Propuesta de un Mecanismo de Seguridad para el Intercambio de Datos de Usuario en Redes de Próxima Generación

Universidad del Cauca

Jaime Andrés Oliva Ortega
Fabio Joaquín Fuertes Montenegro

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Telemática
Popayán, Octubre de 2009

Propuesta de un Mecanismo de Seguridad para el Intercambio de Datos de Usuario en Redes de Próxima Generación

Universidad del Cauca

Jaime Andrés Oliva Ortega
Fabio Joaquín Fuertes Montenegro

Trabajo de grado presentado como requisito para optar al título de Ingeniero en Electrónica y Telecomunicaciones

Directora
MARY CRISTINA CARRASCAL REYES
Ingeniera en Electrónica y Telecomunicaciones

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Telemática
Popayán, Octubre de 2009

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

INTRODUCCIÓN

Indudablemente el concepto de convergencia, tecnología de paquetes, movilidad, calidad de servicio garantizada y separación (control, transporte, enrutamiento) en las redes de telecomunicaciones y servicios telemáticos ha permitido el nacimiento y desarrollo de múltiples tecnologías y arquitecturas tendientes a soportar estos conceptos en la forma más amplia posible. Es así como NGN (*New Generation Networks*) nace como un modelo de arquitectura de redes de referencia que debe permitir desarrollar toda la gama de servicios IP multimedia de nueva generación (VoIP (voz sobre IP), Video conferencia, mensajería multimedia, integración con IPTV (Televisión sobre IP), etc.) así como la evolución y migración de los actuales servicios de telecomunicaciones [1] [2].

IMS (*IP Multimedia Subsystem*) para NGN constituye el subsistema de control, acceso y ejecución de servicios común y estándar para todas las aplicaciones. Para los operadores de telecomunicaciones los beneficios en la adopción de IMS pasan por la facilidad y aumento en la velocidad de despliegue de servicios, la reutilización de infraestructura de transporte de red y de servidores de aplicaciones así como también la reducción de costos de la red (personal e infraestructura). De cara al usuario final los servicios basados en IMS permiten la comunicación persona a persona y persona a contenido en gran variedad de modos (incluyendo voz, texto, imágenes y vídeo, o una combinación de todas ellas) de una forma altamente personalizada y mucho más sencilla [3].

Necesariamente la proliferación de usuarios y servicios coexistiendo en una misma arquitectura de referencia basada en tecnología IP (*Internet Protocol*), aumentará dramáticamente los riesgos, en términos de seguridad de la información, en el intercambio de información y los procesos subyacentes. Por lo tanto, será una necesidad de los usuarios y una obligación de los proveedores, ofertar servicios de seguridad (Confidencialidad, Integridad, Disponibilidad, Autenticación, No repudio y Control de acceso) en un entorno IMS [4].

En TS 33.203 [5] el 3GPP (*3rd Generation Partnership Project*) define la arquitectura de seguridad para IMS, en ésta se enumeran 5 asociaciones entre las cuales se encuentra la de proveer seguridad dentro del dominio de la red interna en el enlace entre CSCF (*Call Session Control Function*) y HSS (*Home Subscriber Service*).

En el anteproyecto se definió como objetivo general “definir un mecanismo de seguridad adecuado a los requerimientos de manejo de Perfiles de Usuario en el contexto de Redes de Próxima Generación” por tal motivo y entendiendo que la entidad HSS es la entidad principal de almacenamiento de los datos de los usuarios y de los servicios a los cuales se han suscrito, el trabajo propuesto se centra en la asociación de la arquitectura de seguridad para IMS citada en el párrafo anterior. Con relación a los objetivos específicos y tomando como referencia la tesis “Sincronización de datos de usuario en redes de Próxima Generación” (SincIMS) [6], se plantea el diseño de un mecanismo de seguridad que satisfaga los criterios de seguridad expuestos anteriormente. Las estrategias usadas para cumplir con estos objetivos son las siguientes:

- Construcción de la Base de conocimiento: Apropiación del conocimiento en cuanto a la arquitectura de IMS, las interacciones entres sus componentes y la forma de agregar seguridad al intercambio de información entre las entidades CSCF y HSS.

- Diseño e implementación del mecanismo de seguridad: Construir una solución que asegure la comunicación entre CSCF y HSS creando túneles IPsec (*Internet Protocol SECurity*) para tal fin.

- Pruebas:
    - De seguridad: Haciendo uso de la Metodología Abierta de Comprobación de la Seguridad (OSSTM v3) [7]. Se determinó el grado de seguridad del mecanismo.
    - De rendimiento: Analizar el impacto en la implementación del mecanismo de seguridad en SincIMS [6].

En la construcción de la base de conocimiento detallamos los siguientes hallazgos:

- Sincronización de Datos de Usuario en Redes de Próxima Generación (SincIMS) [6]: El proyecto fue probado con una arquitectura que no hace uso de las interfaces de comunicación nativas de IMS como la interfaz Sh ya que esta se reemplaza por la plataforma middleware la cual es la encargada de la replicación y sincronización. Las pruebas hechas al prototipo implementado mostraron que al usar la plataforma middleware que implementa *group communication*, especializada en la sincronización de datos, se realiza un uso más eficiente de los recursos de red, y al mismo tiempo provee la facilidad de integración con los repositorios de datos de redes heredadas. Pero esta arquitectura no contempla ningún mecanismo de seguridad en la comunicación entre los diferentes dispositivos involucrados en la sincronización, por lo tanto es necesario analizar cuál es el método más apropiado para minimizar las amenazas de seguridad presentes en la comunicación entre las bases de datos de la arquitectura presentada en el trabajo antes mencionado.

- El 3GPP ha generado una serie de recomendaciones de seguridad para IMS entre las cuales se encuentra la especificación 3GPP TS 33.210: *"Network Domain Security; IP network layer security"* [8], en la que se define la arquitectura de seguridad para la señalización de control sobre las interfaces que comunican los elementos de red de las redes NDS/IP (como las Redes UMTS (*Universal Mobile Telecommunications System)* y redes fijas de banda ancha), un NDS corresponde a redes que son administradas por una única autoridad de administración. En esta especificación se definen las interfaces seguras Za y Zb que proveen métodos de autenticación y cifrado para la información que transporta la interfaz Sh en el *core* de IMS la cual hace uso del protocolo *Diameter* [9]. Como se mencionó anteriormente debido a que en el trabajo de "Sincronización de datos de Usuario en Redes de Próxima Generación" no se hace uso de la interfaz Sh no es posible incorporar al modelo propuesto las interfaces Za y Zb, por lo tanto es necesario definir un mecanismo de seguridad similar al que proporcionan estas interfaces que es lo que se pretende realizar con el desarrollo del mecanismo de seguridad para el intercambio de datos de usuario en NGN.

A continuación se detalla cada uno de los capítulos que constituyen este documento, proporcionan la solución al problema de investigación y viabilizan el cumplimiento de los objetivos:

- **Capítulo 1:** Conceptos de Seguridad para el intercambio de información en NGN, este capítulo resume las generalidades en cuanto a los conceptos más relevantes en seguridad informática (principios, tipos de ataque, medidas, contra medidas, etc.) e intercambio de información en redes IP (protocolos, tramas, etc.)

- **Capítulo 2:** Generalidades del Intercambio de Datos de Usuario y Seguridad en IMS, este capítulo presenta los aspectos básicos de la arquitectura de IMS, posteriormente se habla de los procesos de intercambio de información de usuarios entre repositorios de información en este entorno. Luego, se describe el mecanismo de sincronización de datos propio de IMS y un mecanismo de sincronización alternativo (SincIMS) [6].

- **Capítulo 3:** Mecanismo de Seguridad para el Intercambio de Datos de Usuario en NGN, en este capítulo se realiza el planteamiento del problema en el cual se identifican los puntos que muestran la necesidad de llevar a cabo el proyecto, posteriormente se exponen los requerimientos generales que debe cumplir la solución. Con estos elementos se procede a describir el mecanismo y el diseño general propuesto.

- **Capítulo 4:** Implementación del Prototipo de Seguridad Propuesto, este capítulo parte del anterior tomando los elementos que allí se describen y mostrando los pasos más sobresalientes en la configuración y puesta a punto del prototipo planteado. Para llevar a cabo esta implementación en principio se mencionan algunos ajustes a la instalación de la red base IMS sobre la cual se probará el mecanismo, posteriormente se describen los procesos para la configuración de seguridad.

- **Capítulo 5:** Evaluación de Seguridad y Rendimiento del Mecanismo Propuesto, en este capítulo se toma el prototipo implementado para realizar una validación del mismo en cuanto al nivel de seguridad que proporciona y al rendimiento de la red cuando se incorpora el mecanismo propuesto en un modelo de sincronización de datos de usuario en IMS sin elementos de seguridad.

- **Anexo A:** Endurecimiento del Kernel de Linux en Debian
- **Anexo B:** Generación de certificados digitales X.509
- **Anexo C:** Pruebas de seguridad
- **Anexo D:** Descripción de cabeceras en los protocolos: ARP (*Address Resolution Protocol*), IP (*Internet Protocol*), TCP (*Transmission Control Protocol*), UDP (*User Datagram Protocol*) e ICMP (*Internet Control Message Protocol*)

CAPÍTULO 1: CONCEPTOS DE SEGURIDAD PARA EL INTERCAMBIO DE INFORMACIÓN EN NGN

El presente capítulo intenta relacionar los conceptos de redes IP con las amenazas, en términos de seguridad informática, particulares en este tipo de redes. De esta manera se categorizarán dependiendo del tratamiento que se le haga a la información cuando viaja desde el emisor hasta el receptor. Acudiendo de nuevo a los conceptos de redes IP y entendiendo que NGN (*New Generation Networks*) se basa en este tipo de redes, se analizará la pertinencia de esas amenazas en el contexto NGN y su impacto en el intercambio de información.

Inicialmente se hará un resumen sobre los conceptos básicos en el intercambio de información en redes TCP/IP y NGN, sus protocolos y flujos de información que sean pertinentes en el contexto de este trabajo de investigación. Posteriormente un acercamiento a los conceptos sobre (in)seguridad informática (definiciones y categorización) y las soluciones propuestas en redes IP para mitigar el impacto de los distintos tipos de ataques.

1.1. Intercambio de información en redes TCP/IP y de Nueva Generación

1.1.1. Redes IP [10]

Se definirán las redes IP como aquellas que utilizan los protocolos TCP/IP para su funcionamiento. Internet es una red IP.

Las redes IP se caracterizan por haber sido construidas siguiendo un esquema de capas. Cada capa es la responsable de cada una de las diferentes tareas en la comunicación. De esta forma, se puede definir la familia de protocolos TCP/IP como una combinación de cuatro capas (Figura 1). En este esquema, la capa superior accede únicamente a los servicios prestados por la capa situada justo en el nivel inferior a ella. De esta forma, independizamos una capa del resto de capas inferiores, lo que nos permite tener un esquema modular.

