Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

# Introducción de aspectos de seguridad en la Internet de las cosas en el ámbito de las viviendas inteligentes

Trabajo Fin de Grado

**Grado en Ingeniería Informática**

**Autor**: Alberto Ballestín Pérez

**Tutor**: Joan Josep Fons Cors

2015-2016

# Resumen

*Introducción:* En la era de la información en que nos encontramos, existen múltiples problemas de seguridad en el ámbito del IoT. Nosotros abordamos una solución centrada en un dominio concreto de las viviendas inteligentes.

*Desarrollo:* Realizando un diseño e implementación para asignar permisos con los que controlar objetos ofrecidos por la IoT. Desarrollando un framework de seguridad con mecanismos para autenticar a los usuarios y autorizarlos para utilizar los servicios de manera segura y realizando un prototipo para simular una vivienda inteligente. Para ello se ha creado un directorio propio mediante el protocolo LDAP, con unidades organizativas, grupos y usuarios (servidor phpLDAPadmin) y se ha creado una aplicación web en JSP con un servidor CAS de autenticación.

*Conclusiones:* Se ha conseguido dar una capa de seguridad en el proceso de autenticar a los usuarios y lograr una plataforma de mayor seguridad.

**Palabras clave:** Seguridad, Internet de las Cosas, Viviendas Inteligentes, Computación Ubicua, REST

# Abstract

*Introduction:* In the age of information in which we are, multiple problems of safety are in the area of IoT. We approach a solution focused on a specific domain of intelligent houses.

*Development::* Making a design and a implementation to assign permissions with which it controls objects offered by the IoT. Developing a safety framework, with mechanisms to authenticate the users and to authorize then to use the services in a sure way, and realizing a prototype to simulate, an intelligent housing. For it, as oun directory has been created by the protocol LDAP, with organizational units, groups and users (phpLDAPadmin server) and it has created a web application in JSP, with a server CAS of authentication.

*Conclusions:* A certainy layer has been given to the authentication process to users, and to achieve a greater security platform.

**Keywords :** Security, Internet of Things, Smart Homes, Ubiquitous Computing, REST

# Tabla de contenidos

# Índice de figuras

# Índice de tablas

# 1. Introducción

## 1.1. Objetivo

El principal objetivo es realizar un trabajo desde el inicio añadiendo una capa de seguridad para el IoT en el ámbito de las viviendas inteligentes. Los usuarios se identificarán en el servidor CAS, autorizándolos dependiendo de los privilegios que tengan y los grupos a los que pertenezcan. Para ello se ha desarrollado una página web que simulará una aplicación instalada en un dispositivo de una vivienda inteligente, y se usará un método seguro de autenticación.

## 1.2. Motivación.

En lo últimos años el Internet de las Cosas ha evolucionado considerablemente y está cambiando el mercado para hacer la vida más fácil a los usuarios. Como ejemplo de aplicaciones se encuentra el frigorífico inteligente, que cuando detecta que falta un producto o se aproxima la fecha de caducidad, el sistema realiza un pedido automático al supermercado más cercano (Edriel, 2015).

La principal motivación para desarrollar este proyecto es tratar de dotar de una capa de seguridad al sistema. Sin embargo, un sistema informático nunca es infalible y será necesario dotarlo de más medidas de seguridad para protegerlo, como por ejemplo comunicaciones cifradas, NIDS, Firewall, etc., que quedarán fuera del alcance de este TFG.

Otras motivaciones posibles son:

- Un servicio de limpieza, el cual solo se le permite el acceso a la vivienda un día determinado y en un rango de horas concreto. Del mismo modo ocurre con un servicio de mantenimiento.
- Controlar los permisos de los hijos en una familia para que no puedan encender el aire acondicionado, encender el horno o abrir el frigorífico sin autorización.

En definitiva, controlar el acceso a la vivienda y restringir las acciones a realizar dentro de la misma.

## 1.3. Problemática.

Actualmente, existen innumerables problemas en cuanto a la seguridad en el Internet de las Cosas:

- **Invasión de la privacidad o vigilancia ilegal en dispositivos conectados a internet como**: cámaras, coches, medicina, juguetes, etc. Ahí cualquier usuario malintencionado podría utilizar un dispositivo IoT de un vehículo para ver donde se encuentra, a qué velocidad circula o incluso modificar su dirección en caso de que fuera posible y llegar a producir un accidente.

- **Seguridad en la red y los datos empresariales:** muchas empresas conectan sus sistemas de información a la red y sin darse cuenta están adquiriendo una dependencia tecnológica. El simple hecho de estar en red supone un riesgo para la organización pudiendo tener una puerta trasera explotada por un tercero, el cuál podría filtrar datos de la organización o realizar un ataque de denegación de servicio (DoS).

- **Privilegios en las aplicaciones:** En una aplicación de domótica de una vivienda el servicio de limpieza está contratado los miércoles de 10:00 a 12:00, por lo que no puede entrar a la vivienda en una hora que no sea la establecida, por lo cual se le debería negar el acceso cuando intentará entrar fuera de horas.

No existe una manera perfecta de gestionar todos los dispositivos que están integrados con el IoT ya que la mayoría de los dispositivos utilizan diferentes estándares para comunicarse. Por lo que es un gran problema comunicarse de manera segura y eficiente, y se debería crear una API estandarizada para unificar todas las comunicaciones.

Para proteger la infraestructura se tendría que instalar un firewall en la parte exterior de nuestra red, para bloquear los intentos de intrusión no deseados. También se podría usar un sistema de detección de intrusos en la red (NIDS), el cual detectaría ataques como múltiples acceso de un mismo usuario en un corto espacio de tiempo, o también, anomalías que detecten una posible intrusión creando automáticamente una alerta y avisando a la comisaría de policía más cercana.

## 1.4. Idea global propuesta.

El desarrollo del proyecto se ha dividido en cinco fases distintas:

- Fase 1: Estudio sobre las diferentes aplicaciones y tecnología utilizadas en el mercado sobre el Internet de las cosas en una vivienda, en la cual, se ha observado que teniendo acceso a la aplicación, se obtenía el control total sobre todas las acciones sobre los distintos dispositivos instalados, es decir, consultar el estado de una puerta o modificar dicho estado. Por lo cual se ha investigado sobre cómo autenticar y autorizar a las personas por ejemplo dependiendo del tipo de usuario.

- Fase 2: Simulación de una aplicación relacionada con una vivienda sobre el IoT, para ello se ha creado una página web con distintos recursos.

- Fase 3: Creación de la estructura del directorio de nuestra vivienda con grupos, usuarios, contraseñas, emails. Para el desarrollo de este apartado se ha optado por utilizar phpLDAPadmin (una tecnología web para gestionar grupos y usuarios).

- Fase 4: Creación de un servicio central de autenticación (CAS) para aislar la aplicación del módulo de la autenticación y conseguir más seguridad.

- Fase 5: Unificación de los módulos anteriores para que el usuario no se de cuenta de que hay más módulos detrás trabajando.

# 2. Estado del Arte y Contexto Tecnológico

## 2.1. Estado del Arte

Actualmente en el mercado existen pocas aplicaciones o módulos para gestionar la seguridad del IoT ya que es una tecnología que se encuentra en plena expansión. Entre las aplicaciones de IoT similares hemos encontrado Arxan con un módulo para proteger IoT. Y otra aplicación es TaHoma Home Control desarrollada por Somfy, cuya función es gestionar el IoT que se encuentra en el hogar como persianas, televisión, lavadora, puertas e iluminación, la cual, para autenticarse únicamente dispone de una contraseña.

### 2.1.1. Arxan

Arxan Technologies (Technologies, 2015) es una empresa privada fundada en 2001 para ofrecer protección de aplicaciones y soluciones anti sabotajes, su sede principal se encuentra en EEUU y tiene múltiples sedes distribuidas por todo el mundo (Reino Unido, Francia, Alemania, Suecia, Japón y Corea).

Arxan ofrece soluciones de software para dispositivos móviles, portátiles, servidores y plataformas relacionadas con el IoT. Protege más de 400 millones de dispositivos en el ámbito de las industrias, incluyendo: servicios financieros, proveedores de software independientes, manufacturas, sanidad, medios digitales, juegos y otros (fuente: *www.arxan.com*).

También protege aplicaciones trabajando en el ámbito del IoT en numerosas vías como aplicaciones móviles que controlan en IoT desde el dispositivo, firmware IoT y aplicaciones abiertas en plataformas de IoT.

Más concretamente ofrece soluciones contra los siguientes riesgos:

- **Operaciones inseguras:** Las aplicaciones o dispositivos IoT son susceptibles de modificaciones por código malicioso, sin pasar por los controles y la alteración de la integridad de los datos.
- **Exposición o pérdida de información:** Aplicaciones o dispositivos IoT pueden revelar información privada protegida como claves y credenciales.
- **Robo de la propiedad intelectual**: Aplicaciones o dispositivos IoT desprotegidos podrían ser analizados, robados o pirateados.
- **Exposición a vulnerabilidades desconocidas:** Aplicar parches a dispositivos IoT es un reto para prevenir la exposición a vulnerabilidades desconocidas, se recomienda ocultar el código de explotación para evitar que los hacker lo analicen o puedan realizar ingeniería inversa.
- **Protegiendo la seguridad de los componentes:** Muchos proveedores de IoT han establecido módulos de seguridad comunes para proteger sus aplicaciones. Arxan protege la lógica y las bibliotecas con seguridad propia.

Sistemas que protege Arxan:

- Coches conectados.
- Dispositivos portátiles conectados.
- Hogares conectados.
- Ciudades conectadas e internet industrial.

### 2.1.2. Gemalto

Gemalto es una empresa vendedora de tarjetas inteligentes y productos de seguridad digital con su sede central en Amsterdam, tiene más de 11.000 empleados distribuidos por 120 países, se creó a raíz de la fusión entre Axalto y Gemplus International SA (Gemalto, 2015).

Están asociados con fabricantes industriales, de dispositivos originales, con operadores de redes comerciales donde la conectividad, los datos, la nube y la seguridad de los servicios son esenciales para el éxito .

Tiene soluciones para proteger de forma segura objetos inteligentes para que las empresas, los gobiernos y los consumidores los puedan aprovechar. Además, de ofrecer hardware robustos, plataformas seguras y soluciones para conectar y proteger cualquier dispositivo M2M (Máquina a Máquina) o de electrónica de consumo, explicadas a continuación:

- Electrónica de consumo utilizada para añadir funciones y servicios innovadores y seguros para diferenciar sus productos, desde la autenticación y los pagos hasta los servicios empresariales y la conectividad.
- Máquina a Máquina optimizando mediante una conexión segura los activos de su negocio con módulos M2M de Cinterion, los módulos MIM y la plataforma SensorLogic.
- Conectividad del IoT, es una fuente única para la gestión segura del ciclo de vida de las suscripciones, adaptada a las necesidades de los operadores o fabricantes de los dispositivos.

Gemalto no ofrece una solución directamente con el IoT relacionado con la vivienda pero sí que brinda soluciones similares para proteger dispositivos y módulos M2M entre otros.

## 2.2. Contexto Tecnológico

En el siguiente bloque se describen los tipos de lenguajes, tecnologías utilizadas y entornos de programación que se han utilizado para el desarrollo del framework, todo esto sin adentrarnos en detalle de su implementación

### 2.2.1. Tecnologías utilizadas en el proyecto

#### 2.2.1.1. VirtualBox


Oracle VM VirtualBox es un software de virtualización para arquitecturas x86 y amd64, los creadores originales fueron una empresa alemana Innotek GmBh aunque actualmente es desarrollado y mantenido por Oracle Corporation. Gracias a VirtualBox se pueden instalar múltiples sistemas operativos conocidos como "sistemas invitados" dentro de otro sistema operativo "anfitrión" cada uno con su espacio virtualizado.

Estando en modo anfitrión soporta los sistemas GNU/Linux, Mac OS X, OS/2 Warp, Microsoft Windows y Solaris. Dentro de estos sistemas es posible virtualizar muchos otros como Windows, MS-DOS, FreeBSD, GNU/Linux y OpenBSD.

