**Arquitectura de Software**

**Experimento 2**



**Realizado por:**

|  |  |  |
| --- | --- | --- |
| **Persona** | **Rol** | **CódigoUniandes** |
| Carlos Ernesto González Vargas | Ingeniero de Requerimientos | 200819123 |
| Sandra Milena Gómez Ríos | Ingeniero de Producción | 201110951 |
| Andrés Mauricio Erazo Benavides | Ingeniero de Soporte | 201110949 |
| David Pérez Chibuque | Ingeniero de Calidad | 201117818 |
| Willian Alejandro Idrobo Luna | Arquitecto Jefe | 201110544 |
| Erik Fernando Arcos Franco | Ingeniero de Administración | 201110856 |

**Contenido**

**Pag.**

[**1.** **Definición del Sistema** 3](#_Toc297580945)

[**1.1.** **Identificación de patrones** 5](#_Toc297580946)

[**2.** **Experimento A – Seguridad** 7](#_Toc297580947)

[**2.1.** **A1 – Autenticación y Autorización** 7](#_Toc297580948)

[**2.2.** **A2 – Denegación de Servicio / Suplantación** 10](#_Toc297580949)

[**3.** **Experimento B – Disponibilidad** 11](#_Toc297580950)

[**3.1.** **B – Disponibilidad** 11](#_Toc297580951)

**Arquitectura de Software**

**Experimento 2**

1. **Definición del Sistema**

La empresa **Alpes Smart Home (ASH)** desea entrar a participar en el mercado de las casas inteligentes dado el considerable avance de la tecnología en dispositivos móviles, sensores inalámbricos y sistemas embebidos que ha habido durante los últimos años. Por esta razón la empresa **ASH** plantea el proyecto ***Casa Inteligente***, el cual surge para dar respuesta al diseño de arquitectura de solución de un sistema que permita la administración de viviendas inteligentes.

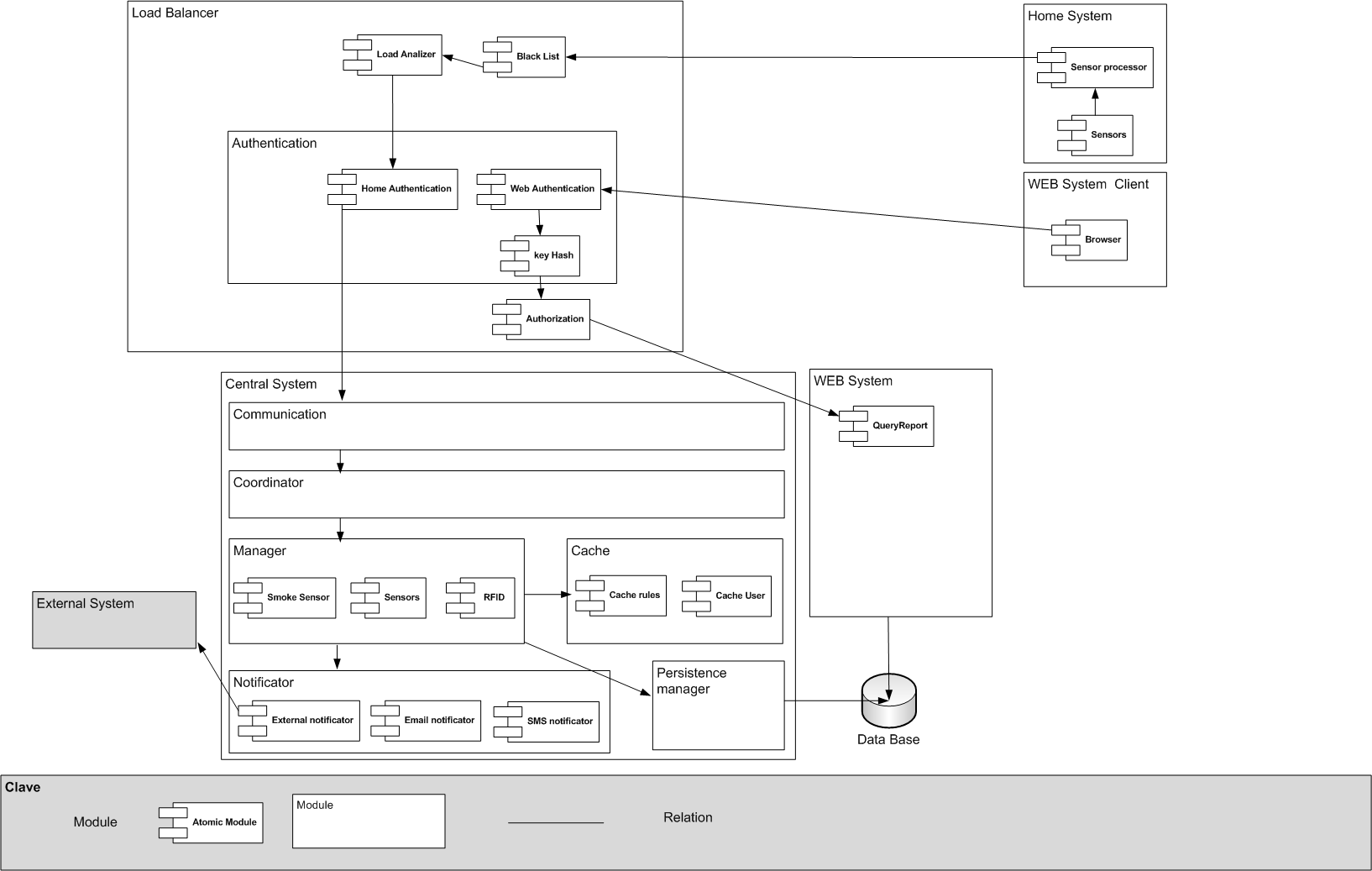
El sistema se debe cumplir con las necesidades iniciales del sistema, descritas a continuación:

* Analizar la información enviada por los sensores de humo, ventanas y puertas de las oficinas y viviendas monitoreadas.
* Una vez se detecta humo en una posición específica de la vivienda u oficina, se procede a notificar a los servicios de control de emergencias. Si se detecta que la vivienda está habitada, se informa a los bomberos y servicios de ambulancias, el número estimado de personas que están en ese momento en el lugar, así como las posibles ubicaciones.
* Si se trata de un sensor de ventana o puerta abierta sin autorización, se debe notificar a la policía sobre el evento, enviando como información adicional, la localización de la ventana dentro del inmueble y el número estimado de personas que están en la vivienda o los sectores de la vivienda en los que se ha detectado movimiento.
* Generar un reporte consolidado mostrando todos los eventos registrados (ordenadas por tipo de evento) de forma cronológica (hora, día, mes, año).
* Evitar consulta y modificación de información no autorizadas. También deberá evitar suplantación de servicio y ante el fallo de un componente de sistema se debe continuar respondiendo solicitudes sin perder el estado del sistema.
* Por cada inmueble u oficina se tienen en promedio 50 tags RFID enviando información cada segundo a cuatro antenas ubicadas en diferentes posiciones de la casa u oficina. Estas antenas a su vez consolidan la información y la envían a la central.
* Cuando se detecta que un tag RFID ha violado una de las reglas definidas para su comportamiento, se debe generar una alarma indicando a todas las porterías y a los responsables de la vivienda u oficina lo ocurrido.

Para el experimento descrito a continuación se tiene el siguiente alcance:

* Diseñar la arquitectura necesaria para recibir la información de los sensores de humo, ventas, puertas y RFIDs con la cual se logrará un segundo en el procesamiento del mensaje. Segundo que será contado a partir de la llegada del mensaje (trama) al sistema central hasta el envío de la notificación a su correspondiente parte. Se usará tácticas de firmas digitales para evitar suplantación y ataques de denegación de servicio.
* Diseñar la arquitectura necesaria para generar un reporte consolidado de los eventos generados por las viviendas u oficinas. Este reporte debe demorarse máximo dos segundos en ser presentado al usuario. Es importante anotar que usuario que generará el reporte será el usuario administrador y podrá ver el consolidado de todos los eventos de todas las viviendas u oficinas. Se garantizarán tácticas de autenticación y autorización.
* Determinar la escalabilidad del sistema a través de unas pruebas de carga, generadas en el sistema central. Para determinar la escalabilidad se tendrá en cuenta que se debe mantener un segundo de respuesta en el procesamiento del evento, como se mencionó anteriormente. Se garantizará alta disponibilidad mediante el uso de balanceo de cargas.

**Vista funcional**



**Vista de despliegue**



* 1. **Identificación de patrones**

Para el desarrollo de este sistema se tuvieron en cuenta los siguientes patrones:

***Acceso a recursos – temprana***

Se aplica en la parte de obtención de la conexión a la base de datos para la generación de reportes.