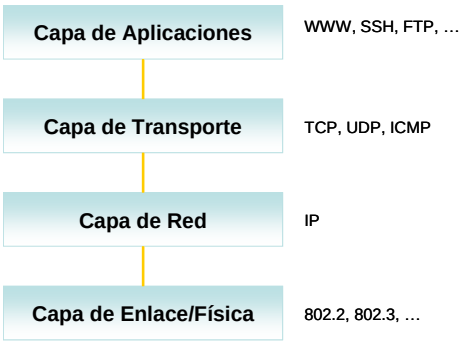

Figura 1. Estructura de cuatro capas

- La capa de enlace o capa física, también denominada la capa de datos o capa de acceso a red, incluye los mecanismos necesarios que permiten enviar y recibir información a través de la red a la que se encuentra físicamente conectado (Ethernet, FDDI (*Fiber Distributed Data Interface*), ISDN (*Integrated Services Digital Network*), etc.). Esta capa corresponde a la capa 1 y 2 del modelo OSI (*Open System Interconnection*).

- La capa de red o capa de interconexión de redes, también denominada capa de Internet, es la encargada de mover los paquetes de información a través de las diferentes redes para llegar a su destino. En esta capa encontramos los protocolos de más bajo nivel, destacando el IP. Esta capa corresponde a la capa 3 del modelo OSI.

- La capa de transporte, es la encargada de proporcionar un flujo de datos entre dos equipos. Este flujo de datos puede ser fiable y orientado a conexión (TCP) o no fiable y no orientado a conexión (UDP). Esta capa corresponde a la capa 4 del modelo OSI.

- La capa de aplicaciones, es la encargada de manejar los detalles particulares relativos a las diferentes aplicaciones que utilizará el usuario (WWW (*World Wide Web*), TELNET (*TELecommunication NETwork*), FTP (*File Transfer Protocol*), etc.). Esta capa corresponde a las capas 5, 6 y 7 del modelo OSI.

Para asegurar el flujo de información adecuado, al momento de iniciar la transferencia (envío) cada capa adiciona información de control a la capa siguiente, el proceso es contrario en recepción, es decir, cada capa desagrega información hasta llegar al receptor de la misma.

Una vez introducidas las bases en las que se fundamenta el diseño de la familia de protocolos TCP/IP, se pasa a la descripción de los paquetes que conforman los protocolos básicos mencionados (principalmente la cabecera de cada uno de ellos), ya que es de vital importancia entender el funcionamiento de estos protocolos para su posterior análisis en términos de la seguridad en la comunicación y en el transporte de información.

- Protocolo ARP [RFC-826]: Permite realizar ciertas tareas cuyo objetivo es asociar un dispositivo IP, que a nivel lógico está identificado por una dirección IP, a un dispositivo de red, que a nivel físico posee una dirección física de red. Este protocolo se utiliza típicamente en entornos de redes de área local, Ethernet. Existe un protocolo, RARP (*Reverse ARP*), cuya función es la inversa. Ver Anexo D para la caracterización de la

trama y cada uno de sus componentes.

- Protocolo IP [RFC-791]: Es el protocolo principal de TCP/IP, encargado de la transmisión y enrutamiento de los paquetes de datos al equipo destino. Es un protocolo no fiable, es decir, que no asegura la recepción final en el equipo destinatario de la información. Para el control de los posibles errores dispone de un protocolo de aviso, ICMP. La fiabilidad de la comunicación debe proporcionarla los protocolos superiores, como TCP. Ver Anexo D para la caracterización de la trama y cada uno de sus componentes.

- Protocolo TCP [RFC-791]: Empleado en la mayoría de los servicios que componen Internet actualmente. Es un protocolo fiable, es decir, se asegura que los paquetes da datos llegan al otro extremo mediante el uso de números de secuencia y de confirmaciones de recepción (ACKs *Acknowledgement*), y es orientado a conexión. Ver Anexo D para la caracterización de la trama y cada uno de sus componentes.

- Protocolo UDP [RFC-768]: Es un protocolo muy sencillo, no orientado a conexión y no fiable, por lo que son los protocolos de nivel superior los que deben asegurarse de la recepción de los datos. Cada operación de envío genera un único datagrama UDP. Es empleado principalmente en aplicaciones multimedia, para el envío de flujos de información sin un coste de conexión asociado. Ver Anexo D para la caracterización de la trama y cada uno de sus componentes.

- Protocolo ICMP [RFC-792]: Es considerado a nivel de IP, ya que es empleado por éste para notificar mensajes de error o situaciones que requieren cierta atención. Debido a que los paquetes ICMP viajan en paquetes IP es a veces considerado un nivel por encima. Ver Anexo D para la caracterización de la trama y cada uno de sus componentes.

1.1.2. Redes de Nueva Generación

Según la ITU (*International Telecommunications Union*), una red de nueva generación NGN, es una red basada en paquetes habilitada para proveer servicios, incluyendo los de Telecomunicaciones, capaz de hacer uso de banda-ancha y tecnologías de transporte habilitadas con QoS (*Quality of Service*), en la cual las opciones de servicio son independientes de la tecnología de transporte implícita. Esto ofrecerá acceso irrestricto de los usuarios hacia los diferentes proveedores de servicios, y soportará la movilidad generalizada facilitando la consistencia y ubicuidad para la provisión de servicios a los usuarios [11].

De la definición anterior y basados en [11] podemos relacionar algunas características de NGN con en el intercambio de información [12]:

- Red multiservicio capaz de manejar voz, datos y video: esta característica implica que la naturaleza de la información es heterogénea, por lo tanto el intercambio que subyace estará ligado al tipo de información transmitida.

- Red con las capas de aplicaciones, control y transporte formalmente separadas:

esta quizá es la característica más importante ya que focaliza el trabajo de investigación a la capa que interactúa con la información de los usuarios.

- Red con Calidad de Servicio garantizada para distintos tipos de tráfico: el intercambio de información debe cumplir, entonces, con políticas de calidad concretas.

- Red que permite el acceso irrestricto por parte de los usuarios a cualquier proveedor de servicios: debido a la convergencia y al ofrecimiento ilimitado de servicios, el intercambio de información de usuario deberá entonces contar con altos estándares de seguridad para asegurar la validez de los mismos en distintos servicios

1.2. Aspectos de seguridad en redes TCP/IP y de nueva generación

NGN está basado en IP, por lo tanto hereda todos los problemas de seguridad informática que tienen las redes IP, ahora, si NGN involucra nuevas entidades y protocolos (y su interacción) en su arquitectura de servicios, necesariamente la problemática de seguridad informática involucra nuevos riesgos y desafíos en cuanto al aseguramiento de la misma [13].

Por lo anterior, los conceptos y análisis que se hayan hecho en cuanto a seguridad informática sobre redes IP, se pueden usar para el desarrollo de este trabajo de investigación con las NGN

Definición del concepto de seguridad de la información

De acuerdo a la Real Academia de la Lengua Española, la definición de seguridad es la siguiente: *"Dicho de un mecanismo: Que asegura algún buen funcionamiento, precaviendo que este falle, se frustre o se violente."*. Para el caso particular de la seguridad de la información, se hace necesario extender esta definición y aplicarla en un contexto válido, toda vez que la información necesita una acepción distinta, esta es la modificación. En este sentido se deben tener en cuenta los siguientes servicios de seguridad [14]:

- **Confidencialidad:** La confidencialidad especifica la necesidad de que la información se revele sólo a las personas, entidades, y procesos autorizados en el momento y en forma autorizada

- **Integridad:** La integridad señala que la información no ha sido alterada, borrada, reordenada, copiada, etc., bien durante el proceso de transmisión o en su propio equipo de origen

- **Disponibilidad:** Requiere que los recursos del sistema informático estén disponibles a las entidades autorizadas cuando los necesiten.

- **Autenticación:** Requiere una identificación correcta del origen del mensaje, asegurando que la entidad no sea falsa.
    - De entidad: Asegura la identidad de las entidades participantes en la comunicación, mediante biometría (huellas dactilares, identificación de iris, etc.), tarjetas de banda magnética, contraseñas, o procedimientos similares.
    - De origen de información: Asegura que una unidad de información proviene de cierta entidad, siendo la firma digital el mecanismo más extendido

- **No repudio:** Ofrece protección a un usuario frente a que otro usuario niegue posteriormente que en realidad se realizó cierta comunicación. Esta protección se efectúa por medio de una colección de evidencias irrefutables que permitirán la resolución de cualquier disputa. El no repudio de origen protege al receptor de que el emisor niegue haber enviado el mensaje, mientras que el no repudio de recepción protege al emisor de que el receptor niegue haber recibido el mensaje. Las firmas digitales constituyen el mecanismo más empleado para este fin.

- **Control de acceso:** Requiere que el acceso a los recursos (información, capacidad de cálculo, nodos de comunicaciones, entidades físicas, etc.) sea controlado y limitado por el sistema destino, mediante el uso de contraseñas o llaves hardware, por ejemplo, protegiéndolos frente a usos no autorizados o manipulación.

Por lo tanto, el objetivo de la seguridad de la información no es otro que preservar la confidencialidad, integridad y disponibilidad de la información; asegurando que el acceso a la misma provenga de entidades (usuarios, equipos, servicios, etc.) válidas en un contexto particular de autorización y autenticación.

1.2.1. Tipos de ataque y ataques más comunes [14].

Existe una clasificación que divide los ataques en 4 grades grupos (ver Figura 2):

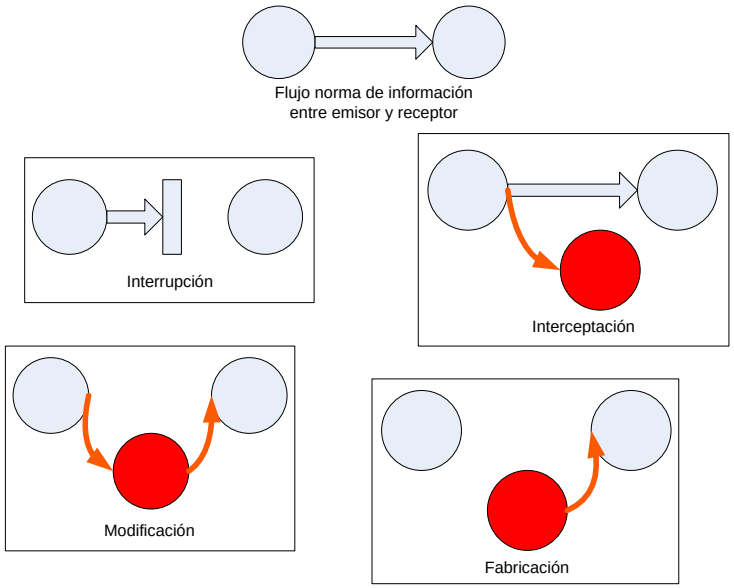

Figura 2. Clasificación de ataques

- **Interrupción:** Un objeto del sistema se pierda, quede inutilizable o no disponible
- **Interceptación:** Un elemento no autorizado consigue un acceso a un determinado objeto del sistema
- **Modificación:** Además de conseguir el acceso consigue modificar el objeto
- **Fabricación:** Modificación destinada a conseguir un objeto similar al atacado de forma que sea difícil distinguir entre el objeto original y el `fabricado'.

