitectura de protocolos SIP:

![Ilustración 1. Arquitectura de protocolos SIP.](attachment:image.png)

Protocolo H323.

Es muy importante mencionar que este protocolo fue diseñado para ser un estándar en la transmisión de audio y datos a través de las redes IP en las cuales no existe garantía en la calidad del servicio, y el transporte de audio y video no es confiable.

La señalización de H.323 es muy rápida sobre todo si se comparan con la de SIP, la cual utiliza paquetes de gran tamaño. Esto es debido a que el formato de los mensajes en H.323 es binario, mientras que en los mensajes SIP el formato es texto plano.

A pesar de estar muy extendido, actualmente el auge de H.323 está descendiendo, ya que está siendo sustituido por SIP, el cual es modular y por tanto mucho más flexible.

**Protocolo IAX.**

IAX (Inter-Asterisk Exchange Protocol), es uno de los protocolos usados por Asterisk, es también un protocolo de señalización. La principal diferencia entre IAX y SIP o H.323 es que IAX no utiliza RTP, sino que en su lugar implementa su propio mecanismo de transmisión de voz.

IAX es mucho más compacto que los dos anteriores ya que ha sido diseñado únicamente para aplicaciones telefónicas, a diferencia de H.323 y sobre todo de SIP, que pueden utilizarse en otros tipos de tráfico media. IAX trabaja junto a UDP con una característica muy especial: todas las comunicaciones (registro, señalización de llamada, transmisión de voz) hacen uso de un único puerto UDP. Por lo tanto el NAT no supone un problema en IAX a diferencia de SIP, ya que tanto los datos de señalización como el audio viajan por el mismo puerto. Un inconveniente de IAX es que no es un estándar, sino un protocolo independiente creado por Mark Spencer, creador de Digium. En este proyecto se trabaja bajo el protocolo SIP, ya que es un estándar por la IETF(Internet Engineering Task Force, en español Grupo de Trabajo de Ingeniería de Internet ) y es soportado por la mayoría de equipos.

**Protocolos de Audio.** En este punto se revisan las reglas utilizadas para el transporte de audio y video en tiempo real, ya que en los protocolos nombrados anteriormente se tiene presente únicamente la disposición para establecer una comunicación entre dos extremos, quedando pendiente la
comunicación media. Para establecer un flujo de comunicación media es necesario un protocolo que intercambie información entre los extremos de dicha comunicación, es decir, que transporte la información entre un origen y su destino, con las técnicas necesarias para evitar problemas que puedan presentarse tales como retardos, perdida de audio, etc.

Los protocolos más usados para esta finalidad son:

**Real Time Protocol (RTP)**

**Real Time Control Protocol (RTCP)**

RTP se encuentra definido en el RFC3550 y es el encargado de transportar tanto audio como vídeo en tiempo real. Utiliza UDP como protocolo de transporte, ya que el uso de TCP y su control de flujo y congestión darían lugar a un retardo elevado durante la comunicación a causa de las retransmisiones. El protocolo RTP, para llevar a cabo su función, hace uso de un número de secuencia, marcas de tiempo, envío de paquetes sin retransmisión, identificación del origen, identificación del contenido, sincronización, etc., lo que le permite en presencia de pérdidas, Jitter o retardo poder continuar con la reproducción del flujo de paquetes. Por lo tanto no puede garantizar que la entrega de tráfico se haga en tiempo real, aunque sí garantiza al menos que lo hará de forma sincronizada.

SDP o Session Description Protocol, definido en el RFC4566, se utiliza durante la negociación que lleva a cabo SIP entre los dos agentes. Su función es la de detallar cómo se va a realizar el intercambio de comunicación posterior mediante protocolos como, por ejemplo, RTP. Para ello indica toda la información relacionada con el tráfico Media tal como IP y puerto donde espera cada agente recibir el audio, el codec a utilizar entre ambos, etc. (Gómez, J; Gil F, 2009).

### 3.1.4 AGI Asterisk Gateway Interface

AGI, Interface de entrada de Asterisk, es una interface que permite a Asterisk comunicarse con sistemas externos, ejecutando programas o scripts, en el momento que la aplicación AGI es invocada
desde el plan de marcado en el dialplan, precisamente en el fichero extensions_custom.conf. Estos scripts son desarrollados en varios lenguajes de programación entre ellos los más utilizados son: Python, Java, C, PHP, entre otros. Para este proyecto el usado es PHP.

Para el funcionamiento del AGI primero el script es llamado de acuerdo a la marcación de una extensión, que dependiendo de la aplicación a ejecutar establece relaciones externas del PBX, ejemplo con plataformas Arduino para el envío y recepción de datos, marcación automática a extensiones entre otras. Es conclusión es aquí donde se amplía la funcionalidad del PBX con muchos lenguajes de programación.

El encabezado de un script AGI, en PHP sus primeras dos líneas siempre debe llevar lo siguiente:

```
#!usr/bin/php –q
<?php
```

La opción –q desactiva los mensajes de error HTML. Los nombres de las variables tienen el prefijo "agi_" y se separan de sus valores de dos puntos y un espacio.

A continuación se nombran los comandos AGI usados:

**Answer**: Responde la llamada entrante.

**GetData**: Recibe los tonos DTMF marcados por el llamante.

**Receive Text**: Recibe texto de canales que lo soportan.

**Receive Char**: Recibe un carácter del canal usado.

**Hangup**: Subrutina que se ejecuta cuando cuelga el canal que se está usando.

**Wait for digit**: Espera que se presione un digito.

Entre otros que se nombrarán en el siguiente capítulo donde se dará la explicación paso a paso de la implementación del sistema. Por el momento se dará paso a la teoría sobre el otro componente pilar
de este proyecto al cual se le define como Arduino que es el hardware que ha de interconectarse con nuestro PBX AsteriskNOW. (Bryant B, Madsen L, Maggelen J, 2013).

3.2 Arduino y Ethernet Shield.

Arduino.

Arduino ha sido diseñado con el propósito de facilitar la electrónica en el desarrollo e implementación de proyectos. Es un sistema programable con un micro-controlador, de bajo costo, fácil de aprender, utilizar, y con un lenguaje de programación sencillo. Trabaja básicamente mediante la recepción de datos a través de sus pines, que pueden estar conectados a diversos sensores y/o actuadores. Su idea nace en el año 2005 como proyecto universitario en Italia, en búsqueda de nuevas alternativas de microcontroladores de bajaran los costos. En su creación contribuyó el estudiante Colombiano Hernando Barragán, quien desarrolló la tarjeta electrónica Wiring, el lenguaje de programación y la plataforma de desarrollo. La primera placa Arduino Fue presentada en el 2011, en los EEUU y ha sufrido tres revisiones por que el modelo actual se suele llamar UNO Rev3 o simplemente UNO R3.

Actualmente hay gran variedad de versiones, entre ellas Arduino Uno, Arduino Leonardo, Arduino Mega, Arduino Nano, entre Otros. Para el proyecto en mención se ha usado el Arduino Uno, esté cuenta con 14 pines digitales de entrada / salida (de los cuales 6 se pueden utilizar como salidas PWM), 6 entradas analógicas, 16 MHz como Frecuencia del Reloj, una conexión USB, un conector de alimentación, Voltaje de entrada de 7 – 12V, Voltaje del sistema 5V, Microcontrolador ATmega328, Memoria Flash de 32Kb y un botón de reinicio. Se programa mediante el uso de un lenguaje propio, basado en un nivel alto de programación C.

Ethernet Shield.