#### 2.2.1.2. Sistema operativo: Debian


Debian es una comunidad conformada por desarrolladores y usuarios, que mantiene un sistema operativo GNU basado en software libre. Es mantenida por sus usuarios y tiene una instalación sencilla, actualizaciones fáciles, soporta múltiples arquitecturas y kernels, el sistema de seguimiento de errores de debían es público, es estable, rápido, ligero en memoria y tiene buena seguridad en el sistema y dispone de software de seguridad

#### 2.2.1.3. PhpLDAPadmin


Es una herramienta para la administración de servidores LDAP realizada en PHP con una interfaz Web. Trabaja en varias plataformas, la cual permite acceder a información guardada de forma centralizada en una red desde cualquier parte de internet usando un navegador Web. Actualmente se encuentra disponible bajo licencia pública.

La estructura de almacenamiento de LDAP es el servicio de directorio, el cual permite acceder a información guardada donde se administra la unidad organizativa, los grupos y usuarios. También se puede dotar a los usuarios de una contraseña en formato hash, es decir, la contraseña cifrada. A diferencia de las base de datos clásicas, no contiene soporte para transacciones o funcionalidad de vuelta atrás (rollback).

Los servicios de directorios pueden ser replicados fácilmente entre distintos ordenadores incrementando la disponibilidad y la fiabilidad, además, se permiten inconsistencias temporales entre las réplicas.

Las ventajas de LDAP son:

- La sencillez de administración.
- Centralización de la información.
- Posibilidad de utilizar protocolos seguros.

Para más información visitaremos la web oficial http://phpldapadmin.sourceforge.net

#### 2.2.1.4. Apache Tomcat


Tomcat funciona como un contenedor de servlets, es decir, un programa capaz de recibir peticiones de páginas web y redireccionar estas peticiones a un objeto servlet, programa que implementa las especificaciones de los servlets y de JavaServer (JSP) de Oracle.

Se ha utilizado la versión 8.0.27 para desarrollar el proyecto, apache ha sido desarrollado por "Apache Software Foundation" y se puede descargar en http://apache.org

#### 2.2.1.5. Variable de entorno

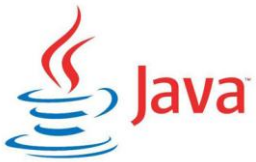
Hay que tener en cuenta que Tomcat depende de java por lo cual buscará la variable de sistema JAVA_HOME que le indica donde está la instalación de java. Para ello se ha descargado la versión 1.8.0_60 de 64 bits y se ha configurado en la máquina DEBIAN.

#### 2.2.1.6. JavaServer Pages (JSP)

JSP es una tecnología que ayuda a los desarrolladores a crear páginas web dinámicas basadas en HTML, XML. JSP es muy similar a PHP, aunque JSP utiliza como lenguaje de programación Java. Para que desplegar y hacer funcionar JSP se requiere de un servidor web compatible con contenedores servlet como el que se ha utilizado de Apache Tomcat.

Una de las ventajas de JSP es que hereda la portabilidad de Java, y puede ejecutar las aplicaciones en diferentes plataformas sin cambios.

#### 2.2.1.7. Java




Java es un lenguaje de programación de propósito general, concurrente y orientado a objetos. Es desarrollado por la compañía Sun Microsystems enfocado siembre a cubrir las necesidades tecnológicas más importantes. Su principal característica es su portabilidad, es un lenguaje independiente de la plataforma en que se ejecute. Utilizando java podemos programar páginas web dinámicas con cualquier tipo de conexión de red entre cualquier sistema.

#### 2.2.1.8. Shell scripts

Es un programa informático que interpreta órdenes en este proyecto ha sido utilizada para automatizar ordenes como control de procesos, modificación de permisos en ficheros, arranque del servicio Apache Tomcat. Hay varias implementaciones del lenguajes como bash, ksh, zsh, ash. A pesar de ello se ha elegido escribir el programa o script con formato .sh.

#### 2.2.1.9. Spring Security


Spring Security es uno de los framework de seguridad J2EE más populares y completos usado por gran parte de instituciones, universidades y empresas. Con Spring Security se consigue incorporar una infraestructura básica en una aplicación ya implementada sin necesidad de escribir líneas de código gracias a su naturaleza no invasiva. Además, incluye componentes ya implantados que permiten la integración con sistemas externos con gran facilidad. En sus inicios se basó en el proyecto Acegi Security, que con los años consiguió convertirse en un producto maduro y robusto, actualmente, forma parte de la familia de productos spring.

#### 2.2.1.10. Single Sign-On

Single-sign-on (SSO) es un procedimiento de autenticación que habilita al usuario para acceder a varios sistemas con una sola instancia de identificación. Se podría resumir como una autenticación o validación única.

#### 2.2.1.11. Central Authentication Service (CAS)

Para realizar SSO se ha utilizado CAS que permite a aplicaciones web autenticar a los usuarios, sin tener acceso a las credenciales de seguridad de un usuario como su contraseña, en este proyecto las contraseñas se administran en el phpLDAPadmin.

El protocolo CAS involucra al menos a tres partes: un navegador cliente web, la aplicación web que solicita la autenticación y el servidor CAS.

Cuando el usuario quiere visitar una aplicación, la cual requiere autenticación esta redirige al CAS. CAS valida la autenticidad del cliente, mediante la comprobación en la base de datos donde están los usuarios y las contraseñas ya sea en Kerberos, Active Directory o LDAP.

Si la autenticación ha sido correcta, CAS redirige al usuario a la aplicación.

#### 2.2.1.12. RBAC

El control de acceso basado en roles en adelante (RBAC) es una alternativa al modelo superusuario en el que el usuario que accede a la aplicación dispone de todos los permisos sobre la aplicación, además, RBAC permite controlar el acceso de usuarios a tareas que normalmente están restringidas al superusuario. A través de la aplicación de atributos de seguridad a procesos y usuarios, RBAC, permite dividir las capacidades de superusuario entre varios administradores. La gestión de derechos de procesos es implementada a través de privilegios y la gestión de derechos de usuarios a través de RBAC.

#### 2.2.1.13. PHP

PHP es un lenguaje de código abierto muy popular, adecuado para desarrollo web y que puede ser incrustado en HTML. Es popular puesto que un gran número de páginas y portales web están creadas con PHP. Código abierto significa que es de uso libre y gratuito para todos los programadores que quieran usarlo. Incrustado en HTML significa que en un mismo archivo vamos a poder combinar código PHP con código HTML, siguiendo unas reglas.

En el presente trabajo se ha utilizado para la programación del phpLDAPadmin.

### 2.2.2. Tecnologías similares no utilizadas relacionadas con el proyecto

#### 2.2.2.1. Active Directory

Active Directory (AD) es un término utilizado por Microsoft para hacer referencia a la implementación del servicio de directorio en una red distribuida de computadores. Entre los protocolos que utiliza están LDAP, DNS, DHCP y Kerberos.

#### 2.2.2.2. Git

Git es un software para llevar un control de las versiones y tener un sistema de copias de seguridad, fue diseñado por Linus Torvalds, diseñado pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente.

#### 2.2.2.3. Shibboleth

Shibboleth es un proyecto OpenSource desarrollado por Internet2 que implementa un sistema de Single Sign On (SSO), con intercambio de atributos basados en estándares abiertos, principalmente SAML. Además tiene una funcionalidad de privacidad extendida que permite al usuario y/o institución controlar los atributos liberados a cada aplicación.

Este sistema federado provee acceso seguro a través de diferentes dominios de seguridad, preservando la privacidad de los datos de sus usuarios, y posibilita la escalabilidad del sistema a través de relaciones de confianza.

Es utilizado en redes de investigación de Suiza, Finlandia, Alemania, Inglaterra, Hungría, Grecia y otros.

#### 2.2.2.4. Oauth

OAuth (Open Authorization) es un protocolo que permite flujos simples de autorización para sitios web o aplicaciones informáticas. Se trata de un protocolo propuesto por *Blaine Cook* y *Chris Messina*, que permite autorización segura de una API de modo estándar y simple para aplicaciones de escritorio, móviles y web.

#### 2.2.2.5. Mysql

Es un sistema de gestión de bases de datos relacional, multihilo y multiusuario. Usa software libre utilizado en aplicaciones web, como Joomla, Wordpress, Drupal o phpBB, en múltiples plataformas.

#### 2.2.2.6. PhpMyAdmin

PhpMyAdmin es una herramienta escrita en PHP con la intención de manejar la administración de MySQL a través de páginas web, utilizando Internet. Actualmente puede crear y eliminar Bases de Datos, crear, eliminar y alterar tablas, borrar, editar y añadir campos, ejecutar cualquier sentencia SQL, administrar claves en campos, administrar privilegios, exportar datos en varios formatos y está disponible en 72 idiomas. Se encuentra disponible bajo la licencia GPL.

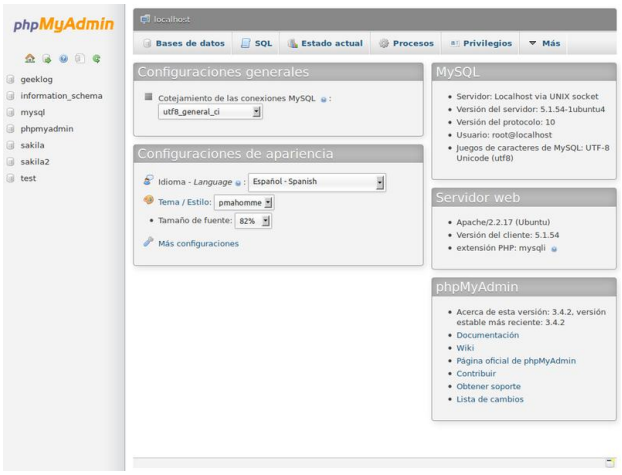



*Figura 1: Pantalla principal de phpMyAdmin*

# 3. Propuesta: Análisis

## 3.1. Qué es IoT

Internet de las cosas hace referencia a sensores y actuadores. Un sensor es un dispositivo que capta magnitudes físicas como variaciones de luz, temperatura, sonido, pulsaciones, nivel, presión, posición de estado sólido, etc. Por otra lado un actuador es un dispositivo capaz de transformar la energía hidráulica, neumática o eléctrica en la activación de un proceso con la finalidad de generar un efecto sobre un proceso automatizado, es decir, un actuador recibe una orden según su programación y actúa a consecuencia (Conner, 2010).

La IoT a grandes rasgos se define como una red de objetos físicos interconectados entre sí valiéndose de internet.

## 3.2. Historia

El origen del internet de las cosas según *postcapes,* se remonta al 1832 cuando se creó el primer telégrafo electromagnético por el barón Schilling en Rusia y en 1833 Carl Friedrich Gauss y Wilhelm Weber inventaron su propio código para comunicarse a una distancia de 1200 metros en Göttingen, Alemania. En 1844, Samuel Morse, envió el primer código morse en un mensaje telegráfico público desde Washington, DC a Baltimore, a una distancia de unos 60 km.

En el año 1926, Nicola Tesla, explicó en un artículo en la revista Colliers: “que se podrá controlar todas las cosas que son partículas y los instrumentos que nos rodean”, esto se compara hoy en día con un smartphone. Con el trabajo de Nikola Tesla se conformaron la base de las comunicaciones inalámbricas y la radio.

En 1950 Alan Turing escribió un artículo en el Oxford Mente Diario de Computing Machinery e Inteligencia, en el cual explicaba cómo se puede proporcionar a una máquina sensores para enseñarla a entender y comunicarse, este proceso es similar a la enseñanza de un niño. En 1969 se envió el primer mensaje utilizando ARPANET, una red operativa

originaria de la Internet global. Diez años después se probó TCP/IP, protocolo en el que actualmente se basa internet, el cual permite la transmisión de datos entre computadoras.

Aproximadamente, en el año 1990 Berners-Lee estableció la primera comunicación con éxito entre un cliente Hypertext Transfer Protocol (HTTP) y un servidor utilizando internet, dando lugar a la World Wide Web (WWW). Un año más tarde, Berners-Lee creó la primera página web. Este logro fue un punto de inflexión para la evolución de internet ya que eso propició una revolución con ese desarrollo tecnológico.