Basados en la anterior taxonomía, procedemos ahora a conceptualizar los ataques más comunes en este tipo de redes:

- **Rastreadores o *sniffers* [15]:** Un rastreador o *sniffer* es un programa de captura de las tramas de red. Es algo común que, por topología de red y necesidad material, el medio de transmisión (cable coaxial, UTP, fibra óptica etc.) sea compartido por varios equipos y dispositivos de red, lo que hace posible que un equipo capture las tramas de información no destinadas a él. Para conseguir esto el *sniffer* pone la tarjeta de red en un estado conocido como "modo promiscuo" en el cual en la capa de enlace de datos no son descartadas las tramas no destinadas a la dirección MAC (*Media Access Control*) de la tarjeta; de esta manera se puede obtener todo tipo de información de cualquier aparato conectado a la red. Por lo general este tipo de ataques es pasivo y no inyecta información en el transporte de la misma, por lo tanto se podría calificar como un ataque del tipo interceptación.

- **Suplantaciones de IP o *spoofing* [16]:** Consiste básicamente en sustituir la dirección IP origen de un paquete TCP/IP por otra dirección IP a la cual se desea suplantar. Esto se consigue generalmente gracias a programas destinados a ello y puede ser usado para cualquier protocolo dentro de TCP/IP como ICMP, UDP o TCP. Hay que tener en cuenta que las respuestas del host que reciba los paquetes irán dirigidas a la IP falsificada. Por ejemplo si enviamos un ping (paquete ICMP "echo request") sustituido (*spoofeado*), la respuesta será recibida por el host al que pertenece la IP legalmente. Este tipo de ataques se clasifica como del tipo fabricación.

- **Ataques de contraseñas:** Son las diversas técnicas que incluyen cualquier acción dirigida a romper, descifrar o borrar contraseñas o cualquier mecanismo de seguridad de las mismas.

- **Ataques de hombre en el medio (ó *MITM -man-in-the-middle-*) [17]:** Es un ataque en el que el intruso adquiere la capacidad de leer, insertar y modificar a voluntad, los mensajes entre dos partes sin que ninguna de ellas conozca que el enlace entre ellos ha sido violado. Entre los ataques de este tipo más conocidos se encuentran:

    - ARP Poisoning (o ARP Spoofing): Es un ataque de MITM para redes Ethernet, que permite al atacante capturar el tráfico que pasa por la LAN (*Local Area Network*) y también detenerlo (DoS –*Denial of Service*-). El ataque consiste en enviar algunos mensajes ARP falsos (*spoofed*). Estos *frames* Ethernet contienen las direcciones MAC manipuladas según la conveniencia del atacante. Estos mensajes confunden a los dispositivos de red (principalmente a los switchs). Como resultado los frames de las víctimas son enviados al

atacante o a un destino no válido en el caso de un DoS.

- DNS spoofing: Este ataque utiliza respuestas falsas a las peticiones de resolución DNS enviadas por una "víctima". Hay dos métodos en los que puede basarse el atacante: DNS *"ID Spoofing"* y *"Cache poisoning"* (envenenamiento de la cache). El método *ID Spoofing* se basa en obtener el ID de las peticiones de resolución, el atacante puede lograr esto a través de algún ataque de *sniffing*, como por ejemplo desbordar la tabla ARP *"MAC Flooding"* de los switchs para ponerlos en un modo conocido como *"failopen"* (esto los transforma en un HUB). Siendo capaz de escuchar los ID de las peticiones, el atacante intenta responder a estas antes que el servidor real, logrando de esta forma engañar a la víctima y llevarla así al destino que desee. El método *"Cache poisoning"* es similar al anterior, salvo que se dirige a los servidores de cache de DNS, redirigiendo así a todos sus clientes al host que indique el atacante.

- *Port Stealing* (robo de puerto): En éste ataque el atacante envía muchos frames Ethernet, con la dirección MAC de la víctima como origen, y como destino su propia dirección MAC. Esto hace que el *switch* crea que la víctima está conectada en el puerto del atacante (de ahí el nombre de esta técnica). Cuando el atacante recibe un paquete destinado a la víctima, este genera un *ARP request* preguntando por la MAC asociada a la IP de la víctima. Cuando la víctima responde el *switch* vuelve a conocer en donde está ubicada realmente la víctima, es entonces cuando el atacante reenvía el paquete recibido (intacto o modificado, dependiendo de los intereses del atacante). Luego vuelve a robar el puerto y espera por el próximo paquete con destino a la víctima.

- *DHCP Spoofing:* Las requerimientos de DHCP son hechos con frames de tipo *broadcast*, ya que deben ser escuchados por todos los dispositivos dentro de la red local. Si un atacante responde antes que el verdadero servidor, este puede pasarle información errónea a la víctima, como por ejemplo puede decirle que la puerta de enlace es él.

- **Ataques DoS [18]:** Se genera mediante la saturación de los puertos con flujo de información, haciendo que el servidor se sobrecargue y no pueda seguir prestando servicios, por eso se le dice "denegación", pues hace que el servidor no dé abasto a la cantidad de peticiones realizadas. Entre los ataques de este tipo más conocidos se encuentran:

  - *SYN Floods*: La inundación SYN envía un flujo de paquetes TCP/SYN (varias peticiones con el FLAG SYN en la cabecera), muchas veces con la dirección de origen falsificada. Cada uno de los paquetes recibidos es tratado por el destino como una petición de conexión, causando que el servidor intente establecer una conexión al responder con un paquete TCP/SYN-ACK y esperando el paquete de respuesta TCP/ACK. Sin embargo, debido a que la dirección de origen es falsa o la dirección IP real no ha solicitado la conexión, nunca llega la respuesta.

- Ataque LAND: Un ataque LAND se realiza al enviar un paquete TCP/SYN falsificado con la dirección IP del servidor objetivo como si fuera la dirección origen y la dirección destino a la vez, además de usar un puerto abierto TCP, tanto de destino como de origen. Esto causa que el servidor se responda a sí mismo continuamente hasta colapsar sus recursos.

- *ICMP Floods:* Es una técnica DoS que pretende agotar el ancho de banda de la víctima. Consiste en enviar de forma continuada un número elevado de paquetes ICMP *echo request* (ping) de tamaño considerable a la víctima, de forma que esta ha de responder con paquetes ICMP *echo reply* (ping) lo que supone una sobrecarga tanto en la red como en el sistema de la víctima.

- SMURF: El atacante dirige paquetes ICMP tipo *echo request* (ping) a una dirección IP de broadcast, usando como dirección IP origen, la dirección de la víctima (*Spoofing*). Se espera que los equipos conectados respondan a la petición, usando *echo reply*, a la maquina origen (víctima).

- *UDP Floods:* Básicamente este ataque consiste en generar grandes cantidades de paquetes UDP contra la víctima elegida. Debido a la naturaleza sin conexión del protocolo UDP, este tipo de ataques suele venir acompañado de IP *spoofing*.

- **Ataques a nivel de aplicación para explotar vulnerabilidades conocidas:** Este tipo de ataques difiere en la forma de ejecutarse dada la gran variedad de aplicaciones encontradas en la realidad actual de Internet, pasando por sistemas operativos, servidores de aplicaciones, motores de bases de datos, lenguajes de programación, la combinación entre ellos, etc. Por lo tanto, de forma general se puede concluir que este tipo de ataques se puede llevar a cabo, necesariamente comprometiendo alguna parte (servidor de aplicaciones, motor de bases de datos, el aplicativo, etc.) de la arquitectura de la aplicación en mención. Cabe anotar que el impacto de este tipo de ataques está íntimamente ligado a la severidad en la vulnerabilidad explotada y la relación que exista con las demás piezas en la arquitectura de despliegue de la aplicación.

- **Caballos de Troya, virus y otros códigos maliciosos:** Por lo general este tipo de ataques necesitan la intervención del usuario para la infección y están asociados a la explotación de alguna vulnerabilidad en la máquina comprometida.

1.2.2. Protocolos seguros – Nivel de aplicación [SSH] [19]

SSH (*Secure SHell*) es un protocolo de nivel de aplicación para crear conexiones seguras entre dos sistemas sobre redes no seguras, entre las características principales se encuentran:

- Proporciona terminal de sesión cifrada con autenticación fuerte del servidor y el cliente, usando criptografía de clave pública.
- Variedad de mecanismos de autenticación de usuarios.
- Conexiones TCP arbitrarias de *tunneling* a través de la sesión SSH, protegiendo protocolos inseguros como IMAP (*Internet Message Access Protocol*) y permitiendo el

paso seguro a través de cortafuegos.
- Soporte para métodos de autenticación externa, incluyendo Kerberos
- Transferencias seguras de ficheros.
- SSH está basado en protocolos documentados por el IETF (*Internet Engineering Task Force*).
- Secuencia de eventos de una conexión SSH:

    1. Se crea una capa de transporte segura para que el cliente sepa que está efectivamente comunicando con el servidor correcto. Luego se cifra la comunicación entre el cliente y el servidor por medio de un código simétrico.

    2. Con la conexión segura al servidor en su lugar, el cliente se autentifica ante el servidor sin preocuparse de que la información de autentificación pudiese exponerse a peligro. OpenSSH usa claves DSA (*Digital Signature Algorithm*) o RSA (iniciales de los apellidos de R. Rivest, A. Shamir y L. Adleman, sus creadores) y la versión 2.0 del protocolo SSH para autenticaciones predeterminadas.

    3. Con el cliente autentificado ante el servidor, se pueden usar varios servicios diferentes con seguridad a través de la conexión, como una sesión *shell* interactiva, aplicaciones X11 (*X Windows System v11*) y túneles TCP/IP.

1.2.3. Protocolos seguros – Nivel de transporte [SSL] [20]

SSL (*Secure Socket Layers*) es un proceso que administra la seguridad de las transacciones que se realizan a través de Internet. Se basa en un proceso de cifrado de clave pública que garantiza la seguridad de los datos que se envían a través de Internet. Su principio consiste en el establecimiento de un canal de comunicación seguro (cifrado) entre dos equipos (el cliente y el servidor) después de una fase de autenticación.

El sistema SSL es independiente del protocolo utilizado; esto significa que puede asegurar transacciones realizadas en la Web a través del protocolo HTTP (*HyperText Transfer Protocol*) y también conexiones a través de los protocolos FTP, POP (*Post Office Protocol*) e IMAP. SSL actúa como una capa adicional que permite garantizar la seguridad de los datos y que se ubica entre la capa de la aplicación y la capa de transporte.

De esta forma, SSL es transparente para el usuario (es decir, el usuario puede no conocer que está usando SSL). Por ejemplo, un usuario que utiliza un navegador de Internet para conectarse a una página Web de comercio electrónico protegido por SSL enviará datos cifrados sin tener que realizar ninguna operación especial.

Está compuesto por dos capas (Ver Figura 3):
- La primera capa (*SSL Record Protocol*), encapsula los protocolos de nivel más alto y construye el canal de comunicaciones seguro.