La tarjeta Ethernet Shield es un accesorio, la cual permite al Arduino conectarse a una red cableada TCP/IP. Una vez conectado este shield sobre la placa UNO gracias a que los pines encajan
perfectamente arriba y abajo, para nuestros circuitos se utiliza a partir de entonces las entradas y
salidas ofrecidas por los pines hembra del shield Ethernet. Estas entradas y salidas tienen exactamente
la misma disposición y funcionalidad que las de la placa UNO.
Esta placa Arduino Ethernet permite, transferir datos entre ella misma y cualquier otro dispositivo
conectado a su misma red LAN, o viceversa. También se puede lograr, gracias al establecimiento de
enrutamiento de paquetes adecuado, comunicar nuestra placa Ethernet con cualquier dispositivo
conectado a cualquier red del mundo fuera de nuestra LAN privada, incluyendo Internet, mediante la
librería de programación Ethernet, y la correcta configuración de direcciones IP, direcciones MAC,
etc; con lo que sus posibilidades de uso se disparan. La placa Arduino Ethernet también dispone de
un zócalo para insertar una tarjeta micro SD, la cual puede ser usada mediante la librería de
programación SD incorporada por defecto dentro del lenguaje Arduino, para guardar diferentes tipos
de ficheros y ofrecerlos a través de la red. Hay que tener en cuenta que si la tarjeta micro SD está
presente, el pin 4 está reservado para el control de esta.

3.2.1  **Estructura General de Un sketch.**

Un programa diseñado para ejecutarse sobre un Arduino, un sketch siempre se compone de tres
secciones:

- La sección de declaraciones de variables globales: Ubicada directamente al principio del
código.

- La sección llamada Void Setup(): Delimitada por llaves de apertura y cierre, donde las
instrucciones escritas dentro de esta sección se ejecutan una única vez, en el momento de
encender la placa Arduino.

- La sección llamada Void Loop(): Delimitada por llaves de apertura y cierre. Las instrucciones
escritas dentro se ejecutan infinitas veces hasta que la placa se apague(o se resetee). Es en este
punto donde está el programa en sí y que funciona continuamente.(Torrente, O 2013).
3.3 Softphone

Los softphone son teléfonos implementados por software. Normalmente, un softphone es parte de un entorno voz sobre IP y puede estar basado en el estándar SIP. Éstos proporcionan a un dispositivo que no es propiamente un teléfono, como un ordenador o una PDA, las funcionalidades de un teléfono VoIP. Para ello solo es necesario que el equipo cuente con audio adecuado y alguna forma de conectarse a una red TCP/IP. En este proyecto se ha usado uno de los softphone típicos basados en SIP, Zoiper. Se ejecuta en diferentes plataformas, MAC, Linux, Windows, IPhone, Android, y es compatible con la mayoría de proveedores de servicios VoIP y PBX. Cuenta con una versión no comercial, Free Zoiper en la cual se pueden realizar varias funciones entre ellas hacer o recibir Llamadas, Grabar Llamadas, Iniciar y manejar conferencias, entre otras funciones básicas de la telefonía. La principal ventaja de un softphone sobre un teléfono VoIP hardware es el costo, Gratuito. Los softphone tienen por delante un largo camino hasta que puedan ser aceptados por la mayoría de usuarios. A la costumbre de usar un teléfono tradicional. En la ilustración 4 se muestra la interfaz al usuario de este software. (Gómez, J; Gil F, 2009).

Ilustración 2 Softphone Zoiper.
3.4 Estado del arte

Más de 314.000 Arduino vendidos y desarrollos de hardware para aplicaciones en el mundo real. La integración de estos dos sistemas es una integración sin límite ya que se tiene acceso a todos los códigos fuente, a todos los esquemas, y los costos son nulos. Cada día se ven más proyectos interesantes donde tanto como Arduino y Asterisk se adentran un poco más en la vida cotidiana. Muchas veces se necesita una solución a un determinado problema logrando resolverlo de una forma sencilla y práctica utilizando componentes simples como los son estos dos sistemas, a continuación se relacionan desarrollos con estas dos tecnologías:

3.4.1 Proyecto: 1:

Título: BOTANICALLS

Proponentes: Rob Faludi - Kate Hartman - Kati London.
UNIVERSIDAD DE NEWYORK

Descripción: Es un dispositivo que mide la humedad de un cultivo o planta, y en el momento que se baja de un determinado nivel de humedad se realiza una llamada a través del sistema Asterisk a un teléfono configurado y reproduce un mensaje dando por entendido que es necesario de su asistencia humana para estabilizar los niveles censados. Hoy día el sistema envía un mensaje en twitter reportando dichos niveles de humedad, tiene un costo aproximado al mercado de 150USD. Este desarrollo está basado en el uso de Arduino, Xbee y Asterisk, un sistema de telefonía de código Abierto. Un sensor de humedad del suelo en cada planta envía información a una placa Arduino que luego pasa la información a una red ZigBee. Un XPort se utiliza para enviar datos a través de PHP a una base de datos MySQL y para Asterisk, que impulsa las llamadas telefónicas. Incluso si el usuario desea puede hacer una llamada a la planta ya que previamente se le ha configurado un número telefónico y podrá conocer el estado de salud de este vegetal.
3.4.2 Proyecto 2:

Título: DESARROLLO E IMPLEMENTACIÓN DE UNA PBX INTEGRADA CON VOZ SOBRE IP Y UN SISTEMA DE RESPUESTA DE VOZ INTERACTIVO BASADO EN EL SOFTWARE LIBRE ASTERISK

Proponentes: BOJACÁ CABRA JOHN NEIVER
UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS.

Problemática: Este trabajo pretende dar a conocer la teoría introductoria para la telefonía, la redes de datos y la voz sobre IP, para que el lector pueda comprender los términos en los se hace referencia al desarrollo de la práctica. También una pequeña descripción de la red digital de datos que posee la empresa, mostrando algunas de las Desarrollo e implementación de una PBX integrada con voz sobre IP y un sistema de respuesta de voz interactivo basado en el software libre ASTERISK características fundamentales y que conciernen a la práctica. La situación actual de la empresa al inicio de la pasantía, con respecto al sistema electrónico de comunicación que posee, en este caso una PBX Panasonic, describiendo el funcionamiento de este sistema. También, mostrar al lector que definitivamente este sistema no es deficiente, sólo que el desarrollo de nuevas tecnologías que incorporan integradamente toda una gama de soluciones de comunicaciones en telefonía, ha hecho que estas soluciones no sean suficientes para satisfacer las necesidades de la comunicación usuario empresa. Lo que se montó en esta red, son protocolos que permitan tener Voz sobre IP, posibilitando la comunicación integrada tanto dentro de las oficinas de la empresa, como en la interconexión de la empresa con el exterior. Además Voz sobre IP ofrece muchas otras características que hacen que este entorno de comunicación sea más cordial (Centros de Llamadas, Multi conferencia, etc). Este proyecto fue desarrollado en el 2007, facultad de ingeniería, como Trabajo de grado para aplicar al título de Ingeniería en Electrónica. Es una pasantia enfocada a la empresa UBIQUANDO que se dedica fundamentalmente al desarrollo e integración de software libre, velando por varias necesidades provenientes de las empresas colombianas en el ramo de las telecomunicaciones, y la informática.
3.4.3 Proyecto 3:

Título: APROXIMACIÓN DE LA CALIDAD DE VOZ Y COBERTURA EN UN RED GSM DE EMERGENCIA.