***Acceso a recursos – caché***

Es aplicada para el manejo de y la información de los usuarios y las reglas que son configuradas para cada casa u oficina.

***Manejo de eventos – respuesta reactiva a eventos***

Al llegar un evento al sistema se delega la responsabilidad de manejar el evento a un hilo que se encargue de procesarlo de acuerdo a las reglas configuradas.

***Concurrencia y sincronización – coordinador***

Existe un componente receptor de todos los eventos que se encarga de crear una instancia de un manejador que procesa el evento de acuerdo a su tipo.

**Autenticación / Autorización**

La generación del reporte requiere que el usuario cuente con el rol de administrador. Es por esto que se usa tácticas de autenticación / autorización.

Se implementó parte de la estructura de PACE para manejar la seguridad del sistema.

**Mantener datos confidenciales**

El password de la cuenta del usuario se reconoce como información sensible y debe ser manejado como tal. Es por esto que se usa tácticas de cifrado de datos (hash) para conservar la confidencialidad de este campo.

**Firmas Digitales**

Para evitar la suplantación de identidad y verificar la legitimidad de fuente de información se implementó la táctica de firmas digitales.

**Redundancia Activa**

Para favorecer alta disponibilidad se hace uso de balanceo de cargas a servidores distintos usando el algoritmo round robin.

1. **Experimento A – Seguridad**

El objetivo de este experimento es validar las decisiones de diseño tomadas para favorecer la seguridad tanto en la generación del reporte como en el envío de tramas desde los dispositivos del hogar hasta el sistema central.

* 1. **A1 – Autenticación y Autorización**

Tabla 1. Descripción del Experimento A1

|  |  |
| --- | --- |
| **Descripción del Experimento** | |
| ***Título*:** Autenticación y autorización | ***ID*:** A1 |
| ***Descripción*:** Este experimento busca aplicar las tácticas de autenticación y autorización para evitar modificación y consulta de información no autorizada. | ***Responsable*:** Ingenium |
| ***Propósito:***  ( ) Reparación, actualizar, clarificar  ( X ) Obtener Información técnica  ( ) Obtener información de negocio  ( ) Otros: | |
| ***Propósito:***  Validar los desarrollos del sistema de autenticación, autorización para la modificación de datos relacionado a los usuarios. | |
| ***Descripción del experimento:***  El sistema debe ser capaz de:   * Registrar nuevos usuarios con su nombre de usuario, password y rol asociado. * Validar el ingreso de un usuario registrado. * Determinar las acciones y permisos para cada usuario. | |
| ***Artefactos Creados:***  ASHWebSystemAuthentication -> Es el proyecto donde se valida la autenticación para el ingreso del usuario y el cifrado Hash.  ASHWebSystemAuthorization -> Autorización para la ejecución de funciones al rol asociado.  ASHWebSystem -> Se encarga de creación, edición de usuarios del sistema. | |
| ***Criterio de terminación:***  Un usuario no registrado en el sistema no debe poder acceder las funciones disponibles (generación de reporte).  El usuario no autorizado no puede acceder a las funciones a las cuales no tenga permiso. | |
| ***Recursos Requeridos:***  Recurso humano -> Se necesita el trabajo de 2 personas con conocimientos en java, JEE, JSF, autorización y autenticación.  Recursos técnicos ->NetBeans IDE 7.0, Microsoft Visio, Selenium IDE 1.0.11 | |
| ***Duración estimada:***  13 horas | |

Tabla 2. Resultados del Experimento A1

|  |
| --- |
| **Resultados del Experimento** |
| ***Resumen de los resultados:***  Se implementaron pruebas en la herramienta Selenium IDE para probar el funcionamiento del sistema de autenticación y autorización.  En el primer caso de prueba se intenta acceder con datos de usuario incorrectos, arrojando un mensaje de error en los datos ingresados.    En el segundo caso de prueba se accede con un usuario que no tiene permiso de visualización de la información de reportes de alertas, mostrando al usuario que no tienen autorización a dicha información.    En la tercera prueba, Se accede con usuario que tiene permiso de visualización del reporte y se puede acceder a dicha información.    En este experimento se obtuvo un tiempo de respuesta inferior a 2 segundos. |
| ***Duración Real:***  14 horas |
| ***Recursos Reales:***  Recurso humano -> Se necesita el trabajo de 2 personas con conocimientos en java, JEE, JSF, autorización y autenticación.  Recursos técnicos ->NetBeans IDE 7.0, Microsoft Visio, Selenium IDE 1.0.11 |
| ***Recomendaciones:***  Se puede hacer uso de las herramientas dispuestas para seguridad y autorización de las tecnologías Java, haciendo un manejo de sesiones más sencillo y dejando el manejo de la autenticación y autorización al contenedor de aplicaciones. |