- La segunda capa está formada por tres protocolos:

    - *SSL Handshake protocol*: Se encarga de gestionar la negociación de los

algoritmos de cifrado, y la autenticación entre el cliente y el servidor. Este protocolo define algunas fases para el establecimiento de la comunicación entre el servidor y el cliente:

- Mensajes de Hola
  - Hola del Cliente (inicio de sesión):
    - Propone la versión del protocolo
    - Los cifradores a ser utilizados
    - Es el servidor quien escoge los algoritmos criptográficos a ser usados
  - Hola del Servidor (respuesta al inicio de sesión):
    - versión: versión propuesta por el cliente si la soporta el servidor, si no, la más alta soportada por el servidor.
    - Identificador de sesión: Se acepta la sugerida por el cliente si el servidor la soporta, en caso contrario, el servidor asigna un identificador
    - Lo mismo ocurre con las sugerencias del cliente para los algoritmos de cifrado
- Mensajes de certificados e intercambio de llaves.
  - Intercambio de un secreto pre-maestro.
  - El secreto maestro se deriva a partir de éste.
  - Las llaves necesarias se derivan del secreto maestro
- Cambio de especificación de cifrado y mensajes de finalización

- SSL *Assert Protocol*: Señaliza errores y problemas en la sesión establecida. Cada mensaje tiene 2 bytes:
  - Un byte para el nivel de seguridad (severidad)
    - *warning*: conexión puede reanudarse
    - *fatal*: la conexión se termina inmediatamente
  - Un byte para el código de alerta
    - Mensaje inesperado, falla en el MAC (*Message Authentication Code*) o en el descomprimido.
    - Falla en el intercambio (no pudo establecerse acuerdo), parámetros ilegales (inconsistentes o irreconocibles)
    - Tiempo insuficiente para procesar.
    - Sin certificado, mal certificado, certificado no soportado, certificado revocado, certificado expirado, certificado desconocido.

- *SSL Change Cipher Espec Protocol:* consiste en un solo mensaje de 1 byte que sirve para notificar cambios en la estrategia de cifrado.

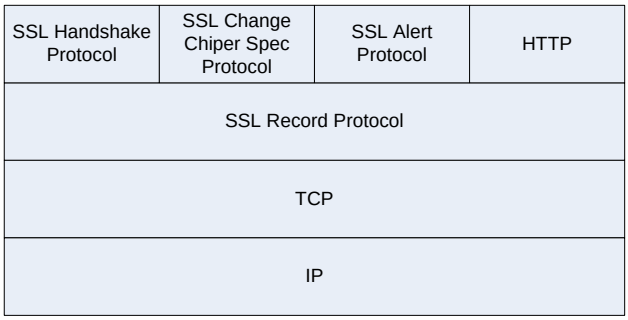

Figura 3. Capas de SSL

Su funcionamiento es el siguiente:

1. El cliente al hacer la conexión informa sobre los sistemas criptográficos que tiene disponibles, y el servidor responde con un identificador de la conexión, su clave certificada e información sobre los sistemas criptográficos que soporta.
2. El cliente deberá elegir un sistema criptográfico, verificará la clave pública del servidor. Entonces se genera una clave cifrada con la clave del servidor. Este es uno de los puntos importantes del protocolo SSL, porque si alguien pudiese descifrar la información, sólo conseguiría romper esa conexión, y una conexión posterior requeriría una clave criptográfica diferente.
3. Una vez finalizado este proceso, los protocolos toman el control de nivel de aplicación, de modo que SSL nos asegura que:
    - Los mensajes que enviamos o recibimos no han sido modificados
    - Ninguna persona sin autorización puede leer la información transmitida
    - Efectivamente recibe la información quien debe recibirla

1.2.4. Protocolos seguros – Nivel de red [IPSec] [21]

IPsec es un conjunto de protocolos cuya función es asegurar las comunicaciones sobre IP, autenticando y/o cifrando cada paquete en un flujo de datos. IPsec también incluye protocolos para el establecimiento de claves de cifrado.

Los protocolos de IPsec actúan en la capa de red, esto hace que sea más flexible, ya que puede ser utilizado para proteger protocolos de la capa de transporte, incluyendo TCP y UDP. IPsec tiene una ventaja sobre SSL y otros métodos que operan en capas superiores. Para que una aplicación pueda usar IPsec no hay que hacer ningún cambio, mientras que para usar SSL y otros protocolos de niveles superiores, las aplicaciones tienen que modificar su código.

IPsec está implementado por un conjunto de protocolos criptográficos para:

- Asegurar el flujo de paquetes
- Garantizar la autenticación mutua
- Establecer parámetros criptográficos.

IPsec fue proyectado para proporcionar seguridad en modo transporte (extremo a extremo) del tráfico de paquetes en el que los equipos de los extremos finales realizan el

procesado de seguridad, o en modo túnel (puerta a puerta) en el que la seguridad del tráfico de paquetes es proporcionada a varios equipo (incluso a toda la red de área local) por un único nodo.

- Modo transporte (IP Seguro)
  - Se protege la carga útil IP (*payload*) (capa de transporte)
  - Comunicación segura extremo a extremo
  - Requiere implementación de IPSec en ambos *hosts*

- Modo túnel (IP seguro dentro de IP estándar)
  - Se protegen paquetes IP (capa de red)
  - Para la comunicación segura entre routers/gateways de seguridad sólo se puede usar este modo
  - Permite incorporar IPSec sin afectar a los *hosts*
  - Se integra cómodamente con VPNs (*Virtual Private Networks*)

La arquitectura de IPSec (Ver Figura 4) utiliza el concepto de asociación de seguridad (SA) como base para construir funciones de seguridad en IP. Una asociación de seguridad es simplemente el paquete de algoritmos y parámetros (tales como las claves) que se está usando para cifrar y autenticar un flujo particular en una dirección. Por lo tanto, en el tráfico normal bidireccional, los flujos son asegurados por un par de asociaciones de seguridad. La decisión final de los algoritmos de cifrado y autenticación (de una lista definida) le corresponde al administrador de IPsec.

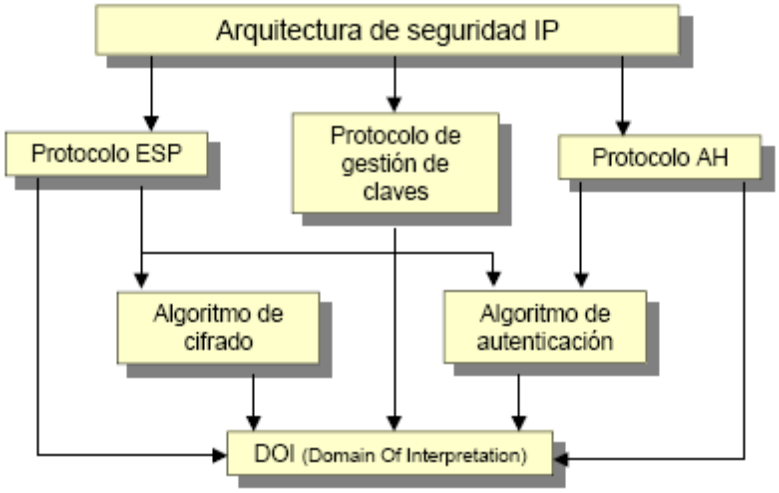

Figura 4. Arquitectura de IPSec

Componentes fundamentales de esta arquitectura:

- Protocolos de seguridad:
  - AH (*Authentication Header*): Encabezado IPSec para proveer servicios de integridad de datos, autenticación del origen de los datos y *antireplay* para IP.
  - ESP (*Encapsulation Security Payload*): Encabezado insertado en el datagrama IP para proveer servicios de confidencialidad, autenticación del origen de los datos, *antireplay* e integridad de datos a IP.

- SA (*Security Association*): Una SA es una clase de conexión que permite establecer los servicios de seguridad del tráfico.
- IKE (*Internet Key Exchange*): Crea SA de forma dinámica, no es parte de IPSec.
- Algoritmos de autenticación y cifrado.

1.2.4.1. AH

AH está dirigido a garantizar integridad sin conexión y autenticación de los datos de origen de los datagramas IP. Para ello, calcula un HMAC (*Hash Message Authentication Code*) a través de algún algoritmo hash operando sobre una clave secreta, el contenido del paquete IP y las partes inmutables del datagrama. Por otro lado, AH puede proteger opcionalmente contra ataques de repetición utilizando la técnica de ventana deslizante y descartando paquetes viejos. AH protege la carga útil IP y todos los campos de la cabecera de un datagrama IP excepto los campos mutantes, es decir, aquellos que pueden ser alterados en el tránsito. En IPv4, los campos de la cabecera IP mutantes (y por lo tanto no autenticados) incluyen TOS (tipo de servicio), *Flags*, Offset de fragmentos, TTL y suma de verificación de la cabecera. A continuación se muestra el diagrama de una cabecera AH (Ver Figura 5):

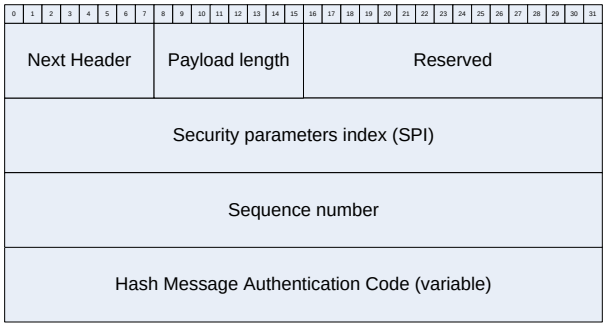

Figura 5. Cabecera AH

- Next header: Identifica el protocolo de los datos transferidos.
- Payload length: Tamaño del paquete AH.
- Reserved: Reservado para uso futuro (hasta entonces todo ceros).
- SPI (*Security Parameters Index*): Indica los parámetros de seguridad que, en combinación con la dirección IP, identifican la asociación de seguridad implementada con este paquete.
- Sequence number: Un número siempre creciente, utilizado para evitar ataques de repetición.
- HMAC: Contiene el valor de verificación de integridad (ICV) necesario para autenticar el paquete; puede contener relleno.