La primera vez que se habló del concepto Internet de las Cosas (Internet off things "IoT") fue en 1999 cuando Kevin Ashton impartió una conferencia en Procter & Gamble. Desde que se inició el nuevo siglo se han creado publicaciones en The Guardian, Scientific American y the Boston Globe. También se empezó a desplegar la tecnología RFID, una tecnología de identificación mediante radiofrecuencia, desplegada por el Departamento de Defensa de los EEUU y los almacenes Walmart a nivel comercial.

En el año 2005 tuvo la aparición Arduino, una plataforma Hardware de código abierto, en la que cualquiera puede aportar sus ideas o sus logros, simplemente es una placa sencilla con entradas y salidas sin interfaz visual. Es un dispositivo mediante el cual podemos conectar el mundo físico con el mundo virtual, o el mundo analógico con el digital.

En el 2006 se comercializo Nabaztag fabricado por la empresa Violet de origen francés. Se trata de un pequeño conejo conectado a Internet por wifi. El cual consulta la información en internet para después difundirla como información meteorológica, la bolsa, el estado de la circulación, el CO2 de la atmósfera, todo esto emitiendo mensajes vocales, luminosos o moviendo las orejas.

En el 2008 un grupo de empresas se unen formando el IPSO Alliance, una organización con el objetivo de promover el protocolo de Internet en redes de objetos y darle un impulso al IoT. En la actualidad en el IPSO Alliance participan empresas de todo el mundo como Motorola, Google, Bosch, Cisco, Toshiba o Fujitsu. En el mismo año comenzó el proyecto Pachube, se trata de una plataforma web construida para gestionar datos generados por todo el mundo en tiempo real, es decir, una herramienta que ofrece a cualquier persona poder

compartir, colaborar y hacer uso de la información para generar aplicaciones IoT. En julio de 2011 la plataforma fue adquirida por LogMeIn, un proveedor de servicios basado en la Nube.

En el año 2011 se lanzó el protocolo IPv6, ya que estaban acabando las IPs disponibles con IP v4 de $2^{32}$ direcciones de host diferentes, un número inadecuado para dar una dirección IP a cada persona del planeta, con IPv6 dieron la posibilidad de tener $2^{128}$ direcciones IP, lo cual para el IoT supone un logro para poder conectar cualquier dispositivo a internet

## 3.3. Estado actual.

En la actualidad Internet of things (IoT) está provocando estragos en el día a día de los ciudadanos. Un informe publicado por www.computerworld.es titulado "internet of things research study" afirmó que el 70% de los dispositivos relacionados con IoT, podrían estar afectados con vulnerabilidades debidas a algoritmos de cifrado, permisos en ficheros o carpetas y problemas relacionados con la debilidad en sus passwords.

El ecosistema del IoT está muy fragmentado debido a los distintos protocolos y plataformas de comunicación que utilizan las máquinas, para solucionar esto empresas importantes del sector como Cisco, IBM e Intel están firmando alianzas para ayudar a minimizar los riesgos de integración de aplicaciones IoT y avanzar todos en el mismo sentido definiendo estándares.

Los principales riesgos del IoT son:

- **Falta de una infraestructura compartida:** Cada dispositivo IoT actualmente está instalado en su plataforma y ecosistema.
- **Falta de estándares:** Actualmente no existe una organización mundial para estandarizar protocolos, hardware y software en el internet de las cosas.
- **Duración de la batería.** En la actualidad, la batería para los dispositivos es una lacra ya que estos tienen una vida útil limitada. Se debería aumentar la duración de la batería utilizando fuentes de energía existentes o buscando nuevas.
- **Control de los datos.** Se debe controlar quien accede a los datos, esto se ha implementado en el proyecto.

- **Intercambio de información.** En el IoT los datos son lo más valioso. Hay empresas que ofrecen servicios gratuitos a cambio de tus datos, los cuales se venden a terceros con el fin de desarrollar otros productos o servicios útiles para consumidores.

Lo que hoy en día se busca es el acceso a la información desde cualquier sitio en cualquier momento como:

- Acceso directo a servicios.
- Redes inalámbricas.
- Teletrabajo.
- Dispositivos Móviles.
- BYOD.
- Servicios en la nube.
- Convergencia ToIP.
- Conexión de sistemas industriales.
- Internet de las cosas (IoT, IoE, IIoT).

## 3.4. Propuesta

En el presente trabajo, inicialmente se ha realizado un estudio de aplicaciones existentes realizadas para llevar un control sobre el IoT, en la que se ha descubierto que cualquier usuario con privilegios para gestionar la aplicación podía realizar cualquier acción, es decir, tenía un control total. Para no permitir que cualquier usuario tenga control total sobre los objetos se han asignado a cada usuario o grupos permisos para que según sus privilegios puedan realizar o no acciones sobre las cosas

# 4. Diseño e implementación

## 4.1. Estructura del directorio

En los siguientes apartados se ha procedido a explicar los pasos y programas necesarios más relevantes utilizados para desarrollar el proyecto

### 4.1.1. Instalación virtualbox

Se ha procedido a instalar virtualbox concretamente la versión 5.0.2 r102062 sobre una máquina Linux para virtualizar el sistema operativo Debian, también se ha utilizado para llevar un control de versiones al crear checkpoints una vez se van consiguiendo objetivos. Para ello se realizan clonaciones de la máquina virtual completa.

### 4.1.2. Instalación Debian

En el siguiente paso se ha realizado una instalación limpia con una máquina Debian con la versión 8.2.0 de 64 bits sobre virtualbox descargada de www.debian.org. Tanto esta fase como la anterior se ha decidido no entrar en profundidad  explicando los pasos llevados para su instalación y configuración ya que no es el objetivo principal del proyecto.

### 4.1.3. Política de contraseñas

Se recomienda configurar una contraseña diferente para el acceso al sistema, estas contraseñas se considera deben cumplir con las siguientes directrices:

- Longitud mínima de 12 caracteres.
- Compuesta por letras, números y caracteres no alfanuméricos.
- Caducidad máxima de 3 meses. Dependiendo del tipo de activo considerado como críticos podría definirse una caducidad de 1 mes.

- La política debe impedir repetir las últimas 12 contraseñas.
- Evitar la reutilización de contraseñas.

### 4.1.4. Bastionado del sistema

Después de haber instalado el sistema, se tiene una instalación limpia, la cual no quiere decir que ya tenemos un sistema seguro, para ello se va a proceder a bastionar nuestro sistema para evitar que este pueda ser atacado por un tercero, para tal fin se van a realizar los siguientes pasos:

- Introducir una contraseña en la BIOS.
- Comprobar que están creadas las particiones lógicas.

```
#!/bin/bash
#Verificar que están todas las particiones instaladas

outputroot=$(df -k | grep -w /)
outputusr=$(df -k | grep /usr)
outputtmp=$(df -k | grep /tmp)
outputvar=$(df -k | grep /var)
outputhome=$(df -k | grep /home)
outputboot=$(df -k | grep /boot)

if [ "$outputroot" != "" ]; then
        echo -e "El sistema cuenta con la particion /"
else
        echo -e "El sistema no cuenta con la particion /"
fi

if [ "$outputusr" != "" ]; then
      echo -e "El sistema cuenta con la particion /usr"
else
```

```
        echo -e "El sistema no cuenta con la particion /usr"
fi

if [ "$outputtmp" != "" ]; then
      echo -e "El sistema cuenta con la particion /tmp"
else
        echo -e "El sistema no cuenta con la particion /tmp"
fi

if [ "$outputvar" != "" ]; then
      echo -e "El sistema cuenta con la particion /var"
else
        echo -e "El sistema no cuenta con la particion /var"
fi

if [ "$outputhome" != "" ]; then
      echo -e "El sistema cuenta con la particion /home"
else
        echo -e "El sistema no cuenta con la particion /home"
fi

if [ "$outputboot" != "" ]; then
      echo -e "El sistema cuenta con la particion /boot"
else
        echo -e "El sistema no cuenta con la particion /boot"
fi
```

*Tabla 1: Script para comprobar si el sistema esta bastionado*

- Asignar al equipo una dirección fija.
- Restringir los privilegios.
    - Restricción de logins desde “root”.
    - Restringir usuarios al grupo de administradores.

- Introducir fecha de caducidad a las contraseñas.
- Comprobar los permios de ficheros passwd, group, shadow y gshadow, para ello se ha creado el siguiente script para verificar si tiene los permisos correctos.

```
#!/bin/bash
#
#Comprobar permisos de ficheros passwd, group, shadow y gshadow

#Permisos que tiene que tener passwd y group 644
PERPASSWDGROUP=-rw-r--r--
#Permisos que tiene que tener shadow y gshadow 400
PERSHADOWS=-r--------

PERPASSWD=$(ls -l /etc/passwd | awk -F '.'1 '{ print $1'})
PERGROUP=$(ls -l /etc/group | awk -F'.'1 '{ print $1'})
PERSHADOW=$(ls -l /etc/shadow | awk -F'.'1 '{ print $1'})
PERGSHADOW=$(ls -l /etc/gshadow | awk -F'.'1 '{ print $1'})

if [ "$PERPASSWD" = "$PERPASSWDGROUP" ]; then
        echo -e "Permisos del fichero /etc/passwd correctos"
else
        echo -e "Permisos del fichero /etc/passwd incorrectos"
fi

if [ "$PERGROUP" = "$PERPASSWDGROUP" ]; then
        echo -e "Permisos del fichero /etc/group correctos"
else
        echo -e "Permisos del fichero /etc/group incorrectos"
fi

if [ "$PERSHADOW" = "$PERSHADOWS" ]; then
```

```
        echo -e "Permisos del fichero /etc/shadow correctos"
else
        echo -e "Permisos del fichero /etc/shadow incorrectos"
fi

if [ "$PERGSHADOW" = "$PERSHADOWS" ]; then
        echo -e "Permisos del fichero /etc/gshadow correctos"
else
        echo -e "Permisos del fichero /etc/gshadow incorrectos"
fi
```

*Tabla 2: Script para controlar permisos*

En el caso de que los permisos no estén correctamente establecidos los podemos modificar con el siguiente script.

```
#!/bin/bash
#
#Este paso no tiene restore
if ! restoring
then
        #Asegura los permisos de archivos esenciales
        chown root:root /etc/{passwd,shadow,group,gshadow}
        result=$?
        chmod 644 /etc/{passwd,group}
        resultado $? "Establecer los permisos de /etc/passwd y /etc/group"
        chmod 400 /etc/{shadow,gshadow}
        resultado $? "Establecer los permisos de /etc/shadow y /etc/gshadow"
fi
```

*Tabla 3: Script para establecer permisos*

- Modificar el Banner
- Determinar qué servicios son iniciados en el arranque del sistema

```
#!/bin/bash
#Comprobar que servicios hay arrancados en el sistema

echo -e "\n"
servicios=$(chkconfig --list | grep ':activo\|:on' | awk '{print $1}')
echo "Servicios iniciados en el arranque del Sistema"
echo -e "-----------------------------------------------------\n"
result=""
regex="^Habilitar"

for i in $servicios
do
  data=$(cat scripts/lista.txt | grep $i)
  if [ "$data" = "" ]; then
    result=$result":$i:A determinar por el usuario\n"
  else
    read recom < <(awk -v var="$data" 'BEGIN{split(var,aux,":");print aux[2]}')
    if [[ $recom =~ $regex ]]; then
      if [ ${#recom} = "10" ]; then
        result=$result":$data\n"
      fi
    else
      result=$result":$data\n"
    fi
  fi
done
echo -e $result | column -s ":" -t
echo -e "\n"
```

*Tabla 4: Script que muestra los servicios arrancados en el sistema*

Se adjunta el archivo lista.txt con los servicios que hay que habilitar o deshabilitar si es posible.