Proponentes: Elvis Eduardo Gaona García - Miguel Antonio Avila Angulo - Elkin Gabriel Muskus
UNIVERSIDAD MILITAR NUEVA GRANADA – BOGOTA COLOMBIA

El objetivo fundamental de esta investigación fue brindar una solución propuesta de telecomunicaciones para la atención de desastres en situaciones de emergencia, haciendo uso de software libre para interconectar llamadas, protocolos de comunicaciones empleados en las redes GSM en la banda DCS-1800, que facilita el uso de cualquier terminal móvil comercial, utilizando un USRP (Universal Software Radio Peripheral) en conjunto con un sistema de telefónía IP basado en Asterisk. El USRP se encarga de emular la operación de una BTS (Base Transceiver Station) al ejecutar las funciones de un PBX (Private Branch Exchange), permitiendo la coordinación de las entidades responsables de responder y atender este tipo de emergencias. Las mediciones del desempeño de la red se realizaron a partir de tres parámetros probabilísticos: GoS (Grade of service), DCR (Dropped Call Rate) y MOS (Mean Opinion Score). Esta investigación fue desarrollada en el 2014.

3.4.4 Proyecto 4:

Título: CONTROL DE ACCESO RFID A TRAVÉS DE ARDUINO ETHERNET SHIELD CON EXTENSION ASTERISK.

Proponente: Kleid Mevaller

Otro interesante proyecto es un control de acceso que utiliza una placa Arduino Ethernet y un sistema de lector RFID. El sistema detecta las tarjetas RFID de los clientes y mediante una red Ethernet
procesa la información permitiendo cambiar el comportamiento de una extensión del Asterisk, en función si la persona se encuentra dentro o fuera de una ubicación sin necesidad de estar programando desvíos. Es decir que usa el control de presencia de la oficina para poder redirigir las llamadas cuando la persona no se encuentre en su lugar de trabajo, a su móvil. Pero si la persona se encuentra dentro de la oficina las llamadas entran a su extensión.

El lector RFID se conecta al Arduino por un puerto serie, está recibe un número de serie de la tarjeta comprobando que el número de serie es válido, se compara con una base de datos, y finalmente se manda un comando al Asterisk quien se encarga de realizar la llamada a la extensión o móvil telefónico para realizar los desvíos de una forma automática.

**3.4.5 Proyecto 5**

Título: IMPLEMENTACIÓN DE ARDUINO RASPBERRYX USANDO UN SERVIDOR ASTERISK.

Los usuarios de Ras PBX obtienen una opción adicional hoy, el conjunto completo de aplicaciones de PBX, como instalar un opcional en la parte superior de Ras PBX. Los servidores PBX han sido famosos durante muchos años en las plataformas x86, y también este proyecto cuenta con el Raspberry Pi desde mediados de 2012. Ahora con el nuevo enfoque de un instalador add-on, los usuarios obtienen los beneficios combinados de ambos Ras PBX y el servidor PBX. El Raspberry Pi ha tenido gran auge desde que se anunció por primera vez. Se convierte en una computadora del tamaño de una placa, y es capaz de muchas de las cosas que hace un PC de escritorio, como hojas de cálculo, procesadores de texto y juegos. También reproduce vídeo de alta definición. Se puede ejecutar en varias versiones de Linux y está siendo utilizado para enseñar a los niños de todo el mundo cómo programar. En los días actuales, si se conecta cualquier dispositivo directamente a Internet sin un sistema de protección mínima, se pueden encontrar problemas, por lo que este proyecto va a ayudar a proporcionar cierta protección contra los piratas informáticos, para la instalación de Raspberry Pi Asterisk.
CAPITULO 4
IMPLEMENTACIÓN DEL PROYECTO

La implementación del sistema en general se divide en 2 partes que son hardware y software. Respecto al hardware se utilizó un ordenador de escritorio con procesador Intel Core 1.50 GHz, 6GB de memoria RAM, Sistema operativo de 64 Bits, Windows 8, 450 GB de disco duro, un módulo Arduino referencia UNO R3 con su respectiva tarjeta Ethernet Shield, un teléfono celular con SO Android, un sensor de movimiento, un sensor infrarrojo, como actuador un electroimán y un modem que permite la comunicación LAN y WAN entre los dispositivos nombrados.

En lo referente al software, se instalan 3 máquinas virtuales bajo la plataforma VMware Workstation Versión 8, distribuidas de la siguiente manera, la primera máquina virtual se usó para el servidor PBX donde instaló la distribución para Windows de AsteriskNOW Versión 6.12 64-bit, y en las otras dos máquinas se ubican dos softphone de la comercialización de Zoiper; adicional se descarga y ejecuta directamente en el PC el software Arduino versión 1.6.5. En los siguientes subcapítulos se describe en detalle todo el hardware y software de este sistema.

4.1 Hardware del Sistema

4.1.1 Sensor infrarrojo

Se utilizó un sensor PIR (Infrarrojo Pasivo) para detectar los cambios de radiación por movimiento que genera un cuerpo el cual que estaría en el lugar o vivienda donde se implementó el sistema. El sensor utilizado se muestra en la ilustración 3 y tiene las siguientes características principales:

- Alcance frontal de 18 metros, lateral de 26 metros, y altura de 2,3 metros.
- Voltaje de Alimentación 12VDC, corriente mínima de operación de 17mA.
- Rango de temperatura funcionamiento -10°C a 55°C.
- Protección contra Radiofrecuencia de 30V/m a 1GHz. (Honeywell, 2008).

Ilustración 3 Sensor Infrarrojo PIR IS2560T

4.1.2 Sensor magnético.

Este sensor mostrado en la ilustración 4 se implementa para supervisar continuamente el estado de la puerta o ventana del área a proteger, simplemente funcionará como un contacto seco, que cambiará de estado manualmente al abrir la puerta, y este estado se visualizará desde el Softphone con el siguiente mensaje en pantalla “Arduino, Sensor Magnético Activo <000>”.

Ilustración 4 Sensor Magnético 7939WG-WH
4.1.3 Electroimán 12VDC.

Este elemento será el actuador del sistema, para permitir el cierre y ajuste de la puerta en caso de emergencia si así lo desea y ordena el usuario final del Softphone. El dispositivo es el que se puede visualizar en la imagen 5.

![Ilustración 5 Electroimán 12VDC](image)

**Fuentes de alimentación del sistema**

Para el sistema es necesaria la alimentación del Arduino, el sensor infrarrojo, y el electroimán. A continuación se muestra una pequeña tabla con la relación de potencias requeridas por lo elementos:

<table>
<thead>
<tr>
<th>Elemento</th>
<th>Voltaje suministrado</th>
<th>Potencia requerida</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arduino UNO e Ethernet Shield</td>
<td>5 VDC</td>
<td>250 mA</td>
</tr>
<tr>
<td>Sensor Infrarrojo</td>
<td>12 VDC</td>
<td>204 mA</td>
</tr>
<tr>
<td>Electroimán</td>
<td>12 VDC</td>
<td>235 mA</td>
</tr>
</tbody>
</table>

Tabla 2 Potencias requeridas por los elementos.
De esta forma se concluyó que las fuentes a utilizar deben ser una fuente de 12 VDC a 1 Amperio, y una fuente de 5 VDC a 0.5 Amperios, estas son independientes para evitar posibles fallas de potencia con el electroimán que pueden generar inconvenientes sobre todo en el funcionamiento del microcontrolador de la placa de Arduino UNO. A continuación el diseño del circuito electrónico para la conexión de actuadores y sensores a la placa de Arduino UNO.