Los servicios de seguridad que ofrece pueden ser entre:

- Dos hosts
- Un host y un *gateway* de seguridad
- Dos gateways de seguridad

1.2.4.2. ESP

El protocolo ESP proporciona autenticidad de origen, integridad y protección de confidencialidad de un paquete. ESP también soporta configuraciones de sólo cifrado y sólo autenticación, pero utilizar cifrado sin autenticación está altamente desaconsejado porque es inseguro. Al contrario que con AH, la cabecera del paquete IP no está protegida por ESP (aunque en ESP en modo túnel, la protección es proporcionada a todo el paquete IP interno, incluyendo la cabecera interna; la cabecera externa permanece sin proteger). ESP Garantiza que el contenido no pueda ser examinado por terceros o, que si lo es, no pueda ser interpretado. Opcionalmente puede incluir la función de AH. A continuación se muestra el diagrama de una cabecera ESP (Ver Figura 6):

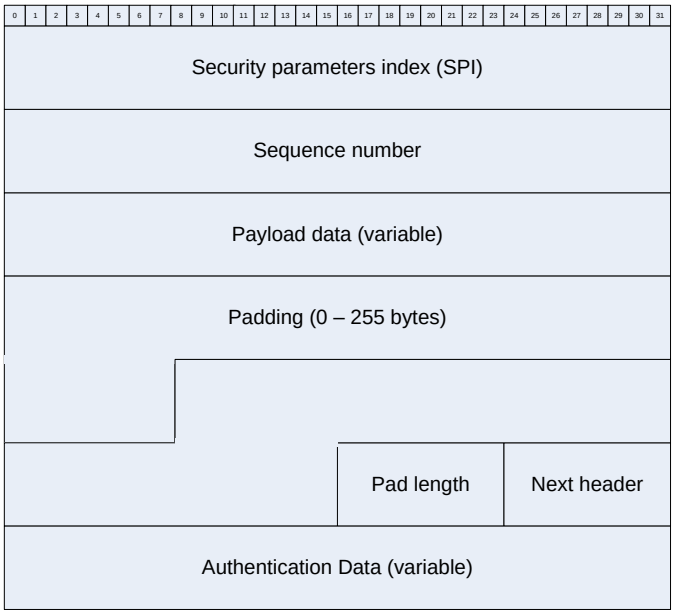

Figura 6. Cabecera ESP

- SPI: Identifica los parámetros de seguridad en combinación con la dirección IP.
- Sequence number: Un número siempre creciente, utilizado para evitar ataques de repetición.
- Payload data: Los datos a transferir.
- Padding: Usado por algunos algoritmos criptográficos para rellenar por completo los bloques.
- Pad length: Tamaño del relleno en bytes.
- Next header: Identifica el protocolo de los datos transferidos.
- Authentication data: Contiene los datos utilizados para autenticar el paquete.

1.2.4.3. SA

Una SA es una clase de conexión que permite establecer los servicios de seguridad del tráfico. En cada SA los servicios de seguridad pueden hacer uso de AH o ESP, pero no de ambos simultáneamente, para utilizar los dos, es necesario establecer dos SA. Una

SA se identifica unívocamente por tres valores:

- SPI
- Dirección IP destino
- Identificador del protocolo de seguridad de IPSec (AH o ESP)

Se pueden definir dos tipos de SA:

- Modo transporte: se trata de una SA entre dos hosts
- Modo túnel: se trata de una SA aplicada a un túnel IP (en este modo existen dos encabezados IP, uno que es el externo que lleva los datos del destino del túnel y otro interno a este que indica el destino final)

1.2.4.4. IKE

IKE es un protocolo de gestión automatizada para los SA, resuelve el problema más importante del establecimiento de comunicaciones seguras: la autenticación de los participantes y el intercambio de claves simétricas. Los dos protocolos de autentificación principales son:

- Clave pre-compartida (*PreShared*): En cada uno de los comunicantes IPSec se configura una misma clave. Los comunicantes IKE se autentifican recíprocamente efectuando un troceo de datos y enviando éstos en clave mediante la clave PreShared configurada. Si el comunicante receptor es capaz de obtener el mismo troceo de manera independiente mediante su propia clave PreShared, tendrá la certeza de que ambos comunicantes comparten el mismo secreto, autentificando de ese modo al otro comunicante.

- Firma RSA: Este sistema utiliza un método en virtud del cual cada dispositivo firma digitalmente un conjunto de datos y los envía a la otra parte. Las firmas RSA utilizan una CA (C*ertificate Authority*) para generar un único certificado digital, que es asignado a cada comunicante a efectos de autentificación. El certificado digital realiza una función similar a la clave PreShared, pero ofrece mucha mayor seguridad.

1.2.4.5. Certificados Digitales [22]

Un certificado es un documento utilizado en PKI (Public Key Infrastructure) emitido y firmado digitalmente por una CA (Certificate Authority), el cual tiene como finalidad comprobar si la clave publica de una entidad o persona es verídica; en otras palabras verificar si en realidad un individuo es quien dice ser.

X.509 es el estándar definido por la UIT-T el cual especifica el formato para los certificados digitales utilizados para transacciones seguras en redes de comunicación abierta como internet. Los certificados X.509 se han definido con la siguiente estructura:

- Un identificador del propietario del certificado.

- Un identificador de la CA.
- Fecha de inicio y final de validez del certificado.
- Número de serie único emitido por la misma CA.
- Firma de la CA.

Algunas de las utilidades de los certificados digitales son:

- Firmar digitalmente documentos garantizando su autenticidad.
- Autenticar pares con el fin de establecer comunicaciones seguras.
- Cifrar digitalmente documentos con la llave púbica del destinatario para que únicamente él pueda tener acceso al documento des-cifrado.
- Intercambiar de manera segura claves compartidas para el establecimiento de una comunicación cifrada.

Es posible aceptar o no la validez de un certificado dependiendo de la confianza previa establecida con la CA que lo emite, esto es, si el certificado autofirmado de la CA se encuentra en la base de datos local como entidad de confianza, todos los certificados firmados por esta serán aceptados.

Así mismo, la integridad del certificado se valida gracias a que el cliente determina el resumen o hash del certificado y compara este dato con la firma de la CA la cual es conocida previamente, si han habido cambios en el documento original los hash no van a coincidir y el certificado se marca como inválido.

CAPÍTULO 2: GENERALIDADES DEL INTERCAMBIO DE DATOS DE USUARIO Y SEGURIDAD EN IMS

2.1. Introducción

En este capítulo, se definen de manera general conceptos referentes a IMS, los cuales son la base del presente trabajo.

En primer lugar se presentan los aspectos básicos de la arquitectura de IMS, posteriormente se habla de los procesos de intercambio de información de usuarios entre repositorios de información en este entorno. Luego, se describe el mecanismo de sincronización de datos propio de IMS y un mecanismo de sincronización alternativo (SincIMS), el cual se propuso y desarrolló como tesis en la FIET (Facultad de Ingeniería Electrónica y Telecomunicaciones – Universidad del Cauca) en 2008. A este trabajo se le da continuidad en el presente proyecto para complementar su funcionalidad al agregarle un mecanismo de seguridad.

Por último, se mencionan los aspectos más sobresalientes de la arquitectura de seguridad planteada por el 3GPP para IMS, de la cual se tomarán algunos elementos para llevar a cabo la propuesta del mecanismo de seguridad en capítulos posteriores.

2.2. Generalidades de IMS

El estándar IMS define una arquitectura genérica para ofrecer servicios de VoIP y multimedia. Fue especificado por el 3GPP y ahora está siendo adoptado por otros grupos de estandarización como ETSI/TISPAN (*European Telecommunications Standards Institute/Telecommunications and Internet converged Services and Protocols for Advanced Networking*); el estándar soporta múltiples tipos de acceso móvil, incluyendo GSM (*Group Special Mobile*), WCDMA (*Wideband Code Division Multiple Access*), CDMA2000 (*Code Division Multiple Access*) y WLAN (*Wireless Local Area Network*) [23].

Los servicios ofrecidos por IMS permiten que los usuarios se comuniquen ya sea con otros usuarios o con el contenido que brinda el servicio en particular. Todo esto en una variedad de modos que incluyen voz, texto, imagen y video de una manera personalizada y controlada [24].

En lo concerniente a los operadores de red, IMS define una arquitectura horizontal (ver sección 2.2.1), donde elementos de servicios y funciones comunes pueden ser reutilizados por múltiples aplicaciones. Del mismo modo, esta arquitectura permite la interoperabilidad entre tecnologías de acceso y proveedores múltiples, además de implementar *roaming*. Igualmente, IMS se integra con redes de voz y datos existentes,

adoptando así muchos de los beneficios más importantes del dominio de las Tecnologías De la Información [24].

2.2.1. Arquitectura de IMS

IMS distribuye sus entidades funcionales dentro de cuatro capas de una arquitectura horizontal, los cuales interactúan entre sí mediante unas interfaces estandarizadas y donde los servicios y las capacidades pueden ser reutilizados por múltiples aplicaciones [3]. La Figura 7 muestra estas capas, denominadas Acceso, Conectividad, Control y Aplicaciones [25] [26].

- Acceso: representa a las tecnologías de acceso (UMTS, Wi-Fi (*Wireless Fidelity*), xDSL (*x Digital Subscriber Line*), CDMA2000, Wimax (*Worldwide Interoperability for Microwave Access*), etc.).

- Conectividad: representa a la capa física del núcleo de red IP, está compuesta de enrutadores interconectados, capaces de ofrecer QoS tanto para las redes de acceso como para las de tránsito.

- Control: es la encargada de la lógica de la señalización, permitiendo establecer, modificar y terminar sesiones por medio de los servidores de control del estado de la llamada (CSCF). En esta capa también se encuentra el servidor de funciones de recursos multimedia (MRF, *Multimedia Resource Function*") que ejecutan servicios de valor agregado para el usuario.

- Aplicación: en ella se encuentran los servidores de aplicaciones (AS, Application Server), en los cuales el operador puede ofrecer servicios propios o de terceros que le permitan diferenciarse de sus competidores.

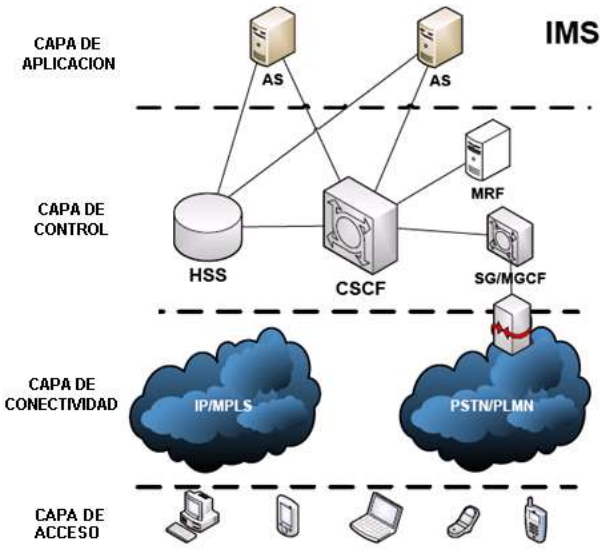

Figura 7. Arquitectura general de IMS

2.2.2. Entidades Funcionales de IMS [26]

CSCF: Grupo de servidores SIP o *proxies*, quienes procesan los mensajes de señalización en diferentes situaciones. Estos son el P-CSCF, el I-CSCF y el S-CSCF.

- P-CSCF (Proxy-CSCF): Es el primer punto de contacto con la red IMS. Su función principal es determinar el Interrogating-CSCF (I-CSCF) de la red local del usuario, además interviene en la aplicación de las políticas de calidad y protección de señalización, su comportamiento es el de un proxy al recibir, procesar o reenviar peticiones de servicios, en roaming es el nodo de la red visitada que dirige la señalización hasta la red local.

- I-CSCF (Interrogating-CSCF): Es el punto de frontera del dominio IMS para comunicarse con otros dominios, con la ayuda del servidor local de usuario (HSS, Home Subscriber Server) determina el Serving-CSCF (S-CSCF) apropiado para un usuario local en el proceso de registro. Además, puede generar los registros de tarificación de los diferentes servicios prestados por la red.