acpid: Habilitar
anacron: Deshabilitar si es posible
apmd: Deshabilitar si es posible
atd: Configurar
auditd: Configurar
autofs: Deshabilitar si es posible
avahi-daemon: Deshabilitar si es posible
bluetooth: Deshabilitar si es posible
cpuspeed: Habilitar
crond: Configurar
cups: Deshabilitar si es posible
firstboot: Deshabilitar si es posible
gpm: Deshabilitar si es posible
haldaemon: Deshabilitar si es posible
hidd: Deshabilitar si es posible
hplip: Deshabilitar si es posible
ip6tables: Configurar
iptables: Configurar
irqbalance: Habilitar
isdn: Deshabilitar si es posible
kdump: Deshabilitar si es posible
kudzu: Deshabilitar si es posible
mcstrans: Deshabilitar si es posible
mdmonitor: Deshabilitar si es posible
messagebus: Deshabilitar si es posible
microcode_ctl: Deshabilitar si es posible
netfs: Deshabilitar si es posible
network: Habilitar
nfslock: Deshabilitar si es posible
pcscd: Deshabilitar si es posible
portmap: Deshabilitar si es posible

| |
|---|
| readahead_early: Deshabilitar si es posible |
| readahead_later: Deshabilitar si es posible |
| restorecond: Habilitar |
| rhnsd: Deshabilitar si es posible |
| rpcgssd: Deshabilitar si es posible |
| rpcidmapd: Deshabilitar si es posible |
| sendmail: Configurar |
| setroubleshoot: Deshabilitar si es posible |
| smartd: Habilitar |
| sshd: Habilitar solo en Servidores |
| syslog: Configurar |
| xfs: Deshabilitar si es posible |
| yum-updatesd: Deshabilitar si es posible |

*Tabla 5: Fichero lista.txt*

### 4.1.5. Estructura de la organización

En este apartado se describe como se ha formado la estructura del directorio de nuestra organización a la que se ha llamado *iotsec*, es decir, los trabajadores que la forman, la actividad que realiza cada uno, su nombre de identificador de usuario, su contraseña y su email, el cual mostramos en la (Tabla 6: Estructura de IoTsec).

Usuarios de iotsec

| Trabajador (cn) | Actividad (title) | Identificador de usuario (uid) | Contraseña | E-mail (mail) |
|---|---|---|---|---|
| Juan Sánchez | Administrador | jsanchez | jsanchez15 | jsanchez@iotsec.com |
| Ana Gil | Personal limpieza | agil | agil15 | agil@iotsec.com |
| José López | Personal mantenimiento | jlopez | jlopez15 | jlopez@iotsec.com |
| Carla Sánchez | Hija | csanchez | csanchez15 | csanchez@iotsec.com |
| Andrés García | Invitado | agarcia | agarcia15 | agarcia@iotsec.com |

*Tabla 6: Estructura de IoTsec*

Los roles necesarios para gestionar la aplicación son los siguientes:

| Rol | Descripción |
|---|---|
| Administrador (AD) | Control total sobre la aplicación |
| Personal limpieza (PL) | Limpiar la vivienda |
| Personal Mantenimiento (PM) | Repara y mantiene la estructura física y los equipos de la instalación |
| Familiar (F) | Puede modificar campos y tiene restricciones en otros |
| Invitado (I) | Consulta datos y modificar ciertos campos, también tiene restricciones |

*Tabla 7: Descripción de los roles*

La asignación de roles a cada trabajador es la siguiente:

| Personal vivienda | Actividad | Roles |
|---|---|---|
| Juan Sánchez | Administrador | AD,PL,PM,F,I |
| Ana Gil | Personal limpieza | PL,I |
| José Lopez | Personal mantenimiento | PM,I |
| Carla Sanchez | Familiar | F,I |
| Andres Garcia | Invitado | I |

*Tabla 8: Asignación de los roles*

Los roles asignados a cada grupo dentro de la vivienda para ejecutar las diferentes operaciones de cada módulo son los siguientes:

| Lugar de la vivienda | Acción | Permisos | | | | |
|---|---|---|---|---|---|---|
| | | Administrador | Limpieza | Mantenimiento | Familia | Invitados |
| Entrada | Abrir puerta de casa | Sí | Sí | Sí | Sí | Sí |
| Pasillo | Encender o apagar luces | Sí | Sí | Sí | Sí | Sí |
| | Abrir puerta habitación principal | Sí | Sí | Sí | No | No |
| | Abrir puerta habitación familiares | Sí | Sí | Sí | Sí | No |
| | Abrir puerta cocina | Sí | Sí | Sí | Sí | Sí |
| | Abrir puerta salón | Sí | Sí | Sí | Sí | Sí |
| | Abrir puerta baño | Sí | Sí | Sí | Sí | Sí |

| | | | | | | |
|---|---|---|---|---|---|---|
| Habitación Principal | Encender o apagar luces | Sí | Sí | Sí | Sí | Sí |
| | Consultar estado A/C | Sí | Sí | Sí | Sí | Sí |
| | Modificar estado A/C | Sí | No | No | Sí | No |
| | Subir o bajar persianas | Sí | Sí | Sí | Sí | Sí |
| | Abrir caja fuerte | Sí | No | No | No | No |
| Habitación Familiares | Encender o apagar luces | Sí | Sí | Sí | Sí | Sí |
| | Consultar estado A/C | Sí | Sí | Sí | Sí | Sí |
| | Modificar estado A/C | Sí | No | No | Sí | No |
| | Subir o bajar persianas | Sí | Sí | Sí | Sí | Sí |
| Cocina | Encender o apagar luces | Sí | Sí | Sí | Sí | Sí |
| | Consultar estado A/C | Sí | Sí | Sí | Sí | Sí |
| | Consultar estado cafetera | Sí | Sí | Sí | Sí | Sí |
| | Consultar estado horno | Sí | Sí | Sí | Sí | Sí |

| | | | | | | |
|---|---|---|---|---|---|---|
| | Consultar estado lavadora | Sí | Sí | Sí | Sí | Sí |
| | Consultar estado microondas | Sí | Sí | Sí | Sí | Sí |
| | Consultar estado vitro | Sí | Sí | Sí | Sí | Sí |
| | Modificar estado A/C | Sí | No | Sí | Sí | No |
| | Modificar estado cafetera | Sí | No | No | Sí | No |
| | Modificar estado horno | Sí | No | No | Sí | No |
| | Modificar estado lavadora | Sí | Sí | No | Sí | No |
| | Modificar estado microondas | Sí | No | No | Sí | Sí |
| Baño | Consultar estado A/C | Sí | Sí | Sí | Sí | Sí |
| | Modificar estado A/C | Sí | No | Sí | Sí | No |
| | Encender o apagar luces | Sí | Sí | Sí | Sí | Sí |
| Salón | Consultar estado A/C | Sí | Sí | Sí | Sí | Sí |

| | | | | | | |
|---|---|---|---|---|---|---|
| | Modificar estado A/C | Sí | No | Sí | Sí | No |
| | Encender o apagar luces | Sí | Sí | Sí | Sí | Sí |
| | Subir o bajar persianas | Sí | Sí | Sí | Sí | Sí |
| | Modificar estado TV | Sí | No | No | Sí | No |

*Tabla 9: Roles asignados a cada grupo de usuarios*

A continuación se muestra en alto nivel la estructura del directorio que se ha implementado mediante OpenLDAP:

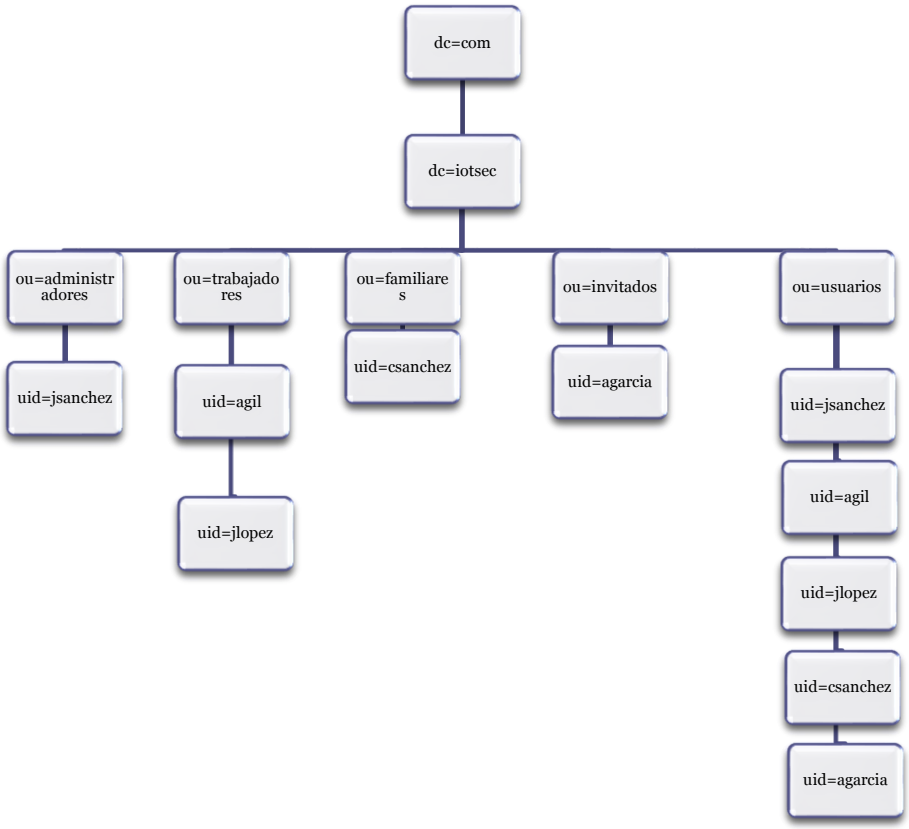



*Figura 2: Estructura del directorio*

Lista de posibles aplicaciones de IoT:

- Iluminación
- Puertas
- Persianas
- Cortinas
- Aire acondicionado y calefacción
- Televisión
- Lavadora
- Frigorífico
- Tostadora
- Microondas
- Vitrocerámica
- Horno
- Sistema de riego
- Inodoro
- Robot de limpieza
- Mantenimiento piscina
- Sistema video vigilancia
- Estación meteorológica
- Ordenador
- Impresora

### 4.1.6. Instalación y configuración phpLDAPadmin

En este paso se describe la instalación y configuración que se ha utilizado para instalar phpLDAPadmin, desde DEBIAN utilizando la terminal se ejecutaran los siguientes comandos.

```
$ sudo apt-get update
$ sudo apt-get install slapd ldap-utils
```

*Tabla 10: Instalación phpLDAPadmin*

Ahora nos indica si queremos continuar, seleccionamos que sí.

En el siguiente paso se indicará la contraseña de administrador de slapd.

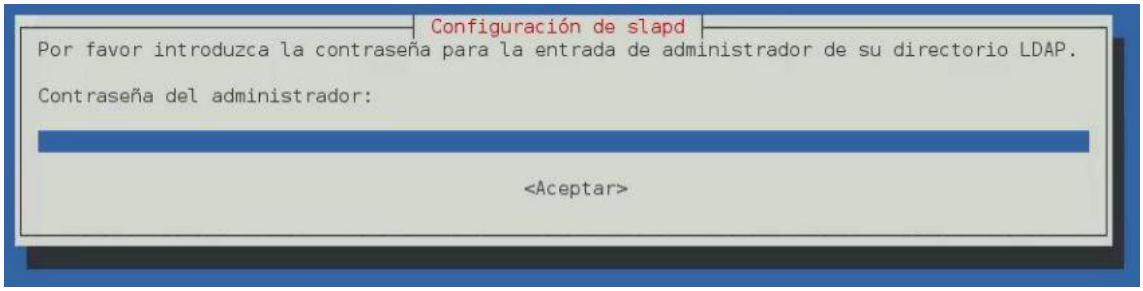



*Figura 3: Configuración de slapd*

```
$ sudo dpkg-reconfigure slapd
```

En las ventanas emergentes que aparecen se introducirá el nombre del dominio DNS, el nombre de la organización , la contraseña de la de administrador, y los demás campos se dejarán con la opción por defecto.