### 4.1.4 Arduino UNO y tarjeta Ethernet Shield

En la ilustración 6 muestra la placa Arduino UNO (Izquierda) en su variante convencional, la cual lleva el encapsulado físico del microcontrolador montado en formato DIP(“Dual In-Line Package”). Para la alimentación de esta placa se dispone de dos medios: por USB o por un adaptador AC/DC. En el primer caso se dispone de un conector USB hembra tipo B. La alimentación recibida de esta manera está regulada permanentemente a los 5V de trabajo y ofrece un máximo de hasta 500mA de corriente, por lo tanto la potencia consumida por placa es de aproximadamente unos 2,5 W. Si en algún momento por el conector USB pasa más intensidad de la deseable, la placa Arduino está protegida mediante un polifusible reseable que automáticamente rompe la conexión hasta que las condiciones eléctricas vuelven a la normalidad. Y en el segundo caso, la placa dispone de un zócalo donde se puede enchufar una clavija de 2,1 milímetros de tipo “Jack”. En teoría el rango recomendado de voltaje de entrada es de 7 a 12 voltios, lo ideal sería un adaptador de 9 ó 12 V DC a 250 mA; este voltaje de entrada ofrecido por la fuente externa siempre es rebajado a los 5 V de trabajo mediante un circuito regulador de tensión que ya viene incorporado dentro de la placa. (Torrente, O 2013).

La placa Ethernet Shield (Derecha) que se muestra en la siguiente imagen es la que nos ha permitido la conexión LAN e Internet de nuestro sistema. Funciona como servidor que escucha y acepta peticiones remotas del Asterisk en el puerto especificado o como Cliente que puede conectarse a una dirección IP especificada a internet.
En la ilustración 7 siguiente se muestra el circuito usado para la conexión y adaptación de los sensores al Arduino, se empleo un relé a 12VDC manejado por un transistor 2N3904 que se activa desde una de las salidas digitales del Arduino. El diodo que está conectado en paralelo con la bobina del relé, se usa para evitar los posibles picos de corriente en retorno al golpear el núcleo de la bobina del relé.
4.1.5 Teléfono Android

El teléfono es el medio en que el usuario podrá interactuar con el sistema de forma remota a través de una red Internet, y donde recibirá las alertas emitidas por los sensores en tiempo real. Es allí donde se instala el software Zoiper. En esta ocasión se ha hecho uso un teléfono Android claro está que se puede usar otros dispositivos con sistemas operativos de Smartphone como iOS, Windows Mobile.

4.1.6 Modem

También se hace necesario un modem, un enrutador o un dispositivo que combine ambos. Modem un acrónimo formado por los terminos modulación y demodulación, que convierte las señales digitales en análogicas y viceversa, que nos permite conectar los equipos (PC – Servidor Asterisk y Arduino) para acceder a la Internet.

4.1.7 PC, Servidor PBX Asterisk

En cuanto a los requerimientos de y elección de hardware para la instalación de nuestro servidor PBX Asterisk se hace sencillo, ya que este software es poco exigente a la hora que solo se utilizara proveedores de telefonía VoIP y no se usara líneas telefónicas convencionales, así que recurrimos a una máquina virtual en un entorno de WMware Workstation, y se integró AsteriskNOW, sin ningún inconveniente, ahorrando máquinas y aprovechando el funcionamiento de los equipos.

Finalmente se consolidan los elementos hardware del sistema en la siguiente ilustración 8
4.2 Software del Sistema

4.2.1 Servidor PBX AsteriskNOW

Para la implementación de este proyecto se escogió la distribución Microsoft Windows del sistema operativo Linux y la versión 6.12.65 de AsteriskNow 64 Bits. (Russell B, 2012, Descargas http://www.asterisk.org). Para la visualización del servidor PBX, es muy importante decir que se utilizó una máquina virtual, esto se hace por prevención de fallos y errores en el PC ya que de esta forma pueden ser corregidos más fácilmente. Adicionalmente la utilización de máquinas virtuales permite la creación de varios “usuarios”, para hacer pruebas de funcionamiento. En la tabla 3 se hace un comparativo entre algunos software que sirven para generar máquinas virtuales sobre los PC.
<table>
<thead>
<tr>
<th></th>
<th>VMWare</th>
<th>VirtualBox</th>
<th>Xen</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conocimiento requerido para la</td>
<td>Medio</td>
<td>Bajo</td>
<td>Alto</td>
</tr>
<tr>
<td>administración.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Soporte Network Bridge</td>
<td>Si</td>
<td>Si</td>
<td>Si</td>
</tr>
<tr>
<td>Sistemas operativos que soporta</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Windows XP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Windows 2000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Windows 2000 Server</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Windows 2003 Server</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Linux Fedora</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Linux Fedora</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Integração Video I/O</td>
<td>Medio</td>
<td>Alto</td>
<td>Bajo</td>
</tr>
</tbody>
</table>

Tabla 3 Máquinas Virtuales.

Como conclusión de esto, se utilizará el software que más se adapte a las necesidades y exigencias del proyecto, Virtual Box es una herramienta básica, y que sirve a sus usuarios para explorar otros sistemas operativos, puede ser utilizada de manera óptima, pero su uso no ha sido familiar para ninguno de los proponentes de este proyecto. Por otro lado, Xen es un gran sistema complejo, que solicita licencias a un elevado precio, y aunque puede significar una gran herramienta de desarrollo, el simple hecho de tener un valor económico va en contra de las políticas de este proyecto, Desarrollo libre sobre plataformas de software y hardware libres.

Para la implementación de los scripts que dieron paso al cumplimiento de los objetivos se configuraron los siguientes archivos sip.conf y extensions.conf (extensions_custom.conf para AsteriskNOW), este es uno de los archivos más importantes de la PBX, ya que es en el cual se describe el plan de marcación más conocido como el Dialplan, la columna vertebral del sistema, donde funciones, aplicaciones y recursos se van intercambiando para formar programas, procedimientos que suplan diferentes necesidades de un usuario dentro del mundo de las PBX. Cuando se habla de Plan de Marcación, se hace referencia literalmente a la marcación de un teléfono...
o dispositivo, que lleva a un determinado contexto generalmente denotado entre llaves [contexto] correspondiente a la marcación hecha, ver la ilustración 9.

Dentro de ese contexto, están definidas las extensiones, prioridades con las que empieza a recorrer el script secuencialmente y a partir de ahí puede surgir una serie de eventos, ya que por ejemplo, podría darse como encender la luz de la oficina, abrir puertas, entre otras aplicaciones que permite el mundo de la comunicación. Los contextos son la forma de categorizar o englobar, una serie de procedimientos en función de la marcación a la que se hace referencia desde el dispositivo usado. Si se marca a la extensión de un usuario y no está agregada al contexto que le corresponde o no existe dicho contexto, entonces no se genera la llamada y se mostrará el log de eventos en la consola de Asterisk indicando el inconveniente.

4.2.2. Software Arduino

Para nuestro desarrollo se quiere tener una aplicación domótica donde se obtengan datos de diferentes sensores y a partir de esta información ejecutar actuadores, esto se realiza gracias al Arduino que es
programado con su propio compilador y última versión 1.6.5 compatible con cualquier referencia de Arduino. Para la implementación del código se tuvo encuentra la siguiente lógica algorítmica, que se muestra en la ilustración 10.