- S-CSCF (Serving-CSCF): Es la entidad que ejecuta los servicios de control de sesión para el terminal de usuario dentro del núcleo IMS y mantiene el estado de dicha sesión para poder prestar un correcto soporte de los servicios. Para realizar la provisión de los servicios el S-CSCF se basa en el perfil de usuario que obtiene del HSS usando el protocolo DIAMETER y decide para cual servidor de aplicaciones deben ir dirigidos los mensajes SIP.

- Además, el S-CSCF realiza ciertas tareas en el proceso de registro y autenticación del usuario IMS y vela por el cumplimiento de las políticas del operador de red para evitar el uso no autorizado de los servicios.

HSS: El HSS es la base de datos principal de IMS que contiene información relacionada con la subscripción para dar soporte a las entidades de red que manejan las sesiones o llamadas.
Por cada usuario ésta base de datos contiene tanto, datos dinámicos (por ejemplo la localización del terminal de usuario), como datos estáticos (por ejemplo números y direcciones asociadas al usuario); así mismo, en ésta se guarda información relacionada con la autenticación, el cifrado y el perfil para cada usuario.

Además, el HSS tiene la función de proveer soporte a las funciones de control del IMS tales como el CSCF, lo que resulta necesario para habilitar al abonado el uso de los servicios IMS.

AS: Proporcionan la plataforma de servicios para IMS, soportan los protocolos SIP y Diameter para la comunicación con el S-CSCF y el HSS respectivamente.

Función de recursos multimedia (MRF): Esta entidad proporciona el procesamiento de los flujos multimedia necesarios para una aplicación y funcionalmente se divide en dos partes: control MRF (MRFC, Controller MRF) y procesamiento MRF (MRFP, Processor MRF). El MRFC muestra una interfaz SIP a los otros componentes de la red, y proporciona las funciones de señalización y control, mientras que el MRFP proporciona

las capacidades de procesamiento multimedia.

2.2.3. Servicios Nativos en IMS

En un entorno IMS los Servidores de Aplicación pueden encontrarse tanto en la red local o *red home,* como en redes de terceros, esto se muestra en la Figura 8. Igualmente en la gráfica se observan las interacciones necesarias entre los AS y otras entidades de la red IMS para la prestación de servicios.

El S-CSCF se encarga de seleccionar el AS adecuado para la prestación de un determinado servicio a un usuario que lo demande. Para cumplir con esta tarea el S-CSCF hace uso de la información contenida en el Perfil de Usuario que se encuentra guardado en el HSS y que es descargado al S-CSCF en el momento del registro de dicho usuario.

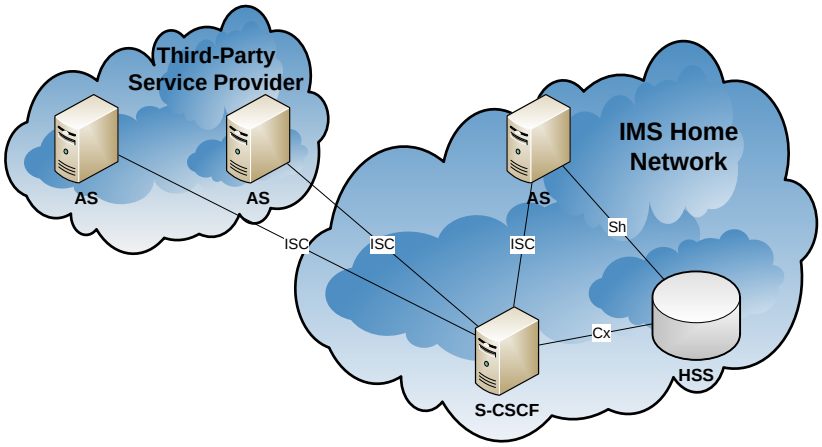

Figura 8. Arquitectura para la prestación de servicios IMS

El camino de señalización con el cliente se completa con la comunicación del S-CSCF con el P-CSCF asignado y de este con el equipo de usuario IMS.

2.2.4. Intercambio de Información relacionada con el Usuario en IMS [25]

Cuando un usuario demanda un servicio determinado en IMS, se produce el intercambio de cierta información de usuario entre las funciones de la arquitectura de servicios mostrados en la Figura 8 (HSS, S-CSCF y SIP AS) a través de interfaces definidas especialmente pera estas interacciones (Sh, Cx, ISC), las cuales utilizan protocolos de comunicación específicos. Este intercambio de información se realiza en el momento del registro del usuario y algunos datos pueden variar una vez el usuario se encuentre registrado en la red.

Los parámetros que se intercambian durante la prestación de servicios IMS, para las interfaces mostradas en la Figura 8 se listan a continuación.

- **Interfaz ISC:** La interfaz ISC (*IMS Service Control*), es usada para la comunicación entre el S-CSCF y los SIP-AS mediante el uso del protocolo SIP. A través de esta se transmite información para notificar al AS desde el S-CSCF sobre un registro implícito de *Public User Identities*, el estado del registro y las capacidades y características del Equipo de Usuario.

- **Interfaz Cx:** Esta interfaz se encuentra especificada entre el S-CSCF y el HSS, aunque también se ha estandarizado como conexión entre el HSS y el I-CSCF, utiliza el protocolo DIAMETER para la comunicación. DIAMETER es un protocolo usado para el manejo de procesos de autorización, autenticación y contabilidad (AAA, por sus siglas en inglés) en IMS. Entre los datos que se transfieren entre el HSS y el S-CSCF se encuentran:

    - Identidades públicas y privadas de usuario y de servicio
    - Lista de redes visitadas autorizadas
    - Información relacionada con el estado de registro
    - Información de autenticación y cifrado
    - Nombre en *Display*
    - Nombre y dirección de AS
    - Nombre y dirección de S-CSCF
    - Información de tarificación
    - Perfil de Usuario

- **Interfaz Sh:** La interfaz Sh se usa para transferir información entre un AS y el HSS mediante el protocolo DIAMETER. Se caracteriza porque es una interfaz intra-operador; es decir, que los dos puntos que conecta se encuentran en la misma red. Entre los datos transportados sobre esta interfaz se encuentran:

    - Datos transparentes a la interfaz, la cual es información específica sobre el servicio que el AS se encuentra prestando y que se guarda en *Repository Data* del HSS.
    - Identidades públicas
    - Estado del usuario
    - Información de localización
    - Información de tarificación
    - Perfil de usuario

Parte de esta información puede ser modificada, como los datos transparentes, el estado de usuario, el nombre del S-CSCF y la información contenida en el *Repository Data*. Estos datos deben ser actualizados automáticamente en el HSS o en el AS, dependiendo de donde se haga la modificación.

2.2.5. Sincronización de Datos de usuario a través de la Interfaz Sh en IMS

Como se mencionó en 2.2.4, la interfaz Sh comunica un AS con el HSS con el fin de:

- Descargar y actualizar datos de usuario

- Pedir y enviar notificaciones en cambios en datos de usuario

De este modo, la sincronización de información de usuario entre el HSS y los AS se resume en los anteriores puntos. Para cumplir con este propósito se usa el protocolo DIAMETER en su forma base con una extensión que maneja los mensajes definidos para esta interfaz [27].

El 3GPP ha especificado una serie de *Command Codes* que extiende el protocolo base DIAMETER, los cuales se utilizan en el mecanismo de sincronización de información usando la interfaz Sh mediante los siguientes procedimientos [27]:

- Lectura de Datos (Sh-Pull)

  Este procedimiento es invocado por el AS para leer información de un usuario específico desde el HSS.

- Actualización de Datos (Sh -Update)

  Procedimiento utilizado para permitir al AS actualizar información transparente guardada en el *Repository Data* del HSS.

- Subscripción a Notificaciones (Sh-Subs-Notif)

  Este procedimiento se invoca por el AS y se usa para subscribirse a notificaciones cuando la información de un usuario en particular haya sido actualizada desde el HSS.

- Notificaciones (Sh-Notif)

  Este procedimiento es utilizado para permitir al HSS informar al AS sobre cambios en la información a la cual éste se ha suscrito previamente para recibir notificaciones.

2.2.6. Sincronización de datos de Usuario a través de Middleware en IMS

Este mecanismo fue propuesto en la tesis “Sincronización de datos de usuario en redes de Próxima Generación” [6] (SincIMS) desarrollada en la FIET por Eivar Armero y Diego Rodríguez, dirigida por la ingeniera Mary Cristina Carrascal, en la cual se hizo un análisis sobre sincronización (Replicación) de datos de usuario entre repositorios de información en el entorno IMS y se propuso como medida para mejorar el desempeño de la red, la incorporación de un *Middleware Open Source* que implementa directivas de *group communication* y está especializado en la sincronización de datos. En la Figura 9 se puede observar la arquitectura de referencia propuesta en este trabajo.

Como se aprecia en la gráfica, una plataforma especializada se encarga del proceso de replicación interconectando las bases de datos que sean necesarias para compartir en tiempo real la información que se necesite, usando las técnicas de los DDBS (*Distributed DataBase Systems*) ya que los repositorios de información de IMS se asemejan a este tipo de sistemas y se usa IP como protocolo base para la conexión entre ellas.

Por otra parte, la utilización de una arquitectura *Middleware* ofrece independencia de los fabricantes de BD, lo cual permite la interoperabilidad en entornos de bases de datos heterogéneas y heredadas, facilita la portabilidad de aplicaciones y hace que la solución sea fácil de implementar y con alto grado de transparencia.

Adicionalmente, una arquitectura *Middleware* utilizando *group communication* garantiza la consistencia de la información, ayuda a lograr un transporte oportuno de la misma y permite un uso eficiente de los recursos de red ya que reduce considerablemente el *overhead* en los procesos de replicación.