A continuación se instalara el módulo phpLDAPadmin.

```
$ sudo apt-get install phpldapadmin
```

Después de unos segundos esperando para la instalación reiniciaremos el servicio ldap.

```
$ sudo service slapd restart
```

Ahora que ya se ha reiniciado el servicio, se entrará a la interfaz web para comprobar que ésta funciona.


*Figura 4: Menu principal phpLDAPadmin*

En el siguiente archivo se va a editar la configuración para cambiar los valores por defecto por los de nuestra organización.

```
$ sudo gedit /etc/phpLDAPadmin/config.php
```

Se modifican los siguientes campos, dc=example por dc=iotsec como se muestra a continuación para poder acceder al servidor ldap con nuestros datos y nuestra contraseña.

```
$servers->setValue('server','base',array('dc=iotsec,dc=com'));

$servers->setValue('login','bind_id','cn=admin,dc=iotsec,dc=com');
```

*Figura 5: Campos modificados en config.php*

Ahora como se muestra en la siguiente imagen se observa como ya aparece el dc=iotsec correspondiente a nuestra organización y ya se puede entrar con la contraseña.


*Figura 6: Panel de login phpLDAPadmin*

Una vez dentro se observa el Servidor LDAP con la jerarquía inicial.


*Figura 7: Estructura LDAP*

A la cual, se va a añadir los grupos y usuarios correspondientes a nuestra organización, para ello se han creado los siguientes archivos de configuración LDIF y scripts.

El primer LDIF que se ha creado corresponde a los grupos siguientes:

```
#### Administrador del sistema
dn: ou=administradores,dc=iotsec,dc=com
ou: administradores
objectClass: organizationalUnit

#### Trabajadores
dn: ou=trabajadores,dc=iotsec,dc=com
ou: trabajadores
objectClass: organizationalUnit

#### Familiares
dn: ou=familiares,dc=iotsec,dc=com
ou: familiares
objectClass: organizationalUnit

#### Invitados
dn: ou=invitados,dc=iotsec,dc=com
ou: invitados
objectClass: organizationalUnit

#### Usuarios
dn: ou=usuarios,dc=iotsec,dc=com
ou: usuarios
objectClass: organizationalUnit
```

*Tabla 11: Archivo grupos.ldif de los grupos de la unidad organizativa*

Ahora que ya se conoce la estructura de la organización se van a crear los usuarios, para ello se ha creado el siguiente fichero:

```
dn: uid=jsanchez,ou=administradores,dc=iotsec,dc=com
uid: jsanchez
objectClass: top
```

```
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
cn: Juan Sanchez
sn: Sanchez
uidNumber: 1000
gidNumber: 1000
title: Administrador
ou: administradores
mail: jsanchez@iotsec.com
userPassword: {SSHA}cEAS1aJlzUWqnYLwRsNYq61vNU6uDq9b
loginShell: /bin/bash
homeDirectory: /home/jsanchez

dn: uid=agil,ou=trabajadores,dc=iotsec,dc=com
uid: agil
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
cn: Ana Gil
sn: Gil
uidNumber: 1001
gidNumber: 1001
title: Personal Limpieza
ou: trabajadores
mail: agil@iotsec.com
```



```
userPassword: {SSHA}TF1xlfEp8HhB1t+E4yIwei5jqlB9OMyJ
loginShell: /bin/bash
homeDirectory: /home/agil

dn: uid=jlopez,ou=trabajadores,dc=iotsec,dc=com
uid: jlopez
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
cn: Jose Lopez
sn: Lopez
uidNumber: 1002
gidNumber: 1002
title: Personal Mantenimiento
ou: trabajadores
mail: jlopez@iotsec.com
userPassword: {SSHA}m4Y27nX7mGOhXOAhXjHEnH+eZAXVXzCi
loginShell: /bin/bash
homeDirectory: /home/jlopez

dn: uid=csanchez,ou=familiares,dc=iotsec,dc=com
uid: csanchez
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
```

```
cn: Carla Sanchez
sn: Sanchez
uidNumber: 1003
gidNumber: 1003
title: Hija
ou: familiares
mail: csanchez@iotsec.com
userPassword: {SSHA}PWoXHSR4Uab2i7AcKe4N1Qo4fygtbeIK
loginShell: /bin/bash
homeDirectory: /home/csanchez

dn: uid=agarcia,ou=invitados,dc=iotsec,dc=com
uid: agarcia
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
cn: Andres Garcia
sn: Garcia
uidNumber: 1004
gidNumber: 1004
title: Invitado
ou: invitados
mail: agarcia@iotsec.com
userPassword: {SSHA}Xo2OpFAriKAu8X8lpPhzqkL+zKIT+0Tp
loginShell: /bin/bash
homeDirectory: /home/agarcia

dn: uid=jsanchez,ou=usuarios,dc=iotsec,dc=com
```



```
uid: jsanchez
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
cn: Juan Sanchez
sn: Sanchez
uidNumber: 1000
gidNumber: 1000
title: Administrador
ou: usuarios
mail: jsanchez@iotsec.com
userPassword: {SSHA}cEAS1aJlzUWqnYLwRsNYq61vNU6uDq9b
loginShell: /bin/bash
homeDirectory: /home/jsanchez

dn: uid=agil,ou=usuarios,dc=iotsec,dc=com
uid: agil
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
cn: Ana Gil
sn: Gil
uidNumber: 1001
gidNumber: 1001
title: Personal Limpieza
```

```
ou: usuarios
mail: agil@iotsec.com
userPassword: {SSHA}TF1xlfEp8HhB1t+E4yIwei5jqlB9OMyJ
loginShell: /bin/bash
homeDirectory: /home/agil

dn: uid=jlopez,ou=usuarios,dc=iotsec,dc=com
uid: jlopez
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
cn: Jose Lopez
sn: Lopez
uidNumber: 1002
gidNumber: 1002
title: Personal Mantenimiento
ou: usuarios
mail: jlopez@iotsec.com
userPassword: {SSHA}m4Y27nX7mGOhXOAhXjHEnH+eZAXVXzCi
loginShell: /bin/bash
homeDirectory: /home/jlopez

dn: uid=csanchez,ou=usuarios,dc=iotsec,dc=com
uid: csanchez
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
```



```
objectClass: posixAccount
objectClass: shadowAccount
cn: Carla Sanchez
sn: Sanchez
uidNumber: 1003
gidNumber: 1003
title: Hija
ou: usuarios
mail: csanchez@iotsec.com
userPassword: {SSHA}PWoXHSR4Uab2i7AcKe4N1Qo4fygtbeIK
loginShell: /bin/bash
homeDirectory: /home/csanchez

dn: uid=agarcia,ou=usuarios,dc=iotsec,dc=com
uid: agarcia
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
cn: Andres Garcia
sn: Garcia
uidNumber: 1004
gidNumber: 1004
title: Invitado
ou: usuarios
mail: agarcia@iotsec.com
userPassword: {SSHA}Xo2OpFAriKAu8X8lpPhzqkL+zKIT+0Tp
loginShell: /bin/bash
homeDirectory: /home/agarcia
```

*Tabla 12: Archivo usuarios.ldif de los usuarios de la unidad organizativa*

Para no mostrar la contraseña en claro en el archivo de configuración se ha optado por cifrar la contraseña al formato SSHA, para ello se ha utilizado un conversor online.

Para ejecutar los archivos anteriores se han creado los siguientes scripts.

```
#!/bin/bash
/etc/init.d/slapd stop
slapadd -c -v -l grupos.ldif
```

*Tabla 13: Script grupos.sh*

```
#!/bin/bash
/etc/init.d/slapd stop
slapadd -c -v -l usuarios.ldif
```

*Tabla 14: Script usuarios.sh*

Una vez ejecutados los scripts anteriores, se debe reiniciar el servicio ldap con el siguiente comando:

$sudo /etc/init.d/slapd restart

Ahora ya se ha completado la instalación y configuración del phpLDAPadmin, como se observar en la (Figura 8) la estructura definitiva con todos los grupos y usuarios


*Figura 8: Estructura definitiva con todos los grupos y usuarios*

### 4.1.7. Instalación apache tomcat para web y servidor CAS

En este apartado se ha creado la web de la aplicación, que se ha ubicado en el servidor web apache. Se ha utilizado la versión apache-tomcat-8.0.27, la cual se puede descargar desde la web oficial http://tomcat.apache.org/.

Una vez descargado apache, se descomprime el archivo y se extrae en el directorio /opt y se crean dos copias una con el nombre apache-tomcat-8.0.27 para el servidor web y otra tomcat-cas-8.0.27 para el servidor de autenticación.

### 4.1.8. Creación aplicación Web con autenticación vía servidor single sign-on

#### 4.1.8.1. Introducción

El objetivo de este apartado es crear una aplicación Web de forma que la autenticación se realice utilizando un servicio externo de single sign-on. La autorización se realizará de forma local en la misma aplicación Web. La aplicación resultante se ha basado en la supuesta vivienda domotizada con internet de las cosas llamada "IoTsec". La aplicación ha permitido a los usuarios realizar unas acciones u otras en función de su rol. Estas acciones no se han implementado, es decir, no se ha creado un módulo para abrir una puerta o encender el aire acondicionado realmente, ya que lo que se ha creado es un prototipo que en un futuro se le podría dar funcionalidad. Sólo se ha comprobado que las funciones de ese módulo se pueden realizar en función del rol de cada usuario.

Los usuarios de la vivienda y sus datos se han presentado anteriormente en el apartado (4.1.5).

La aplicación Web desarrollada permite que sólo los usuarios registrados puedan acceder a la aplicación con su login y contraseña. A partir de este momento visualizan un menú con la opción para poder entrar a la puerta de casa. En el caso de acceder se visualizará otro menú con (habitación principal, habitación familiares, salón, baño y cocina, y por cada módulo sólo se han mostrado las opciones a las que se tiene acceso según los roles a los que pertenece. En el caso que haya un módulo al que el usuario no tiene acceso a ninguna funcionalidad, se ha mostrado un mensaje indicando que no tiene acceso a dicho módulo.

#### 4.1.8.2. Configuración

En esta parte se ha descargado la última versión de Apache Tomcat 8.0.27 de su página web oficial http://tomcat.apache.org/ a fecha de realización del proyecto y se ha descomprimido en la carpeta /opt dos copias una para el servidor web y otra para el servidor de autenticación CAS.

Para configurar el servidor CAS se ha utilizado el manual de Spring Security y SSO (ehdez73, 2010). Lo que se ha realizado ha sido una configuración de una aplicación web para



añadirle seguridad usando Spring Security y posteriormente se ha configurado la aplicación para la autenticación mediante Single Sign On (usando el servidor CAS de jasig ).

Para la configuración de la página web se ha utilizado un modelo de base para controlar los roles de spring-security del GitHub de Rob Winch (Winch, 2015). Se han anexado los archivos de configuración más importantes al final del documento *(*Tabla 22: ApplicationContext-security*)*, donde se realiza el control de la autorización a los recursos de la aplicación y la configuración necesaria para la comunicación con el servidor CAS y poder llevar a cabo la autenticación de los usuarios de la webapp.

También se han anexado a final del documento uno de los archivos principales la (Tabla 23: Index.jsp), donde se encuentra el menú principal de la Web, y la siguiente (Tabla 24: Puertaentradadecasa.jsp) donde se puede ver una vez se ha conseguido entrar en la vivienda las opciones disponibles y a que roles están asignadas, donde se puede ver que se está controlando la autorización de los recursos de forma local en la aplicación como se observa en el archivo que se ha mencionado anteriormente.

### Autenticación mediante CAS + phpLDAPadmin

En primer lugar, para que la comunicación y autenticación mediante el servidor LDAP funcione correctamente son necesarias tres librerías: cas-server-support-ldap-3.5.2.jar, spring-ldap-1.3.1.RELEASE-all.jar y spring-ldap-core-2.0.2.RELEASE.jar, que han tenido que ser incluidas en el directorio /lib del servidor CAS.