4.2.2 Software Zoiper

Comunicación LAN: Se crean dos máquinas virtuales con Windows XP, en donde se realizará la instalación de los Softphone Zoiper. Cada Windows XP debe responder ping a la IP del Asterisk y entre ellas por eso se hace necesario ir al asistente para configuración de red, e indicar que el equipo pertenece a una red que no tiene una conexión a Internet y activar el uso compartido de archivos. Una
vez se compruebe el estado de la comunicación se puede proceder a crear los softphone con cuentas con formato SIP, bajo los siguientes parámetros:

- **Dominio**: IP Asterisk
- **Username**: Número de la Extensión
- **Password**: Clave de acceso a Asterisk CLI
- **ID Name**: Nombre que se desee Visualizar.

<table>
<thead>
<tr>
<th>Opciones de la cuenta SIP</th>
<th>Registrando</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dominio: 192.168.1.8</td>
<td></td>
</tr>
<tr>
<td>Nombre de usuario: 101</td>
<td></td>
</tr>
<tr>
<td>Contraseña: ···········</td>
<td></td>
</tr>
<tr>
<td>Nombre de origen:</td>
<td></td>
</tr>
</tbody>
</table>

Ilustración 11 Configuración Zoiper Comunicación local.

Comunicación WAN: Se instala la aplicación para dispositivos móviles de Zoiper y se crea una cuenta con los parámetros similares la distribución anterior solo que en dominio o Host se ha de colocar la IP pública del Asterisk, o DNS. Más adelante en el capítulo 5 se mostrará el dominio usado para este proyecto.

### 4.3 Descripción Funcional del Proyecto.

El sistema domótico funciona con un sistema integrado entre Asterisk y Arduino, en donde este último se encuentra monitoreando dos sensores ubicados dentro de una vivienda, uno es un sensor infrarrojo para detectar movimientos, el otro es un sensor magnético para supervisar la apertura de una puerta, ventana, cajón, y otros, según lo que el usuario desee. Estos sensores envían una señal de activación a los pines de entrada de Arduino, el cual reacciona enviando una señal a través de la red de
LAN al servidor virtual PBX de AsteriskNow, quien se encarga de realizar automáticamente una llamada Ethernet informando al usuario, sobre la novedad.

El usuario para recibir la alerta enviada por el sistema debe llevar consigo el softphone con acceso a Internet, permiéndole estar informado de la activación de uno de estos o de ambos sensores, y tendrá la opción de realizar la activación de un electroimán o actuador marcando una tecla del softphone con la finalidad de cerrar una puerta definitivamente energizando este electroimán y tomar medidas correctivas si así lo considera necesario. Este Softphone puede estar ejecutado desde un PC o si el usuario lo desea también a su dispositivo móvil Android.

Un ejemplo claro es, si ingresa un delincuente a robar en una vivienda, será detectado por la apertura de la puerta o ventana, o por realizar movimientos dentro del rango de vista del Infrarrojo, esto ocasionará una llamada automática al dueño de la vivienda quien pulsará una tecla para que el electroimán se active cerrando definitivamente la puerta y dejando al delincuente encerrado en la vivienda, así que es una trampa para delincuentes por decirlo así.

Este proyecto puede llegar a convertirse de esta forma en una gran herramienta para realizar acciones remotas en hogares, empresas, oficinas, industrias, lugares comerciales y demás locaciones desde cualquier lugar del mundo, siempre que se tenga acceso a internet. Así que simplemente esta es una de las tantas aplicaciones que se pueden tener integrando Asterisk y Arduino.

En la ilustración 10 se muestra un esquema del funcionamiento del proyecto:
4.4  Instalación de Asterisk

Para comenzar a desarrollar el proyecto se realizó la instalación del servidor virtual PBX. Como ya se ha mencionado antes se utiliza AsteriskNow, ya que es un paquete de software libre que permite básicamente convertir un PC en una central telefónica PBX y está basado bajo Asterisk y se ejecuta en la máquina virtual y su correcta configuración se muestra a continuación, cada paso para lograr un perfecto funcionamiento de nuestro PBX.

En la ilustración 11 a), se indica que la configuración se realizara manualmente, ya que de la forma típica, se crea el tipo de disco de forma incorrecta. Seguido de eso, Ilustración 6 b), es conveniente
dejar para el final la asociación del .iso en donde se encuentra el programa, ya que una vez haciendo las pruebas esto nos producía error cuando se relacionaba la ruta del archivo .iso (asteriskNow), en este paso.

![Ilustración de instalación de Asterisk](image)

**Ilustración 13 Instalación Asterisk A**

La ilustración 12 agrupa 4 pasos importantes, c) en donde se indica el sistema operativo a trabajar y su versión para este caso es Linux por las razones que ya se mencionaron anteriormente y la versión Red Hart Linux diseñado para su fácil uso; d) se da un nombre y ubicación a la nueva máquina virtual que se está creando, e) es suficiente con 512MB para la memoria que usará el PBX, f) la conexión a internet debe ser bridged networking o puente de red, dado que interconecta segmentos de red (o divide una red en segmentos) haciendo la transferencia de datos de una red hacia otra con base en la dirección física de destino de cada paquete.
Para el tipo de controlador se acepta la recomendada por el sistema busLogic como se muestra en la Ilustración 13 g), se crea un nuevo disco virtual pero debe ser IDE su traducción componente electrónico integrado, ya que de lo contrario con el controlador SCI, AsteriskNOW no encuentra el disco emitiendo el siguiente error “No usable disk have been found”. Así que importante que sea IDE.
Y por último en la ilustración 13 j) se sigue las recomendaciones de instalación dejando 8 GB para el disco de la máquina virtual.

Ilustración 15 Instalación Asterisk C
En la ilustración 14, lo que se hace es continuar y finalizar la instalación. Cada vez más cerca a tener funcionando un central PBX, totalmente gratis y Open Source como lo es Asterisk con su última versión AsteriskNOW 6.12 para 64 Bits.

La máquina virtual queda creada, y debe cargarse el archivo “.iso”, como se muestra en la ilustración 15 por lo que hay que configurar las propiedades de la nueva máquina virtual y se le ingresa la ruta del archivo “.iso” previamente descargado de la página Wiki de Asterisk. (Russell B 2012).
Antes de continuar y para evitar inconvenientes en la red, se recomienda restaurar por default la configuración de red del VMWare, y garantizar que este VMnet0 tipo bridged, tal como se muestra a continuación:
Con la máquina virtual funcionando correctamente, se realizó la instalación de AsteriskNow con los detalles que se muestran en las siguientes imágenes:

![Welcome to FreePBX 6.12.65](image)

Ilustración 19 Configuración Asterisk

Se da los permisos para que la configuración TCP/IP sea por DHCP, aunque más adelante se configura una IP estática que es la 192.168.1.8, se ajusta la zona horaria, seleccionando la opción América/Bogotá, correspondiente a nuestra ubicación geográfica actual.

![Configure TCP/IP](image)

![Time Zone Selection](image)

Ilustración 20 Parámetros básicos.
Ilustración 13. Es importante la asignación de esta contraseña y tenerla presente ya que será la que nos pedirá a la hora de encender el servidor PBX, adicional a esto será la que se configure en cada uno de los softphone a utilizar. Para este caso la contraseña asignada es 747sd951, como elección personal de los proponentes del presente proyecto.

![Ilustración 21 Asignación de Claves.](image)

Una vez completado se llega a la consola de Linux, al comando de inicio de sesión del sistema, ingresando con el usuario y la contraseña registrada anteriormente. Para finalmente ingresar a la interfaz del software de AsteriskNow.

Los datos que se pueden ver en la imagen 14 son la IP que ha tomado por DHCP: 192.168.1.6 y la MAC con la que ya ha de trabajar la máquina virtual: 00:0C:29:93:5F:8A.