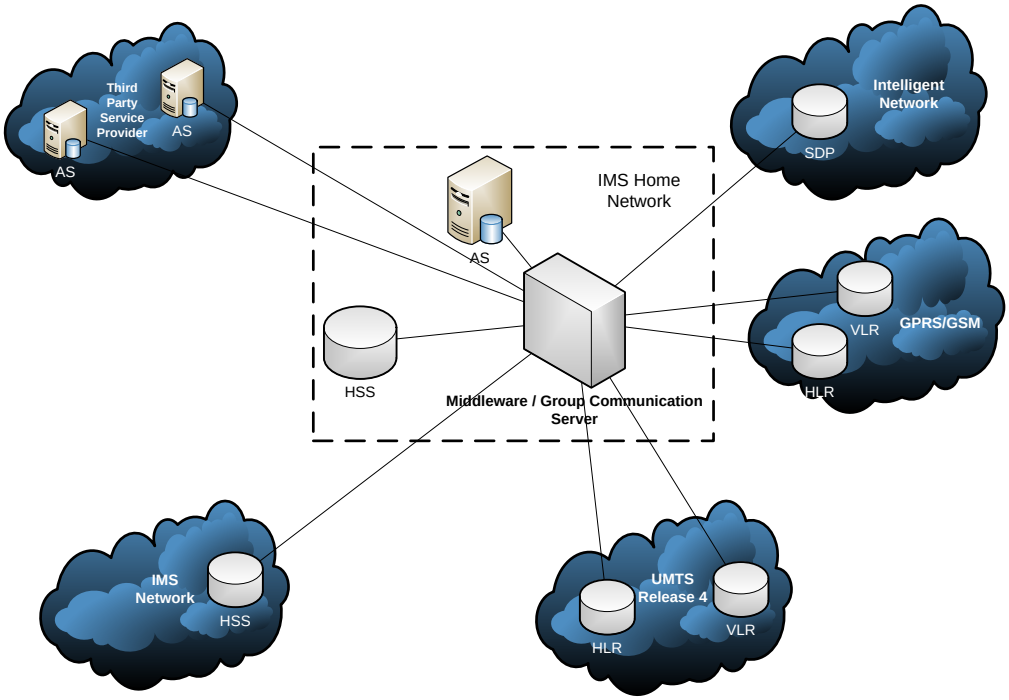

Figura 9. Arquitectura de referencia SincIMS

En el trabajo SincIMS se realizó la implementación de un prototipo para validar este mecanismo, en el cual se sincronizaron los nodos HSS y AS que comparten la información del *Repository Data* para un servicio dado dentro de IMS. Este repositorio es de gran importancia para la prestación de servicios personalizados, pues dentro de sus campos es posible ingresar cualquier tipo de información relacionada con los servicios que un usuario en particular consume [28].

Para esta implementación se utilizaron Open IMS Core como cama de pruebas y la herramienta *open source* Sequoia como plataforma *Middleware*, la cual implementa los protocolos y procedimientos para la replicación de las bases de datos AS y HSS de forma transparente gracias a su capacidad de manejar *clusters*, ofrecer mecanismos de balance de carga y de recuperación de fallos y a su implementación hecha en lenguaje Java lo cual garantiza su portabilidad.

Así mismo, se realizaron pruebas de desempeño comparando el mecanismo tradicional de sincronización de IMS mediante la interfaz Sh y el prototipo de SincIMS, en las cuales

se confirmó un mejor comportamiento del segundo, pues se mostró que entre mayor es la carga para los componentes, mayor es la diferencia de desempeño a favor de SincIMS, mejorando su respuesta en tiempo hasta 16 veces para los casos de mayor exigencia de tráfico en relación con el mecanismo propio de IMS.

2.2.7. Arquitectura de seguridad en IMS

Las especificaciones de seguridad definidas por el 3GPP para IMS [5][8][29], recomiendan un conjunto de mecanismos para ofrecer los servicios de seguridad (autenticación, confidencialidad, integridad) en todos los puntos de la red IMS. La arquitectura de seguridad de IMS se muestra en la Figura 10, la cual consta de cinco asociaciones de seguridad diferentes basadas en las necesidades para la protección de la seguridad según sea el caso.

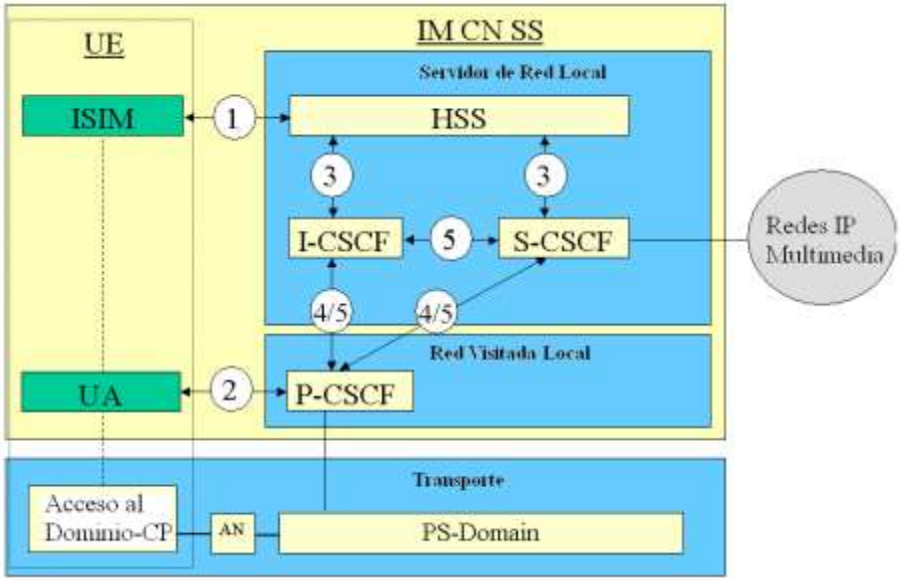

Figura 10. Arquitectura de seguridad IMS

Las diferentes relaciones de seguridad proporcionan esquemas para la gestión de los servicios de seguridad. A continuación se describen estas asociaciones de seguridad según la numeración de la Figura 10 [30]:

1. Provee autenticación mutua entre el UE y el S-CSCF, el HSS delega la función de autenticación del subscriptor al S-CSCF, no obstante el HSS es responsable de generar las llaves y desafíos. En el UE se encuentra la colección de datos de seguridad y funciones asociadas con la identidad privada de usuario (IMPI, IP Multimedia Private Identity) y al menos una identidad pública de usuario externa (IMPU, IP Multimedia Public Identity).

2. Provee un enlace seguro y una asociación de seguridad para proteger el punto de referencia Gm, el cual comunica el UE con el núcleo de red del subsistema IP Multimedia (IM CN, IP multimedia Core Network).

3. Provee seguridad dentro del dominio de la red interna el enlace entre el CSCF y el HSS.

4. Provee seguridad entre nodos SIP de diferentes redes. Esta asociación de seguridad es únicamente necesaria cuando el P-CSCF reside en la red visitante (VN, Visited Network).

5. Provee seguridad entre los nodos SIP de la red interna. Esta asociación de seguridad también se necesita cuando el P-CSCF reside en la VN.

En el diseño del mecanismo de seguridad, presentado en el capítulo 3, se toma como referencia las funcionalidades especificadas en el punto 3, para proporcionar autenticación mutua, confidencialidad e integridad de la información entre el repositorios de información como el HSS y los demás elementos de la red IMS.

En la arquitectura de seguridad IMS, la protección de integridad y confidencialidad de la información se realiza salto a salto en lo que se denomina *cadena de túneles*, como se puede ver en la Figura 11 [29], donde se muestra un dominio de red seguro (NDS, Network Domain Security). Esta arquitectura trata de garantizar integridad y confidencialidad en el intercambio de información salto a salto y usuario a red, pero no usuario a usuario.

Para asegurar las comunicaciones bajo el mismo dominio administrativo se utiliza la interfaz Zb entre las diferentes entidades de la red interna y entre dominios distintos se utiliza la interfaz Za, a través de pasarelas de seguridad (SEG, Security Gateway) [28]. Para estas interfaces se recomienda el uso de IPsec e IKE con diferentes parámetros de establecimiento de asociaciones de seguridad, cifrado de información, funciones de hash, etc.

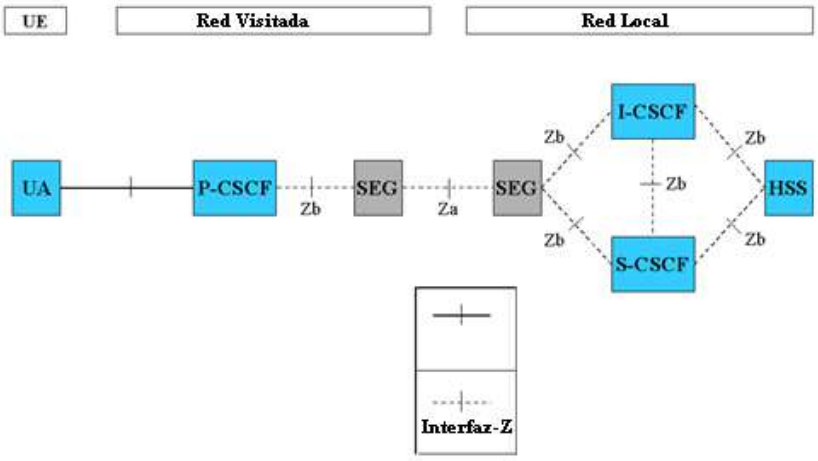

Figura 11. Vista general de la arquitectura IMS y la relación con un NDS

CAPÍTULO 3: MECANISMO DE SEGURIDAD PARA EL INTERCAMBIO DE DATOS DE USUARIO EN NGN

3.1. Introducción

En este capítulo se hace un análisis de algunos conceptos expuestos en los capítulos previos con el fin de llegar al diseño detallado del mecanismo de seguridad propuesto (IMSeg). En primer lugar se realiza el planteamiento del problema en el cual se identifican los puntos que muestran la necesidad de llevar a cabo el proyecto, posteriormente se definen los requerimientos generales que debe cumplir la solución, con esto se tienen los elementos suficientes para caracterizar el mecanismo y plantear un diseño general.

Con lo anterior se llega al diseño del prototipo que validará la solución propuesta, en este se describe detalladamente cada uno de sus componentes y sus funcionalidades. Finalmente se realiza una exploración de las herramientas existentes que permitirán llevar a cabo la implementación del prototipo según el diseño planteado.

3.2. Planteamiento del problema

Como se mencionó en los anteriores capítulos la seguridad es un elemento de importancia crítica en Redes de Nueva Generación y en general en cualquier entorno IP donde los datos se transportan de un nodo a otro de la red a través de caminos inseguros. Dado que uno de los objetivos de las NGN es proveer a los usuarios servicios personalizados y en tiempo real, los riesgos de seguridad deben ser minimizados puesto que para la prestación de este tipo de servicios es indispensable conocer y gestionar adecuadamente el Perfil de Usuario, que es una colección de información relacionada con el usuario la cual se genera a partir de una serie de parámetros activados por diferentes condiciones como su ubicación, las capacidades del dispositivo, las preferencias de *QoS*, la presencia, la adición o la sustracción de ciertos servicios en tiempo real, entre otros. Como es de suponerse esta información debe ser manipulada bajo estrictas medidas de seguridad que permitan garantizar integridad y confidencialidad de la misma y autenticación de las entidades que de alguna manera tienen acceso a ella [31][32].

Este trabajo toma como referencia el proyecto SincIMS [6] descrito en 2.2.6, en el cual fundamentalmente se propone la adición de un *Middleware* especializado en sincronización de datos para realizar el proceso de replicación de información entre repositorios de una red IMS, con esto es posible mejorar el desempeño de la red pues este mecanismo es considerablemente más rápido y genera menos carga que el uso del mecanismo propio de IMS a través de la interfaz Sh.

Sin embargo, el Middleware es quien maneja TODO el acceso a las bases de datos de este dominio IMS, por tanto si este nodo de la red no incorpora conexiones seguras, ninguna de las comunicaciones o accesos a bases de datos de otras redes o del mismo dominio podrán garantizar los servicios de seguridad requeridos en este tipo de comunicaciones, lo que conlleva a poner en riesgo la integridad y confidencialidad de los datos de usuario tanto en la red local como en las visitantes.