Después se ha tenido que configurar el archivo deployerConfigContext.xml, donde se especifica el authenticationHandler para LDAP (Figura 9) y el contextSource para la autenticación, indicando la url del servidor y los datos para autenticarse en el (Figura 10).

```
<bean class="org.jasig.cas.adaptors.ldap.FastBindLdapAuthenticationHandler" >
      <property name="filter" value="uid=%u,ou=usuarios,dc=iotsec,dc=com" />
     <property name="contextSource" ref="contextSource" />
</bean>
```

*Figura 9: AuthenticationHandler para LDAP*

```
<bean id="contextSource" class="org.springframework.ldap.core.support.LdapContextSource">
    <property name="pooled" value="false"/>
    <property name="urls">
        <list><value>ldap://localhost:389</value></list>
    </property>
    <property name="base" value="dc=iotsec,dc=com"/>
    <property name="userDn"
        value="cn=admin,dc=iotsec,dc=com"/>
    <property name="password" value="123abc"/>
    <property name="baseEnvironmentProperties">
        <map>
            <entry>
                <key>
                    <value>java.naming.security.authentication</value>
                </key>
                <value>simple</value>
            </entry>
            <entry>
                <key>
                    <value>com.sun.jndi.ldap.connect.timeout</value>
                </key>
                <value>2000</value>
            </entry>
            <entry>
                <key>
                    <value>com.sun.jndi.ldap.read.timeout</value>
                </key>
                <value>2000</value>
            </entry>
        </map>
    </property>
</bean>
```

*Figura 10: ContextSource para autenticación con LDAP*

### 4.1.9. Bastionado de Apache

En este apartado se recogen los pasos que se pueden realizar para securizar un servidor Apache (Mir, 2012) y (Mir, 2012)

#### 4.1.9.1. Primer paso: Ocultación de información

Cuando se instala Apache por defecto muestra cierta información del entorno donde se ha desplegado. Un posible atacante puede recopilar datos muy útiles que le puedan aportar vectores de ataques adicionales. Por ejemplo si tenemos una versión de apache con vulnerabilidades reportadas, un posible atacante podría aprovechar dicha vulnerabilidad para atacar nuestro sistema. Para eliminar dicha información deberíamos modificar las siguientes directivas en el fichero **/etc/apache2.conf.d/security**:

```
ServerTokens ProductOnly
```

```
ServerSignature Off
```

*Tabla 15: Directivas a modificar*

Con esto se disminuirá la cantidad de información ofrecida en las cabeceras de respuesta HTTP, se ocultara la versión de apache y el sistema donde se ha instalado.

Otra modo de ocultar información, y confundir a las herramientas automatizadas es personalizando los errores de página. Se cambiara el mensaje por defecto devuelto en la respuesta HTTP por una página personalizada, con el propósito de mostrar la menor información posible o en los casos que el código de respuesta sea un 404 o un 500 mostraremos un 200. El lugar para configurarlo es el fichero de virtualhost, dentro de /etc/apache2/sites-available:

```
<Directory>
...
ErrorDocument 404 /myError404.html
ErrorDocument 500 /myError500.html
...
</Directory>
```

*Tabla 16: Personalizar errores*

Donde myError*.html son las páginas personalizadas que se han creado para que se muestren cuando exista un error, en lugar de la página genérica de Apache.

También hay que denegar el listado de directorio con el fin de evitar que puedan navegar por la estructura del directorio. Para ello se volverá a entrar en el fichero anterior de virtual host y se añadirá;

```
<Directory>
...
Option -Indexes
...
</Directory>
```

*Tabla 17: Denegación del listado de directorio*

#### 4.1.9.2. Segundo paso: Activa lo justo y necesario

Se debe asegurar que cada directorio tenga únicamente activado lo que necesite para ello es necesario desactivar las *"options"* y "*overrides"* para el servidor. Cuando se necesite alguna se ira activando conforme se necesite.

Otra recomendación es la de evitar que apache tenga la posibilidad de seguir enlaces simbólicos. Eso se utiliza para evitar que un posible enlace simbólico malicioso pueda acceder a archivos fuera de la estructura de directorios, con el riesgo de que puedan ser modificados o eliminados.

También es importante desactivar la opción de ejecución la entrada de ejecución común (CGI) en todos los directorios de la web, salvo el que este destinado a alojar todos los ejecutables. De esta manera se minimiza el riesgo en el cual un atacante ejecute código de forma aleatoria.

Tampoco se deben activar módulos innecesarios, y desactivar aquellos que no se vayan a usar.

Por último, se deberían controlar de donde se va a consultar la aplicación web. Este caso solo lo podríamos aplicar si nuestra aplicación se va a consultar desde una red conocida, o desde una única IP, se configuraría de la siguiente forma:

Dentro de una red conocida.

```
<Directory>
...
Order Deny,Allow
Deny from all
Allow from 192.168.1.0/24
...
</Directory>
```

*Tabla 18: Permitir una red conocida*

O para una única IP.

```
<Directory>
...
Order Deny,Allow
Deny from all
Allow from 127.0.0.1
...
</Directory>
```

*Tabla 19: Permitir una IP*

### 4.1.10. Scripts para iniciar tomcat

Para iniciar tomcat hay que arrancar los siguientes servicios, para ello se ejecutará el siguiente Script:

```
sudo chmod a+x /opt/apache-tomcat-8.0.27/bin/*.sh
sudo chmod a+x /opt/tomcat-cas-8.0.27/bin/*.sh
/opt/apache-tomcat-8.0.27/bin/startup.sh
/opt/tomcat-cas-8.0.27/bin/startup.sh
```

*Tabla 20: Script para iniciar tomcat*

## 4.2. Bastionado de plataformas y sistemas

### 4.2.1. Seguridad lógica

La Seguridad Lógica consiste en la aplicación de controles y procedimientos que protejan el acceso a la información restringiendo su acceso tan sólo a las personas autorizadas para hacerlo.

Ámbitos en los que aplicar la seguridad Lógica:

- Política de seguridad de los sistemas.
- Control de acceso.
- Identificación y Autenticación.

- Gestión de privilegios y Registro de usuarios.
- Seguridad en los dispositivos portátiles.

### 4.2.2. Política de seguridad de los sistemas

El primer paso es definir una política de seguridad que deba ser aplicada a los sistemas.

En base a este documento se establecerán el conjunto de medidas preventivas y reactivas de las organizaciones y de los sistemas tecnológicos que permitan resguardar y proteger la información buscando mantener la confidencialidad, la disponibilidad e integridad de la misma.

### 4.2.3. Identificación y Autenticación

Es el primer proceso de control del acceso a cualquier sistema, tradicionalmente se conoce como introducir un ID de usuario y contraseña para iniciar sesión en el sistema, actualmente existen otros métodos más "refinados".

Identificar es "reconocer" un usuario en un sistema y autenticar es comprobar que el realmente el usuario es quien dice ser mediante algún tipo de prueba.

La identificación es un aspecto clave en el control de acceso, todos los usuarios deben tener un identificador único en cada sistema, esto nos permite trazar las actividades individuales de los usuarios y también permite asignar responsabilidades por acciones. Como sistemas de identificación existen:

- Contraseñas.
- Tarjetas inteligentes (RFID).
- Sistemas biométricos.

Se deben considerar varios criterios:

- El proceso de emisión de los identificadores debe ser seguro y encontrarse documentado.

- Seguir el formato estándar establecido.
- No descriptivo de la función de trabajo. (“administrador”).
- No se deben compartir identificadores (“medico”, ”operador”).
- Verificable. El identificador debe ser simple y fácilmente validado.
- Único. Cada usuario debe ser positivamente identificado .

| Tipo de Autenticación | Ejemplo | Ventajas | Inconvenientes |
|---|---|---|---|
| **Algo que sabes** | Contraseña, código o pin (personal identification number) | Sencillo y económico de implementar | Sencillo de averiguar<br>Sujeto a ataques de sniffing de contraseñas, de diccionario, ingeniería social, falta de secreto de los usuarios, etc. |
| **Algo que tienes** | Token, tarjeta de memoria, o tarjeta inteligente | Difícil de atacar | Se puede perder o robar<br>Puede ser caro de implementar |
| **Algo que eres** | Dispositivos biométricos, como reconocimiento de voz, huella digital, escáner ocular, etc. | Portátil<br>Proporciona un mecanismo sencillo de autenticación y muy fiable | Puede ser caro de implementar<br>Puede ser difícil de aceptar por los usuarios<br>Relevantes ratios de falsos positivos y negativos |

*Tabla 21: Criterios de autenticación*

Se deben establecer roles y perfiles para tener un control de acceso a la información en cada sistema y/o aplicación y estos establecerse de acuerdo a algún criterio como:

- Necesidad de saber.
- Perfiles departamentales o funcionales (producción, ingeniería, administración, etc.)
- Niveles jerárquicos (administrador, invitado..).
- Clasificación de la información.

En el cuanto a los usuarios administradores de cada plataforma deben ser identificados y muy restringidos, ya que un posible atacante lo más probable es que busque hacerse con el permiso de administrador para obtener el control total sobre el sistema.

# 5. Caso de estudio – Pruebas realizadas

En este apartado se muestran las pruebas realizadas con dos usuarios uno de ellos es el usuario administrador "Juan Sánchez" con todos los privilegios y un usuario invitado "Andrés García" con privilegios restringidos.

## Juan Sánchez

Iniciamos sesión como Juan Sánchez, haciendo un single sign-on (Figura 11).

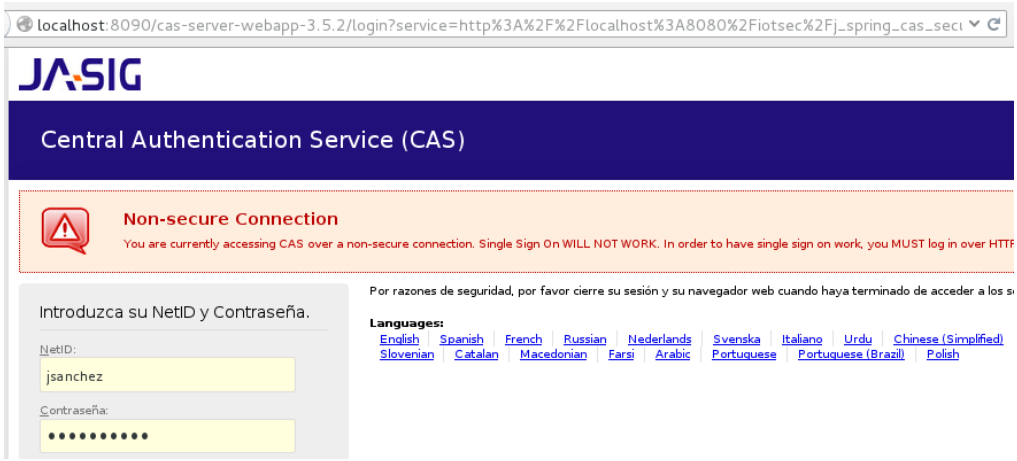



*Figura 11: Login de jsanchez*

Ahora se observa el menú principal de la aplicación web (Figura 12).


*Figura 12: Menú principal de la aplicación web*

Si Jsanchez accede a la puerta entrada de casa, le aparecerá un menú completo con una lista de operaciones, tendrá pleno control y acceso a todas las operaciones, que es el administrador de IoTsec (Figura 13 *y en la* Figura 14)


*Figura 13: Menu entrada de la casa Juan Sánchez*


*Figura 14: Juan Sánchez puede realizar todas las acciones*

Con el usuarios Juan Sánchez se ha mostrado el ejemplo de abrir y cerrar la caja fuerte, este usuario tiene control total sobre toda la aplicación por lo que no se mostraran más ejemplos sobre acciones. Se va a continuar ahora con el usuario invitado.

### Andrés García

Iniciamos sesión como Andrés García, haciendo un single sign-on (Figura 15).