* 1. **A2 – Denegación de Servicio / Suplantación**

Tabla 3. Descripción del Experimento A2

|  |  |
| --- | --- |
| **Descripción del Experimento** | |
| ***Título*:** Denegación de servicio / suplantación | ***ID*:**A2 |
| ***Descripción*:**  Este experimento busca verificar que el sistema se protege contra un ataque de denegación de servicios (DoS) y suplantación de información. | ***Responsable*:**Ingenium |
| ***Propósito:***  ( ) Reparación, actualizar, clarificar  ( X ) Obtener Información técnica  ( ) Obtener información de negocio  ( ) Otros: | |
| ***Propósito:***  Verificar y validar los artefactos desarrollados y la arquitectura planteada en términos de seguridad y disponibilidad. | |
| ***Descripción del experimento:***  Verificar el desarrollo para el sistema de seguridad es capaz de detectar y soportar un ataque de denegación de servicios. Detectar si la información proveniente de una casa es legítima.  Se simulan dos posibles ataques al sistema, el primero un atacante que envía peticiones al sistema con mayor frecuencia que los sistemas de las casas normales, el sistema lo detecta y evita el procesamiento de sus peticiones. El segundo ataque simula un mensaje al sistema central que no ha sido firmado correctamente, el sistema lo detecta y evita procesar este mensaje. En ambos casos se notifica y se cierra la conexión con la casa para evitar la sobrecarga del sistema. | |
| ***Artefactos Creados:***  ASHLoadBalancer BlackList -> El sistema permite una detección de un ataque de denegación de servicios de acuerdo a la frecuencia de llegada de peticiones provenientes de una vivienda, registrando la fuente del ataque y evitando el procesamiento de las peticiones hasta que manualmente sea removido del BlackList.  ASHHomeSystemAuthentication -> Permite verificar que la información proviene de una fuente legitima. Se encarga de establecer una firma desde el sistema Home System para que sea verificada en el sistema de seguridad desde el módulo Home System Autentication. En caso de no verificarse satisfactoriamente, se notifica y se cierra la conexión con esa casa para evitar ataque DOS. | |
| ***Criterio de terminación:***  El sistema detecta los ataques de denegación de servicio y los controla, evitando procesar más peticiones de estas fuentes y avisando acerca del ataque para que sea atendido. | |
| ***Recursos Requeridos:***  Recurso humano -> Se necesita el trabajo de 2 personas con conocimientos en java, estilos y tácticas de arquitectura, bases de datos, sockets, Threads, etc.  Recursos técnicos -> Base de datos Oracle XE, NetBeans IDE 7.0, Microsoft Visio, JUnit. | |
| ***Duración estimada:***  16 horas | |

Tabla 4. Resultados del Experimento A2

|  |
| --- |
| **Resultados del Experimento** |
| ***Resumen de los resultados:***  En la implementación de tácticas como firmas digitales se obtuvo el siguiente resultado.   * Si la trama enviada desde la casa/oficina no se verifica adecuadamente, dicha casa/oficina será bloqueada y se realizará la notificación respectiva. * Si existe alguna casa/oficina que envíe tramas con una frecuencia superior a la normal, dicha casa/oficina fue bloqueada y notificado dicho evento. |
| ***Duración Real:***  22 horas |
| ***Recursos Reales:***  Recurso humano -> Se necesita el trabajo de 2 personas con conocimientos en java, estilos y tácticas de arquitectura, bases de datos, sockets, Threads, etc.  Recursos técnicos -> Base de datos Oracle XE, NetBeans IDE 7.0, Microsoft Visio, JUnit. |
| ***Recomendaciones:***  Se recomienda ubicar los mecanismos de seguridad ( verificación de firmas) en cada uno de los sistemas centrales existentes, debido a que como se encuentra en el momento ( en el servidor del balanceador de cargas) se convierte en un cuello de botella al momento de verificar las firmas digitales provenientes de cada casa/oficina. |

1. **Experimento B – Disponibilidad**

El objetivo de este experimento es validar la decisión de diseño que involucra la implementación de un balanceador de cargas en el sistema. Además se requiere demostrar que los tiempos aún se conservan después de satisfacer atributos de calidad como seguridad (incluido en este experimento)