![Ilustración 22 Interfaz de Usuario AsteriskNow](image)
4.5 Configuración de Archivos Asterisk

Existe una gran cantidad de archivos de configuración de Asterisk que se utilizan según sea el caso, como ya se mencionó en el apartado 4.2.1 el fichero principal es extension_custom.conf, además se tienen otros recursos que requieren fuentes de datos externas. En general, podemos resumir los directorios usados por la PBX Asterisk los cuales se presentan en la Ilustración 9, y en la tabla se muestra la respectiva descripción de cada directorio o subdirectorio.

Ilustración 23 Directorios y subdirectorios usados por Asterisk.

<table>
<thead>
<tr>
<th>DIRECTORIO (RUTA)</th>
<th>LOCACIÓN</th>
</tr>
</thead>
<tbody>
<tr>
<td>/etc/asterisk/</td>
<td>Contiene la mayor parte de archivos de configuración de Asterisk (.conf).</td>
</tr>
<tr>
<td>/etc/sysconfig/network-scripts/ifcfg-eth0/</td>
<td>Este archivo especifica el nombre del dispositivo de la interfaz de red, la dirección IP, la máscara de la red, la dirección de enlace, la dirección de broadcast e información sobre el modo de arranque.</td>
</tr>
<tr>
<td>/var/lib/asterisk/</td>
<td>Contiene diversos ficheros, en distintos subdirectorios importantes para Asterisk, base de Datos etc.</td>
</tr>
<tr>
<td>/var/lib/asterisk/agi-bin/</td>
<td>Directorio donde se encuentran los scripts AGI que pueden ser ejecutados desde el dialplan.</td>
</tr>
<tr>
<td>/var/lib/asterisk/sounds</td>
<td>Contiene archivos de audio que son usados para mensajes de voz, alertas, contestadoras, entre otras</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>/var/www/html/</td>
<td>Es el directorio de los documentos WEB.</td>
</tr>
</tbody>
</table>

Tabla 4 Descripción de directorios usados por Asterisk.

A continuación se detalla la configuración que se debe tener presente en cada uno de los ficheros nombrados y mostrados anteriormente para lograr y cumplir los objetivos planteados.

### 4.5.1 Asignación de IP Estatica.

Es más que necesario, que Asterisk cuente con una IP que no varié cada vez que se inicie el servidor y una comunicación fallida. Se revisa el sistema con “ifconfig”. Este comando se ejecuta fuera de la línea de comandos CLI, es decir se realizará en el primer entorno luego de loguearnos, sin ejecutar asterisk –r, tal como se visualiza en la ilustración 24.

![Ilustración 24 Ifconfig, Configuración de Red en Asterisk](image)

Lo importante en la anterior imagen es validar la IP que tiene el Asterisk, la MAC, la máscara de red, ahora si se desea cambiar esos parámetros se debe bajo el comando **nano** (modificar) entrar a la siguiente ruta: **nano /etc/sysconfig/network-scripts/ifcfg-eth0**, y anexar lo que se muestra en la ilustración 25.
Ilustración 25 Asignación de IP estática a nuestro servidor PBX Asterisk

Ctrl+O, para guardar cambios, Ctrl+X para salir. Se restaura la red con service network restart.

Arrojándonos los resultados que se muestran en la imagen 26.

Ilustración 26 Service Network Restart
4.5.2 Extensions_custom.conf

Ubicado en ruta /etc/asterisk/extensions_custom.conf, cuya disposición final se muestra en la ilustración siguiente encontramos:

Grabación de Audio, Cuando el usuario recibe una alerta escuchará un mensaje como el siguiente, “Conexión Arduino, se ha detectado movimiento si desea activar el electroimán marque 5 después del tono de lo contrario cuelgue para terminar” siguiente mensaje  Al marcar desde la nuestro softphone la extensión 999, se responde la llamada, genera un retardo, invoca la aplicación Record la cual permite realizar una grabación de voz nombrado Llamada_arduino.gsm, se marca numeral para que el sistema identifique la grabación se ha terminado, inmediatamente después se reproduce la grabación hecha y la llamada se colgará. Este archivo queda ubicado en la ruta cd /var/lib/Asterisk/sounds ls.

Extensión del Portero Automático, Se designa el número 103 como la extensión que al ser marcada podrá el usuario accionar el actuador, gracias a la ejecución del AGI. Conexión_Arduino.php que más adelante se mostrará con su detalle.
Por último se tiene el contexto [arduino_call.php], con el ánimo de que una vez el Arduino llame a Asterisk le indique que debe ir al contexto [arduino_call] y ejecutar las instrucciones que allí se plantean, que como podemos ver lo que hace es reproducir el audio previamente nombrado y permitir activar el electroimán gracias al AGI.

Finalizando así la configuración del Dialplan, el corazón de Asterisk cabe aclarar que luego de todo cambio en este archivo se deberá ejecutar el comando dialplan reload, para su respectiva actualización y toma de los cambios hechos.

4.5.3 Implementación de Scripts 1. Conexión_Arduino.php


```
#!/usr/bin/php -q

<?php

require("phpagi.php");
$agi= new AGI();
$arduino_ip= "tcp:////192.160.1.20";
$arduino_port = 23;

$fp = fsockopen($arduino_ip, $arduino_port, $errno,$errstr);
if (!$fp){
exit();
}

$continue= true;
while ($continue){
    $return=$agi->get_data('beep',10000,1);
    if (($return['result'])==0)and($return['result']<9)){
        $ascii=chr($return['result']);
        fwrite($fp,$ascii);
    }
}
fclose($fp);
?>
```

Ilustración 28 AGI: Conexión_Arduino.php
Se requiere de la librería phpagi, la cual debe estar dentro del mismo directorio. Se declara las variables $arduino_ip en donde se almacena la IP que se le ha asignado a nuestro Arduino y $arduino_port, para el puerto de comunicación que se ha usado para la comunicación entre Arduino y Asterisk. Seguido de eso se abre una conexión vía sockets al recurso especificado, con la instrucción fsockopen. Una vez se ha establecido la conexión se recibe el valor de la tecla marcada en el softphone con get_data para ser guardado en la variable $return, que más adelante se envía en su valor ascii por la conexión vía socket al Arduino. Quien tiene como condicional que al recibir un 5 active la salida 4 de la placa, correspondiente al actuador.

4.5.4 Implementación de Scripts 2. Arduino_call.php

La configuración del servidor web, se encuentra en /var/www/html, allí se crea el archivo de nombre arduino_call.php, con nano arduino_call.php incluyendo el código tal como se muestra en la ilustración 29.

Ilustración 29 Configuración para la recepción de una llamada hecha por Arduino a Asterisk. Arduino_call.php
En la línea número 8 del código de figura anterior podemos ver como convocar al contexto “arduino_call”: Sasterisk_context = “arduino_call”; También se ve como se comparan las variables $remote_ip que es de donde se está recibiendo la información, con la variable $arduino_ip que ha sido declarada con la IP con la que se ha configurado el Arduino. Es decir si la IP de la que se recibe la información es exactamente igual a la IP que se ha definido para el Arduino procede a ejecutar el programa donde automáticamente llamará a una extensión programada en este ejemplo a la 101 y ejecutará el contexto arduino_call, que si vemos responde la llamada, reproduce un audio de alerta permitiendo interactuar con el sistema. (Spencer, Mark 2003)

![Diagrama de flujo](image)

Ilustración 30 Implementación Scripts 1 y 2.
5.1 Pruebas y Resultados del Sistema.