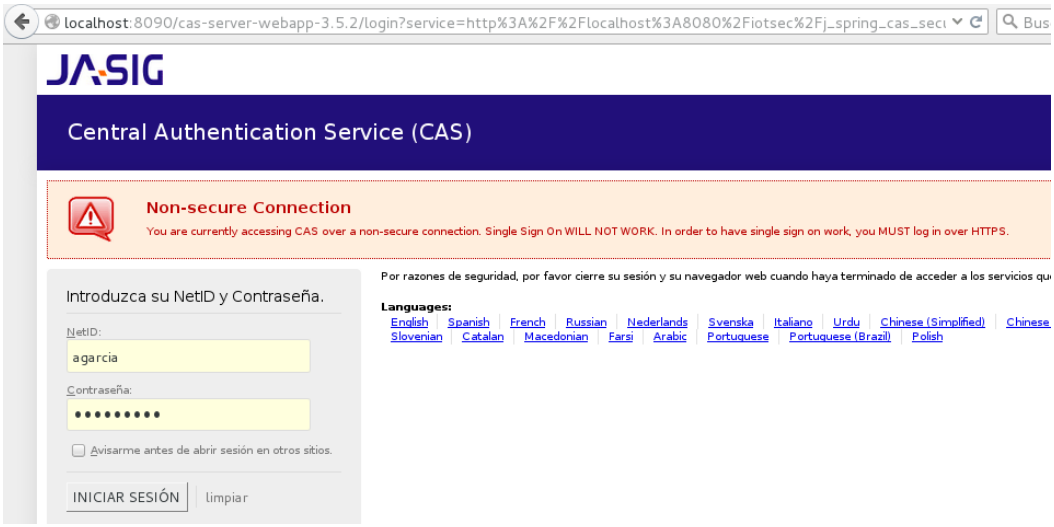



*Figura 15: Andrés García accede a la aplicación web*

Si Andrés García accede a la aplicación se encontrara con el menú principal que se ha podido ver antes en la (Figura 12). Y al entrar a casa le aparecen las siguientes opciones (Figura 16).

Si agarcia intenta acceder a la habitación principal o a la habitación de familiares no se le permitirá acceder ya que no tiene permisos. Por lo que va a acceder a entrar al salón.


*Figura 16: Menú entrada de casa Andrés García*

La (Figura 17) muestra las opciones cuando el usuario invitado, Andrés García consulta las opciones disponibles para realizar en el salón.


*Figura 17: Opciones Salón*

En la (Figura 18) se muestra como Andrés García tiene permisos suficientes para poder consultar el estado del aire acondicionado y en la (Figura 19) se observa como se le deniega el poder modificar el estado del aire acondicionado.


*Figura 18: Andrés García puede consultar el estado del aire acondicionado*

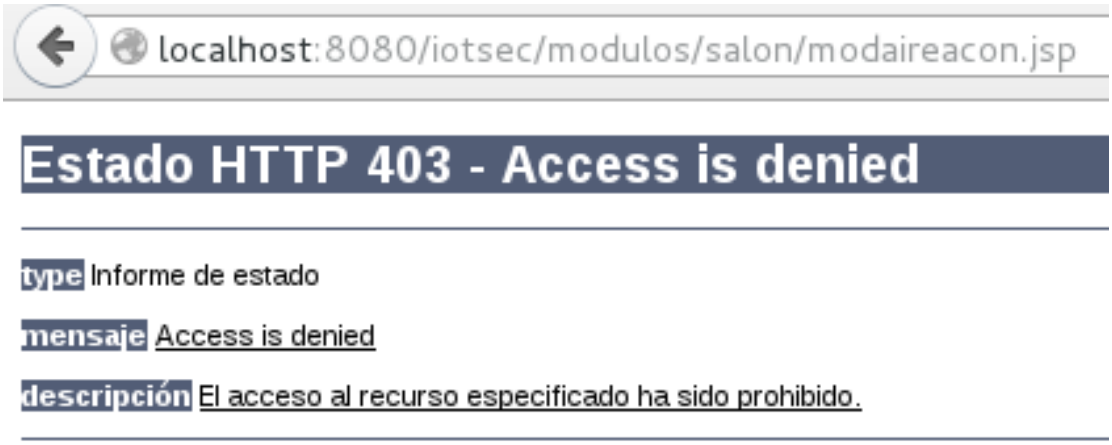



*Figura 19: Andrés García no puede modificar el estado del aire acondicionado*

# 6. Conclusiones y trabajo futuro

## 6.1. Conclusiones

La exposición de las entidades a los riesgos tecnológicos ha aumentado sensiblemente debido a la evolución tecnológica y el uso profuso de las TIC. Como consecuencia, las entidades deben garantizar la confidencialidad, integridad y disponibilidad de sus plataformas y de sus sistemas por lo que es imprescindible ejecutar de manera periódica pruebas de análisis de vulnerabilidades y pentesting para detectar los posibles fallos de seguridad o puntos débiles de los sistemas.

Una adecuada gestión de la seguridad TI debe consistir en:

Determinar la superficie de ataque de su organización, realizar una gestión de parcheo de vulnerabilidades y configurar los sistemas de manera óptima para prevenir la aparición de estas últimas, "bastionar" los sistemas y mantener una monitorización activa de su organización.

## 6.2. Trabajo futuro

Con el propósito de mejorar la seguridad de la infraestructura se colocará un Firewall como el pfsense con las siguientes directrices básicas (Sanz, 2015):

- Utilizar reglas DENY o DROP por defecto.
- Permitir únicamente el tráfico mínimo necesario.
- Fijar cuidadosamente los protocolos.
- Filtrar por interfaces/zonas preferiblemente.
- Filtrar protocolos y tráfico a puertos no conocidos.
- No permitir inicio de conexiones desde la DMZ a la Red Interna.
- No permitir acceso a máquinas o puertos de diagnóstico.
- No permitir protocolos no cifrados si existe una alternativa cifrada.
- Auditorías periódicas.
- Monitorización.

También se podría colocar un Sistema de detección de intrusos en red (NIDS), para analizar el tráfico en tiempo real, contrastar el tráfico analizado con un banco de reglas constantemente actualizado y en el caso de que algún comportamiento en la red coincida con alguna de las reglas, produzca un evento para un análisis posterior por los analistas de seguridad.

Como medidas adicionales se podrían aplicar las siguientes políticas de parcheado:

- Control y documentación de los servicios:
  - Sistema sobre el que corre.
  - Versión.
  - Parches aplicados.
  - Modificaciones en la configuración.
- Notificar bugs encontrados en el Software.
- Listas de correo:
  - Webs especializadas (CERT's, SecurityFocus, etc).
- Aplicación de parches controlada, previamente documentada y autorizada.
- Basada en estándares (ISO 27000).

Con la herramienta Prelude (Alfon, 2011) se podría centralizar logs. monitorizar lo que es normal y cuando detecte algo anormal generar una alerta y finalmente se podría proponer integrar los logs de las aplicaciones en mi monitorización.

# 7. Referencias Bibliográficas

**Agency, National Security. 2011.** Guide to the Secure Configuration of Red Hat Enterprise Linux 5. *NSA.* [En línea] 29 de Febrero de 2011. [Citado el: 14 de Octubre de 2015.] Guide to the Secure Configuration of Red Hat Enterprise Linux 5. https://www.nsa.gov/ia/_files/os/redhat/rhel5-guide-i731.pdf.

**Alfon. 2011.** Instalando y Configurando Centro IDS / IPS con Prelude SIEM, Snort y Prewikka. Parte I. *Seguridad y Redes.* [En línea] 04 de Abril de 2011. [Citado el: 16 de Noviembre de 2015.] https://seguridadyredes.wordpress.com/2011/04/04/instalando-y-configurando-centro-ids-ips-con-prelude-siem-snort-y-prewikka-parte-i/.

**Conner, Margery. 2010.** *Sensors empower the "Internet of Things (Issue 10). pp 32-38.* 2010. 0012-7515.

**CSIRT-CV.** Seguridad en Internet de las Cosas. *Centre de Seguretat TIC de la Comunitat Valenciana.* [En línea] [Citado el: 25 de Octubre de 2015.] http://www.csirtcv.gva.es/sites/all/files/downloads/%5BCSIRT-CV%5D%20Informe-Internet_de_las_Cosas.pdf.

**Edriel. 2015.** Edriel. *El frigorífico inteligente que hace la compra por ti.* [En línea] 18 de Julio de 2015. [Citado el: 2015 de Octubre de 2015.] http://edriel.com/la-nevera-que-hace-la-compra-por-ti/.

**Educación, Ministerio de.** Redes de área local. Aplicaciones y servicios linux. *Institut Puig Castellar.* [En línea] [Citado el: 2 de Octubre de 2015.] http://elpuig.xeill.net/departaments/informatica/fitxers/sistemes-operatius/Redes%20de%20area%20local.Aplicaciones%20y%20servicios%20Linux.pdf/at_download/file.

**ehdez73. 2010.** Cuando trasteo. *Spring Security y SSO.* [En línea] 2010 de Septiembre de 2010. [Citado el: 5 de Octubre de 2015.] http://cuandotrasteo.blogspot.com.es/2010/09/spring-security-y-sso.html.

**Gemalto. 2015.** Gemalto. [En línea] 05 de 11 de 2015. http://www.gemalto.com/.

**Informática, Gits. 2015.** Internet de las cosas y Domótica: Riesgos para la privacidad. *Gits Informática.* [En línea] 2015. [Citado el: 16 de Octubre de 2015.] http://www.gitsinformatica.com/internetdelascosas.html.

**Karen Scarfone, Wayne Jansen y Miles Tracy. 2008.** National Security Agency (NSA). *Guide to General Server Security.* [En línea] Julio de 2008. [Citado el: 15 de Octubre de 2015.] http://csrc.nist.gov/publications/nistpubs/800-123/SP800-123.pdf.

**Kirst, Martin. 2012.** JA-SIG Java Client Simple WebApp Sample. *Jasig.* [En línea] 27 de Abril de 2012. [Citado el: 01 de Noviembre de 2015.] https://wiki.jasig.org/display/CASC/JA-SIG+Java+Client+Simple+WebApp+Sample.

**Mir, Guillermo. 2012.** Bastionando de un servidor web Apache (I). *Security Art Work.* [En línea] 22 de Febrero de 2012. [Citado el: 10 de Noviembre de 2015.] http://www.securityartwork.es/2012/02/22/bastionado-de-un-servidor-web-apache-i/.

**—. 2012.** Bastionando de un servidor Web Apache (II). *Security Art Work.* [En línea] 11 de Abril de 2012. [Citado el: 24 de Noviembre de 2015.] http://www.securityartwork.es/2012/04/11/bastionado-de-un-servidor-web-apache-ii/.

**OWASP. 2015.** Authentication Cheat Sheet. *OWASP.* [En línea] 11 de Agosto de 2015. [Citado el: 22 de Octubre de 2015.] https://www.owasp.org/index.php/Authentication_Cheat_Sheet_Espa%C3%B1ol.

**Rediris. 2002.** Autenticación de usuarios. *Rediris.* [En línea] 15 de Julio de 2002. [Citado el: 3 de Octubre de 2015.] https://www.rediris.es/cert/doc/unixsec/node14.html#SECTION05530000000000000000.

**Report, FTC Staff. 2015.** Internet de las cosas, privacidad y seguridad en el mundo conectado. *Federal Trace Commission Protecting America´s Consumers.* [En línea] Enero de 2015. [Citado el: 29 de Octubre de 2015.] https://www.ftc.gov/system/files/documents/reports/federal-trade-commission-staff-report-november-2013-workshop-entitled-internet-things-privacy/150127iotrpt.pdf.

**Sanz, Juan Manuel. 2015.** Serie de artículos sobre PFsense: firewall perimetral. *Security Art Work.* [En línea] 28 de Mayo de 2015. [Citado el: 15 de Noviembre de 2015.] http://www.securityartwork.es/2014/05/28/pfsense-firewall-perimetral-i/.

**Technologies, Arxan. 2015.** Arxan. [En línea] 04 de 11 de 2015. https://www.arxan.com/.

**Venezuela, Gobierno Bolivariano de.** Curso de LDAP en GNU/Linux. *Institut Puig Castellar.* [En línea] [Citado el: 02 de Octubre de 2015.] http://elpuig.xeill.net/departaments/informatica/fitxers/sistemes-operatius/curso-de-ldap-en-gnu-linux/at_download/file.