* 1. **B – Disponibilidad**

Tabla 6. Descripción del Experimento B

|  |  |
| --- | --- |
| **Descripción del Experimento** | |
| ***Título*:** Disponibilidad | ***ID*:** B |
| ***Descripción*:** Este experimento busca verificar la disponibilidad del sistema | ***Responsable*:**Ingenium |
| ***Propósito:***  ( ) Reparación, actualizar, clarificar  ( X ) Obtener Información técnica  ( ) Obtener información de negocio  ( ) Otros: | |
| ***Propósito:***  Verificar las decisiones de diseño asociadas a la disponibilidad del sistema. | |
| ***Descripción del experimento:***  En este experimento debe simular la recepción de información proveniente de los tags RFID que se encuentran en cada uno de los inmuebles. Suponga que por cada inmueble u oficina se tienen en promedio 50 tags RFID enviando información cada segundo a cuatro antenas ubicadas en diferentes posiciones de la casa u oficina. Estas antenas a su vez consolidan la información y la envían a la central. En la central un servidor cuyo objetivo es balancear cargas y verificar la firma digital asociada a cada trama, reparte la carga en dos servidores principales donde se procesa el mensaje y/o evento. | |
| ***Artefactos Creados:***  ASHCentralSystem ->Es el proyecto principal donde se encuentra toda la lógica de negocio para procesar los eventos recibidos por cada casa u oficina.  ASHHomeModule ->Es el proyecto con la lógica para la simulación de los eventos generados por la casa u oficina.  ASHLoadBalancer -> Es el proyecto que contiene la lógica encargada de re direccionar las peticiones recibidas a cada uno de los servidores centrales disponibles, además de manejar la seguridad del sistema. | |
| ***Criterio de terminación:***  Cuando se detecta que un tag RFID ha violado una de las reglas definidas para su comportamiento, se debe generar una alarma indicando a todas las porterías y a los responsables de la vivienda u oficina lo ocurrido. Esto debe suceder en un tiempo inferior a un segundo.  Si un servidor central, deja de estar disponible, los sistemas centrales restantes deben atender la totalidad de las peticiones recibidas sin perder el mensaje.  En caso de que un sistema central fallé y aún no se haya recibido la respuesta de todos los mensajes que fueron enviados hacia él, dichos mensajes deben reenviarse a los servidores centrales restantes disponibles. | |
| ***Recursos Requeridos:***  Recurso humano -> Se necesita el trabajo de 3 personas con conocimientos en java, sockets, estilos y tácticas de arquitectura.  Recursos técnicos ->NetBeans IDE 7.0, Microsoft Visio, JUnit, JUnitPerf | |
| ***Duración estimada:***  18 horas | |

Tabla 7. Resultados del Experimento B1

|  |
| --- |
| **Resultados del Experimento** |
| ***Resumen de los resultados:***  A continuación se presenta la gráfica de tiempo contra cantidad de usuarios concurrentes. Se puede notar que a medida que se incrementa el número de usuarios concurrentes, el tiempo de respuesta aumenta cada vez más rápido.    En la siguiente gráfica se puede notar como el número de fallas (una falla equivale a que un evento se procesó con un tiempo superior a un segundo) no aumenta hasta llegar a un punto donde se dispara repentinamente (entre 100 y 120 usuarios concurrentes) de manera que en 200 usuarios concurrentes casi el 100% de las peticiones tardan más de un segundo en ser procesadas. Este punto es importante, porque es el que determina la capacidad que tiene el sistema de responder a un número de usuarios concurrentes manteniendo las reglas del negocio (un segundo).    Como puede evidenciarse, con respecto al experimento 1, al incluir nuevos atributos de calidad como seguridad, el tiempo de respuesta aumenta notablemente, pero se está garantizando el procesamiento de todos los mensajes válidos recibidos, a la vez que se ofrece resistencia a un ataque de denegación de servicios y/o spoofing. |
| ***Duración Real:***  14 horas |
| ***Recursos Reales:***  Recurso humano -> Se necesita el trabajo de 2 personas con conocimientos en java, sockets, estilos y tácticas de arquitectura.  Recursos técnicos ->NetBeans IDE 7.0, Microsoft Visio, JUnit, JUnitPerf |
| ***Recomendaciones:***  Para aumentar el desempeño del sistema, se puede mover el módulo de seguridad del servidor de balanceo de cargas a cada uno de los servidores centrales. Para aumentar disponibilidad, se recomienda aumentar un servidor central al balanceador de cargas. |