5.1.1 Primera Prueba. Comunicación LAN

Se establece una llamada entre dos extensiones configuradas dentro de la misma red local. Para entrar a familiarizarnos con la plataforma web de AsteriskNOW, se debe introducir en nuestro navegador la dirección IP que tiene el Asterisk en esta oportunidad es la 192.168.1.6. La primera vez que se realiza el procedimiento se solicita para la configuración del administrador, un nombre de usuario y una contraseña de administrador. Este nombre de usuario y contraseña se utiliza de aquí en adelante para acceder a la pantalla de configuración de FreePBX. Este usuario y contraseña son diferentes a las usadas para entrar en la máquina virtual. En la imagen 31 se muestra la configuración inicial para la interfaz de usuario Web, Free PBX accediendo a la dirección 192.168.1.6:

![Imagen de la interfaz de usuario FreePBX](image)

Ilustración 31 Administración FreePBX.
Una vez se haya accedido, en la pestaña Applications/Extensions, para poder realizar el primer enlace de llamada se crean 2 extensiones tecnología SIP, con los siguientes parámetros:

<table>
<thead>
<tr>
<th>Extensión del usuario: 101</th>
<th>Extensión del usuario: 102</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nombre para mostrar: Sandra</td>
<td>Nombre para mostrar: David</td>
</tr>
<tr>
<td>Secret: 747sd951</td>
<td>Secret: 747sd951</td>
</tr>
</tbody>
</table>

Se aplican cambios y se valida si las extensiones has sido creadas, entrando a la interfaz de línea de comandos, en inglés command-line interface o CLI, con el comando asterisk –r y sip show peers, como figura en la ilustración 32.

![Interfaz de línea de Comando CLI](image)

Ilustración 32 Interfaz de línea de Comando CLI.

Ya con esto se hace efectiva la llamada entre las dos extensiones, con una excelente respuesta y audio. A continuación, en la ilustración 33 y en la ilustración 34 se muestra la tabla estadística que indica que hay una llamada activa entre las dos extensiones creadas y la interfaz de línea de comandos, mostrando los eventos durante la llamada:
Se da cumplimiento al primer objetivo específico teniendo ya implementado un servidor virtual para una central Asterisk en un PC de control del sistema. Dando paso al segundo objetivo que se empieza a construir con teniendo el primer enlace entre Asterisk y Arduino tal como se muestra en el siguiente apartado.
5.1.2 Segunda Prueba, Conexión desde Asterisk a Arduino.

Teniendo el servidor PBX configurado y funcionando correctamente, el siguiente paso fue lograr comunicar el sistema anterior con las dos placas de Arduino UNO y Arduino Ethernet Shield, ya que por medio de este último se puede realizar una conexión a la red local donde está conectado el PBX virtual y los Softphone.

El trabajo se realiza partiendo de la idea que el módulo Arduino está esperando que alguien le envié alguna instrucción que le autorice activar una salida que corresponderá a futuro al electroimán, entonces continuamente estará leyendo lo que ingresa por TCP/IP en el puerto 23. En el lado de Asterisk un AGI(ConexionArduino.php) implementado se encarga de establecer una conexión hacia la dirección IP de la placa Arduino (192.168.1.20), para enviar los dígitos que se marquen en el Softphone, sabiendo que el número que autoriza activar la salida es el número 5.

El siguiente Paso a paso indica cómo poner en marcha la comunicación entre nuestro PBX Asterisk y la placa Arduino:

1. Nos conectamos al Asterisk de forma que nuestro usuario este identificado, siempre el usuario va a ser root, y la contraseña creada en el momento de la instalación.
2. Se debe comprobar la comunicación realizando un ping sobre la Ethernet Shield de Arduino, desde las máquinas virtuales Windows XP, desde el PC de trabajo y se logra una respuesta efectiva tal como se muestra en la ilustración 35.

Ilustración 35 Ping a la placa Arduino.

Ilustración 36 Carpeta Agi-bin.

4. Entrar a la línea de comandos con `asterisk -r`, y se marca desde un softphone se la extensión 103 se sigue las siguientes líneas:

Prioridad 1: La llamada será contestada.

Prioridad 2: Ejecuta el AGI ConexiónArduino.php

Prioridad 3: Cuelga la llamada después de ejecutar el AGI.

5. Arduino, se verifican su programación, se asocian las librerías necesarias para su conexión a la red del servidor PBX, Ethernet y SIP son las únicas usadas, y se asigna lo siguiente:

- Una dirección MAC
- Una dirección IP local
- Un puerto abierto para esta dirección
- Una salida declarada PIN 4
- Se Hace un test sobre la conexión
- Si la conexión se establece, se recibe lo que envía desde el Softphone y lo guarda en una variable
- Se compara la variable si es 5 activa el relevo, de lo contrario se cuelga la llamada.
Con lo anterior se ha logrado un enlace entre el Asterisk y Arduino, momento en que el usuario si desea activar el electroimán puede llamar la extensión 103, presionar 5 y con esto ya la puerta queda cerrada.

5.1.3 Tercer Prueba, Enlace desde Arduino a Asterisk

El objetivo ahora es realizar la comunicación del Arduino hacia el Asterisk, es decir que el Arduino llame automáticamente a una extensión una vez reciba alguna entrada proveniente de los sensores. Programa Arduino. De esta manera Arduino ya podrá realizar una conexión estable y efectiva con Asterisk cuando una de sus entradas muestre una activación. En la imagen 37 se ve como el usuario vería en su pantalla del teléfono la llamada entrante proveniente de una activación en uno de los sensores monitoreados por el Arduino. Así se da cumplimiento en su totalidad al objetivo específico número 2, Desarrollo de un script en Asterisk y el firmware en Arduino para intercambio de datos con la central Asterisk.

![Ilustración 37 Comunicación del Arduino al Asterisk, Activación Sensor magnético.](image)

Todo lo anterior ha funcionado correctamente hasta este punto, pero hay algo muy importante que le da una gran ventaja a este desarrollo, y es la capacidad de realizar sus operaciones a través de Internet. Así que es hora de implementar algunas herramientas para lograr este objetivo.
5.1.4 Cuarta Prueba, Conexión Arduino – Asterisk – Arduino remoto WAN.

Para establecer una conexión bidireccional sobre la red WAN. Debemos tener una IP pública para el Asterisk, pero como nuestro enfoque es tener los mínimos costos en el proyecto hemos creado una cuenta en NO-IP, donde podremos tener acceso a la IP del Asterisk desde una red WAN, creando gratuitamente un DDNS, con el nombre conexionarduino.no-ip.org como se muestra en la Ilustración 38.

Ilustración 38 Configuración NO-IP

En el campo IP Address, se coloca la IP externa del Router que se puede conocer mediante páginas web como ejemplo http://www.cual-es-mi-ip.net/, y para evitar que cada vez que se apague y encienda el modem este dominio cambie, debemos entrar al modem y configurar los DNS.

En cuanto al Asterisk, hay que ingresar al FreePBX web en Settings -> Asterisk SIP Settings -> Chan SIP -> NAT Settings como se muestra en la ilustración 39 habilitar el NAT, seleccionar la configuración de IP Dinamica, y en Dynamic Host, ingresar el hostname creado anteriormente, conexionarduino.no-ip.org.
Ilustración 39 Configuración NAT para tener una conexión WAN y no solo LAN.

Adicional a esto es necesario, cambiar la configuración de la App Zoiper instalada en nuestros teléfonos Android el cual hará la función de Softphone, tal como se muestra en la ilustración 40.