**Waugh, Rob. 2014.** Seguridad en Internet de las Cosas: cómo proteger tus dispositivos smart. *welivesecurity.* [En línea] 25 de Noviembre de 2014. [Citado el: 20 de Octubre de 2015.] http://www.welivesecurity.com/la-es/2014/11/25/seguridad-internet-de-las-cosas/.

**Winch, Rob. 2015.** GitHub. *spring-security.* [En línea] Abril de 2015. [Citado el: 10 de Octubre de 2015.] https://github.com/spring-projects/spring-security/tree/master/samples/preauth-xml/src/main/webapp.

# 8. Anexos

*Tabla 22: ApplicationContext-security*

```
<?xml version="1.0" encoding="UTF-8"?>

<!--
  - Sample namespace-based configuration
  -
  -->


<beans xmlns:security="http://www.springframework.org/schema/security"
    xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
                    http://www.springframework.org/schema/security
http://www.springframework.org/schema/security/spring-security-3.1.xsd">


   <!--
                    Enable security, let the casAuthenticationEntryPoint handle all intercepted
urls.
                    The CAS_FILTER needs to be in the right position within the filter chain.
          -->

//---
          <security:http entry-point-ref="casAuthenticationEntryPoint" auto-config="true">

<!-- Entrada de casa -->

                    <security:intercept-url pattern="/index.jsp"
access="ROLE_I"></security:intercept-url>
                    <security:intercept-url pattern="/modulos/Puertaentradacasa.jsp"
access="ROLE_I"></security:intercept-url>

<!-- Pasillo -->

                    <security:intercept-url pattern="/modulos/iluminacion/onoffluces.jsp"
access="ROLE_I"></security:intercept-url>
                    <security:intercept-url pattern="/modulos/habitacionP/opcioneshp.jsp"
access="ROLE_I"></security:intercept-url>
                    <security:intercept-url pattern="/modulos/habitacionF/opcioneshf.jsp"
access="ROLE_I"></security:intercept-url>
                    <security:intercept-url pattern="/modulos/salon/opcionessalon.jsp"
access="ROLE_I"></security:intercept-url>
                    <security:intercept-url pattern="/modulos/banyo/opcionesbanyo.jsp"
access="ROLE_I"></security:intercept-url>
```

```
                <security:intercept-url pattern="/modulos/cocina/opcionescocina.jsp"
access="ROLE_I"></security:intercept-url>

<!-- Habitación principal -->

                <security:intercept-url
pattern="/modulos/habitacionP/iluminacion/onoffluces.jsp"
access="ROLE_I"></security:intercept-url>
                <security:intercept-url pattern="/modulos/habitacionP/cajafuerte.jsp"
access="ROLE_AD"></security:intercept-url>
                <security:intercept-url pattern="/modulos/habitacionP/persianas.jsp"
access="ROLE_I"></security:intercept-url>
                <security:intercept-url pattern="/modulos/habitacionP/estaireacon.jsp"
access="ROLE_I"></security:intercept-url>
                <security:intercept-url pattern="/modulos/habitacionP/modaireacon.jsp"
access="ROLE_F"></security:intercept-url>

<!-- Habitación familiares -->

                <security:intercept-url
pattern="/modulos/habitacionF/iluminacion/onoffluces.jsp"
access="ROLE_I"></security:intercept-url>
                <security:intercept-url pattern="/modulos/habitacionF/persianas.jsp"
access="ROLE_PL"></security:intercept-url>
                <security:intercept-url pattern="/modulos/habitacionF/estaireacon.jsp"
access="ROLE_I"></security:intercept-url>
                <security:intercept-url pattern="/modulos/habitacionF/modaireacon.jsp"
access="ROLE_PL"></security:intercept-url>

<!-- Cocina -->

                <security:intercept-url pattern="/modulos/cocina/iluminacion/onoffluces.jsp"
access="ROLE_I"></security:intercept-url>
                <security:intercept-url pattern="/modulos/cocina/estacafetera.jsp"
access="ROLE_I"></security:intercept-url>
                <security:intercept-url pattern="/modulos/cocina/estahorno.jsp"
access="ROLE_F"></security:intercept-url>
                <security:intercept-url pattern="/modulos/cocina/estaaireacon.jsp"
access="ROLE_I"></security:intercept-url>
                <security:intercept-url pattern="/modulos/cocina/estalavadora.jsp"
access="ROLE_PL"></security:intercept-url>
                <security:intercept-url pattern="/modulos/cocina/estamicro.jsp"
access="ROLE_F"></security:intercept-url>
                <security:intercept-url pattern="/modulos/cocina/estavitro.jsp"
access="ROLE_F"></security:intercept-url>
                <security:intercept-url pattern="/modulos/cocina/modaireacon.jsp"
access="ROLE_PL"></security:intercept-url>
                <security:intercept-url pattern="/modulos/cocina/modcafetera.jsp"
access="ROLE_I"></security:intercept-url>
                <security:intercept-url pattern="/modulos/cocina/modhorno.jsp"
access="ROLE_F"></security:intercept-url>
```

```
                <security:intercept-url pattern="/modulos/cocina/modlavadora.jsp"
access="ROLE_PL"></security:intercept-url>
                <security:intercept-url pattern="/modulos/cocina/modmicro.jsp"
access="ROLE_F"></security:intercept-url>
                <security:intercept-url pattern="/modulos/cocina/modvitro.jsp"
access="ROLE_F"></security:intercept-url>

<!-- Baño -->

                <security:intercept-url pattern="/modulos/banyo/iluminacion/onoffluces.jsp"
access="ROLE_I"></security:intercept-url>
                <security:intercept-url pattern="/modulos/banyo/estaireacon.jsp"
access="ROLE_PL"></security:intercept-url>
                <security:intercept-url pattern="/modulos/banyo/modaireacon.jsp"
access="ROLE_F"></security:intercept-url>

<!-- Salón -->

                <security:intercept-url pattern="/modulos/salon/iluminacion/onoffluces.jsp"
access="ROLE_I"></security:intercept-url>
                <security:intercept-url pattern="/modulos/salon/persianas.jsp"
access="ROLE_PL"></security:intercept-url>
                <security:intercept-url pattern="/modulos/salon/estaireacon.jsp"
access="ROLE_I"></security:intercept-url>
                <security:intercept-url pattern="/modulos/salon/modaireacon.jsp"
access="ROLE_PL"></security:intercept-url>
                <security:intercept-url pattern="/modulos/salon/tv.jsp"
access="ROLE_I"></security:intercept-url>


                <security:custom-filter position="CAS_FILTER"
ref="casAuthenticationFilter"></security:custom-filter>
        </security:http>


        <!--
                Required for the casProcessingFilter, so define it explicitly set and
                specify an Id Even though the authenticationManager is created by
                default when namespace based config is used.
        -->
        <security:authentication-manager alias="authenticationManager">
                <security:authentication-provider
ref="casAuthenticationProvider"></security:authentication-provider>
        </security:authentication-manager>

        <!--
                This section is used to configure CAS. The service is the
                actual redirect that will be triggered after the CAS login sequence.
        -->
        <bean id="serviceProperties"
class="org.springframework.security.cas.ServiceProperties">
```

```
                <property name="service"
value="http://localhost:8080/iotsec/j_spring_cas_security_check"></property>
                <property name="sendRenew" value="false"></property>
        </bean>

    <!--
                The CAS filter handles the redirect from the CAS server and starts the ticket
validation.
        -->
        <bean id="casAuthenticationFilter"
class="org.springframework.security.cas.web.CasAuthenticationFilter">
                <property name="authenticationManager"
ref="authenticationManager"></property>
        </bean>

        <!--
                The entryPoint intercepts all the CAS authentication requests.
                It redirects to the CAS loginUrl for the CAS login page.
        -->
        <bean id="casAuthenticationEntryPoint"
class="org.springframework.security.cas.web.CasAuthenticationEntryPoint">
                <property name="loginUrl" value="http://localhost:8090/cas-server-webapp-
3.5.2/login"></property>
                <property name="serviceProperties" ref="serviceProperties"></property>
        </bean>

        <!--
                Handles the CAS ticket processing.
         -->
        <bean id="casAuthenticationProvider"
class="org.springframework.security.cas.authentication.CasAuthenticationProvider">
                <property name="userDetailsService" ref="userService"></property>
                <property name="serviceProperties" ref="serviceProperties"></property>
                <property name="ticketValidator">
                        <bean
class="org.jasig.cas.client.validation.Cas20ServiceTicketValidator">
                                <constructor-arg index="0" value="http://localhost:8090/cas-
server-webapp-3.5.2"></constructor-arg>
                        </bean>
                </property>
                <property name="key" value="cas"></property>
        </bean>

        <!--
                The users available for this application.
        -->
        <security:user-service id="userService">
                <security:user name="jsanchez"
authorities="ROLE_AD,ROLE_PL,ROLE_PM,ROLE_F,ROLE_I"></security:user>     <!--
Administrador-->
                <security:user name="agil"
authorities="ROLE_PL,ROLE_I"></security:user>       <!-- Pers limpieza-->
```



```
                <security:user name="jlopez"
authorities="ROLE_PM,ROLE_I"></security:user>      <!-- Pers mantenimiento-->
                <security:user name="csanchez"
authorities="ROLE_F,ROLE_I"></security:user>       <!-- Familiar-->
                <security:user name="agarcia" authorities="ROLE_I"></security:user>
                                <!-- Invitado-->
        </security:user-service>

</beans>
```

*Tabla 23: Index.jsp*

```
<%@ page session="false" %>
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="sec" uri="http://www.springframework.org/security/tags" %>
<html>
<head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Menu Principal</title>
</head>
<body>
        <%-- Se muestra la opción de logout --%>
        <div style="border:1px solid #000000;">
                <p><a href="./j_spring_security_logout">Logout</a></p>
        </div><br>
        <div style="border:1px solid #000000;">
                <h1>Autenticacion para lel IoT (IoTsec)</h1>
                <h3>TFG Grado Ingenieria Informatica ETSINF UPV</h3>
                <h6>Alberto Ballestin Perez</h6>
        </div>

        <%-- Modulos disponibles en la aplicacion --%>
        <h4>... Modulos ...</h4>

        <div><a href="modulos/Puertaentradadecasa.jsp">Puerta entrada de
casa</a></div><br>

</body>
</html>
```

*Tabla 24: Puertaentradadecasa.jsp*

```
<%@ taglib prefix="sec" uri="http://www.springframework.org/security/tags" %>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<html>
<head>
	<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
	<title>Entrada de casa</title>
</head>
<body>
	<%-- Se muestra la opción de logout --%>
	<div style="border:1px solid #000000;">
		<p><a href="../j_spring_security_logout">Logout</a></p>
	</div><br>
	<div style="border:1px solid #000000;">
		<h1>Autenticacion para el IoT (IoTsec)</h1>
		<h3>TFG Grado Ingenieria Informatica ETSINF UPV</h3>
		<h6>Alberto Ballestin Perez</h6>
	</div>


	</div>

	<%-- La autorización a las diferentes operaciones se controla de forma local con
Spring Security --%>
	<h3>... Entrada de casa ...</h3>
	<h4>... Operaciones ...</h4>
	<%if (request.isUserInRole("ROLE_I")) { %>
		<a href="iluminacion/onoffluces.jsp">Encender y apagar luces</a><br>
		<a href="salon/opcionessalon.jsp">Abrir puerta salon</a><br>
		<a href="banyo/opcionesbanyo.jsp">Abrir puerta banyo</a><br>
		<a href="cocina/opcionescocina.jsp">Abrir puerta cocina</a><br>
	<% } %>
	<%if (request.isUserInRole("ROLE_PL")||request.isUserInRole("ROLE_PM")) { %>
		<a href="habitacionP/opcioneshp.jsp">Abrir puerta habitacion
principal</a><br>
	<% } %>
	<%if
(request.isUserInRole("ROLE_F")||request.isUserInRole("ROLE_PL")||request.isUserInRole(
"ROLE_PM")) { %>

		<a href="habitacionF/opcioneshf.jsp">Abrir puerta habitacion
familiares</a><br>

	<% } %>
</body>
</html>
```