Ilustración 40 Configuración Zoiper Android

Como podemos ver para hacer efectiva la autenticación del usuario fuera de una red LAN es necesario que el host de destino sea conexionarduino.no-ip.org, el resto de datos se mantienen, nombre de usuario con la extensión asignada en este caso 102 y clave la que ingreso al momento de instalación del PBX asterisk y habilitar algunos puertos en el Router de acuerdo a la siguiente tabla 5, que se muestra a continuación:
<table>
<thead>
<tr>
<th>IP</th>
<th>Puerto</th>
<th>Número</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.1.8</td>
<td>UDP</td>
<td>5060</td>
</tr>
<tr>
<td>192.168.1.8</td>
<td>TCP/UDP</td>
<td>10000-20000</td>
</tr>
<tr>
<td>192.168.1.8</td>
<td>TCP</td>
<td>23</td>
</tr>
<tr>
<td>192.168.1.20</td>
<td>TCP</td>
<td>23</td>
</tr>
</tbody>
</table>

Tabla 5 Configuración de puertos en el Router

Los puertos RTP ya se encuentran por defecto configurados en el fichero del Asterisk, sin embargo se deberá validar esta información en la siguiente ruta nano /etc/Asterisk/rtp_additional.conf, como se muestra en la imagen 41.

Así damos por finalizado la ejecución de ese proyecto en donde se implementar un servidor virtual para una central Asterisk en un PC de control del sistema, Se desarrollaron scripts en Asterisk y el firmware en Arduino para intercambio de datos, se implementar el sistema domótico y la interfaz de hardware, sensores y actuadores requeridos para la plataforma Arduino, se configurar la central Asterisk para el control del sistema domótico de seguridad desde Softphones a nivel local LAN y remoto WAN.
CONCLUSIONES

- La implementación de Arduino y Asterisk conlleva la manipulación de otras herramientas importantes como máquinas virtuales, pero aun así, se complementan y convergen, mostrándose como una efectiva integración de hardware y de software para desarrollos que requieren automatizar sistemas y permitir al usuario intervención en la toma de decisiones.

- El desarrollo del Script, permitió plasmar la serie de órdenes necesarias para que el usuario final del sistema logre una buena percepción del mismo, logrando una buena interacción con el sistema operativo Linux.

- En la etapa de diseño análoga se encuentra que es un diseño de bajo consumo, gracias a que actuadores, sensores, placas Arduino utilizan poca potencia. Además de tener una buena precisión en la realización de pruebas generales al sistema.

- Fue un acierto sin ninguna duda, la instalación de VMWare, debido a que permite la reproducción de varias máquinas virtuales simultáneamente, lo que resulto siendo una gran ventaja, puesto que estas quedan conectadas en la misma red, y la realización de pruebas se redujo drásticamente con respecto de los tiempos proyectados antes de comenzar el desarrollo del proyecto.

- La configuración sencilla del Softphone no solo en el entorno de Windows XP, sino en la aplicación móvil significó una ventaja con la que no se contaba al principio del proyecto, y que resulta ser altamente efectiva, ya que el usuario no solo tiene acceso desde su lugar remoto en un punto fijo, sino que la movilidad de los dispositivos Android es un plus más para este proyecto.

- Las centrales de VoIP están destinadas a marcar un punto de partida en las comunicaciones de voz a través de la historia, ya que es posible transportar información y datos a través de estas herramientas, además de reducir costos y brindar acceso a mas usuarios a nivel mundial, la lucha
radica en el costo beneficio que puede traer consigo el enfrentamiento de poderes entre los grandes operadores que dominan el mercado actual y el desarrollo tecnológico orientado al libre uso de los recursos.

**BIBLIOGRAFÍA**


#include <Ethernet.h>
#include <SPI.h>

byte mac[] = { 0xFA, 0xBA, 0xDA, 0xFA, 0xBA, 0xDA }; // MAC asignada al Arduino
byte ip[] = { 192, 168, 1, 20 }; // IP Asignada al Arduino
byte server1[] = { 192, 168, 1, 8 }; // IP del servidor PBX Asterisk
byte gateway[] = {192,168,1,1}; //IP del router
byte subnet[] = {255,255,255,0}; //Mascara de la red

EthernetServer server = EthernetServer(23); //Puerto por el que escuchamos al Asterisk
EthernetClient client;

void setup(){
  Ethernet.begin(mac, ip); //Se inicia la IP y MAC
  Serial.begin(9600) //Se inicia el puerto de serie del Arduino a 9600
  pinMode(4,OUTPUT); //Se declara el pin 6 como entrada ( Sensor movimiento)
  pinMode(6,INPUT); //Se declara el pin 6 como entrada ( Sensor movimiento)
  pinMode(5,INPUT); // Se declara el pin 5 como entrada ( Sensor magnético)
  delay(500); // Se da un retardo de 500 milisegundos
}

void loop(){
  int magnetico = 0; // Se declara una variable magnético
  int movimiento =0; // Se declara una variable movimiento
  magnetico = digitalRead(5); // La lectura del pin 5 se guardara en la variable magnético
movimiento = digitalRead(6);                  // La lectura del pin 6 se guardara en la variable movimiento

EthernetClient client = server.available();

if(magnetico == HIGH && movimiento == HIGH){   //Cuando el magnético se active generará la conexión
    if (client.connect(server1, 80)) {        //La conexión Arduino – Asterisk se realiza por el Puerto 80
        client.println("GET /arduino_call.php?sensor=Magnetico HTTP/1.0");
        //En la anterior línea se envía la petición de ubicar el archivo arduino_cal dentro de su servidor web, que es el configurado justo antes, y que envié la palabra magnético que se visualizará en el softphone.
        client.println();
        delay(1000);
    } else {
        Serial.println("connection failed");
    }
} else {
    Serial.println("connecting...");
    Serial.println(Ethernet.localIP());
    if (client.connect(server1, 80)) {
        Serial.println("connected");
        Serial.println("LLamando...");
        client.println("GET /arduino_call.php?sensor=Movimiento HTTP/1.0");
        Serial.println("LLamando Establecida...");
        client.println();
        delay(1000);
    } else {
        Serial.println("connection failed");
    }
}
if(magnetico == HIGH && movimiento == LOW) {

    Serial.println("connecting...");
    Serial.println(Ethernet.localIP());

    if (client.connect(server1, 80)) {
        Serial.println("connected");
        Serial.println("LLamando...");
        client.println("GET /arduino_call.php?sensor=MagneticoyMovimiento HTTP/1.0");
        Serial.println("LLamanda Establecida...");
        client.println();
        delay(1000);
    } else {
        Serial.println("connection failed");
    }
}

if (client) {
    x = client.read(); //Leemos los datos que vienen
    Serial.print("conectado"); //Indicamos por el puerto de serie que tenemos conexión
    Serial.println(x); //Mostramos lo que hemos recibido por TCP/IP
}

if (x == 5) {
    Serial.print(x); //Mostramos lo que hemos recibido por TCP/IP
    Serial.println("activo"); //Informamos por el puerto de serie que activamos el relé
    digitalWrite(4, 1); //Activamos la patilla número 4 del Arduino poniéndola a 5V
    delay (300); //Esperamos 0,3 segundos
digitalWrite(4, 0);  //Desactivamos la patilla número 4 del Arduino poniéndola a 0V
delay (300);         //Esperamos 0,3 segundos
digitalWrite(4, 1);  //Activamos la patilla número 4 del Arduino poniéndola a 5V
delay (300);         //Esperamos 0,3 segundos
digitalWrite(4, 0);  //Desactivamos la patilla número 4 del Arduino poniéndola a 0V
x = 0;               
}