Los resultados del trabajo SincIMS demostraron que la respuesta de los recursos de red utilizando el prototipo propuesto fue mejor que la obtenida implementando el mecanismo propio de IMS (interfaz Sh), pero la arquitectura de prueba no contempla ningún elemento de seguridad que garantice integridad y confidencialidad de la información ni autenticación de las partes involucradas en estos procesos ya que el enfoque del trabajo estaba en mejorar la respuesta del mecanismo de sincronización estandarizado en IMS.

Teniendo en cuenta lo anterior y dado que los proveedores de las NGN tienen responsabilidades específicas dentro de su dominio en lo que respecta a seguridad. Por ejemplo, han de aplicar los servicios y prácticas de seguridad adaptados para:

a) Protegerse a sí mismos

b) Garantizar que la seguridad de extremo a extremo no está comprometida dentro de su red [33].

Es evidente que una implementación de SincIMS tal como está planteado no es viable en un entorno de producción, ya que un posible atacante podría valerse de métodos relativamente sencillos como el uso de un *sniffer* de red para capturar paquetes que le brinden algún tipo de información útil a sus propósitos. Si esto ocurre el atacante estaría en capacidad de obtener datos como nombres de usuario y contraseñas de bases de datos, nombres de tablas, etc., además de datos privados de usuario que se actualicen o consulten mientras el intruso realiza el espionaje, todo esto gracias a que la mayor parte de la información se transmite en texto plano de un nodo a otro de la red.

El hecho de que SincIMS no implemente un proceso de autenticación entre las partes que consultan o intercambian datos de usuario abre otro agujero de seguridad puesto que expone la red a ataques de interceptación tales como el conocido *hombre en el medio,* donde el intruso se hace pasar por una de las partes sin autenticar, adquiriendo la capacidad de leer, insertar y alterar la información, que en este caso puede tratarse de datos privados de usuario del entorno IMS [17].

Como se observa, existe la necesidad de profundizar mediante una investigación que posibilite incorporar al intercambio de datos usuario en NGN, un mecanismo para garantizar en cierta medida integridad, confidencialidad y autenticación de la información al interior de una red IMS.

3.3. Definición de los Requerimientos Para un Mecanismo de Seguridad en NGN

Teniendo en cuenta el problema planteado y después de hacer un análisis a los primeros capítulos, se encontraron los requerimientos más sobresalientes para la solución:

- El mecanismo debe hacer uso de protocolos internacionalmente aceptados basados en IP ya que este constituye la piedra angular para la convergencia de redes, dichos protocolos se deben ajustar a la arquitectura de seguridad estandarizada por el 3GPP en las especificaciones técnicas TS 33.203 y TS 33.210 [5][8].

- Como se mencionó en los capítulos previos los servicios de seguridad que se deben proveer en un Dominio de Red Seguro (NDS) son: Integridad de la información, Autenticación del origen de los datos y Confidencialidad de la información. El mecanismo propuesto debe proporcionar todos estos servicios al interior del NDS.

- El mecanismo de seguridad debe proporcionar seguridad salto a salto al interior del NDS, es decir se debe proveer un canal de seguridad independiente para cada segmento entre los nodos del dominio.

- Si bien incorporar a una red un mecanismo de seguridad que involucre intercambio de llaves de seguridad, establecimiento de túneles seguros, cifrado, descifrado, etc., introducirá tráfico adicional y procesamiento extra a la red, se deben tener en cuenta las condiciones de ancho de banda y uso de CPU de los dispositivos involucrados, con el fin de no sobrepasar los límites requeridos para el correcto funcionamiento de las aplicaciones y procesos que se llevan a cabo normalmente al interior de una red IMS.

- La configuración inicial y puesta en marcha del mecanismo de seguridad deben ser tareas que se realizan previamente al arranque de la red IMS, por lo tanto una vez realizadas las pruebas de seguridad y rendimiento, el mecanismo debe estar incorporado a los nodos de la red como un servicio que pueda iniciarse automáticamente con el sistema operativo de los dispositivos.

- El mecanismo debe ser transparente tanto para las aplicaciones como para los usuarios de la red, esto quiere decir que para integrar el mecanismo al sistema no es necesario realizar modificaciones en las aplicaciones de usuario ni en el software que constituye el *Core* de IMS.

3.4. Caracterización de la solución

Con lo planteado en el apartado anterior y en los capítulos previos, se tienen los elementos suficientes para definir el ambiente en el que se va a plantear el Mecanismo de Seguridad en el intercambio de datos de usuario en NGN.

En primer lugar se tiene por sentado que el estándar IMS se perfila como una solución efectiva y aceptada mundialmente para la consecución de la convergencia de redes y por ende de las NGN. Por este motivo, se enfatizó el análisis en la proposición de un mecanismo que se integre en un entorno IMS.

Con el problema puntualizado a IMS, se sabe que los datos de usuario en éste tipo de redes se almacenan en bases de datos centralizadas (HSS), o dependiendo de la aplicación, algunos de ellos pueden alojarse también en bases de datos asociadas a un

Servidor de Aplicaciones dentro de un dominio IMS, por tanto la atención de este trabajo reside en el intercambio de información que se lleva a cabo dentro de un dominio IMS o NDS y no se tendrán en cuenta las interfaces entre el *core* IMS y el UE del usuario ya que gracias a la naturaleza heterogénea y convergente de las Redes de Próxima Generación, el acceso a una red IMS se puede realizar desde dispositivos y redes con tecnologías diversas, por consiguiente se debe analizar la seguridad en el acceso a una red IMS de manera independiente para cada tecnología.

En el capítulo 2 se mostró que IMS ya cuenta con una estandarización de seguridad, que si bien no es muy detallada, describe los protocolos y métodos que deben utilizarse para brindar los servicios de seguridad requeridos en una red IMS nativa. Dado lo anterior y teniendo en cuenta que los resultados de la investigación del trabajo de referencia (SincIMS) mostraron que es posible mejorar la respuesta de una red IMS típica con la introducción de un middleware encargado de realizar la replicación de datos de usuario entre repositorios de información reemplazando el método nativo de IMS (interfaz Sh), se decidió trabajar con base a la arquitectura de red presentada en este trabajo, puesto que, como ya se expuso, no cuenta con ningún elemento de seguridad y es necesario incorporar este mecanismo para realizar las pruebas respectivas y verificar si los resultados obtenidos siguen siendo mejores que los de una red IMS estándar y así validar la investigación previa.

El hecho de que SincIMS utiliza un middleware que controla el acceso a la información de todo el sistema dentro de un dominio IMS y por tanto el intercambio de información en esta red está centralizado en dicho dispositivo, se ajusta a lo planteado anteriormente en cuanto a que IMSeg está enfocado para funcionar en el *Core* IMS, en consecuencia el mecanismo propuesto debe estar involucrado con todas las interfaces del middleware, lo cual puede facilitar el diseño de la solución, ya que si se realizan conexiones seguras con la plataforma Middleware, todas las comunicaciones entre o hacia bases de datos del dominio estarían aseguradas.

3.5. Mecanismo de Seguridad en el intercambio de información de usuario en NGN

Con los estudios descritos en los capítulos 1 y 2, con los requerimientos claramente establecidos y una vez definida la arquitectura de referencia base (SincIMS) con la cual se va a trabajar, fue posible delimitar el rango de posibilidades existentes para optar por una solución al problema formulado. Así se hicieron los siguientes planteamientos y deducciones:

- Siendo consistentes con el requerimiento de seguridad salto a salto, se deduce que un modelo de operación de seguridad de túneles encadenados (*chained-tunnel/hub-and-spoke*) es una alternativa que está acorde con las especificaciones de seguridad en IMS y es una manera sencilla y eficiente de enfocar la implementación del mecanismo propuesto.

- Dado sus características, IPsec es la mejor opción para implementar la solución dado que este garantiza la autenticación de las partes mediante el establecimiento de

Asociaciones de seguridad. Las funciones de hash aseguran la integridad de la información y su implementación encaja perfectamente con el modelo de operación de túneles encadenados.

- Una implementación de IPsec entre dos hosts no implica realizar cambios a nivel de aplicación puesto que es un conjunto de protocolos que trabajan sobre el Protocolo de Internet o capa de red. Lo cual facilita la integración a la red IMS y está acorde con el requerimiento de la transparencia del mecanismo para las aplicaciones.

- El protocolo de seguridad recomendado por las especificaciones del 3GPP para uso mediante IPsec es ESP, ya que implementa cifrado de la información lo cual incrementa en gran medida la confidencialidad de los datos, por lo tanto IMSeg se debe implementar usando este protocolo.

- En muchas implementaciones de IPsec, incluyendo la arquitectura de seguridad propuesta por el 3GPP para IMS, se recomienda usar el modo túnel debido principalmente a que en este se protegen paquetes IP completos, no solamente la carga útil como lo hace el modo transporte. Además en una comunicación segura entre routers/gateways de seguridad sólo se puede usar este modo, por tanto se decidió implementarlo en la solución propuesta.

- En cualquier entorno de servidores es necesario llevar a cabo procedimientos para proteger al sistema contra ataques y mantener un nivel elevado de seguridad, por lo tanto la primera tarea para implementar el mecanismo de seguridad será “Endurecer” el sistema operativo de cada servidor.

- La implementación de un firewall local en un servidor, que permita únicamente las conexiones desde y hacia puertos TCP/UDP conocidos, es una práctica recomendada en seguridad de centros de datos, debido a que el firewall puede impedir el acceso a posibles aplicaciones inseguras o puertas traseras instaladas en la maquina. Además dado que el túnel IPsec protege todo el tráfico entre dos hosts y no discrimina aplicaciones ni puertos, IMSeg debe incorporar elementos de firewall para disminuir aun más los riesgos de seguridad que comprometan tanto los datos como el sistema en general.

En la Figura 12 se puede apreciar el diseño inicial para la incorporación del mecanismo de seguridad al dominio local IMS (*IMS Home Network*), el cual utiliza una plataforma Middleware para el acceso a las bases de datos AS y HSS y la replicación entre las mismas. Dependiendo de las exigencias, puede replicarse información entre bases de datos de la misma red, con otras redes IMS, con redes de proveedores de servicios o con redes heredadas.

Los nodos de la Red Local cuentan con el elemento IMSeg el cual constituye el mecanismo que en primer lugar filtra el tráfico saliente y entrante para que únicamente exista comunicación entre aplicaciones permitidas y segundo lleva a cabo los procedimientos para establecer los túneles IPsec que garantizan un alto nivel de autenticación integridad y confidencialidad.

Las redes externas se comunican con las bases de datos locales a través del SEG

(Security Gateway). Este se ubica en el borde del dominio con el fin de asegurar las comunicaciones entrantes o salientes del mismo, establece conexiones con SEGs de otros dominios IMS o redes heredadas y en el interior del dominio IMS crea un túnel IPsec hacia la plataforma middleware al igual que los demás nodos.

Pueden existir varios SEGs en un NDS; su número depende de la necesidad de diferenciar los destinos externos alcanzables, la necesidad de balancear la carga de tráfico y evitar puntos de fallo [8].

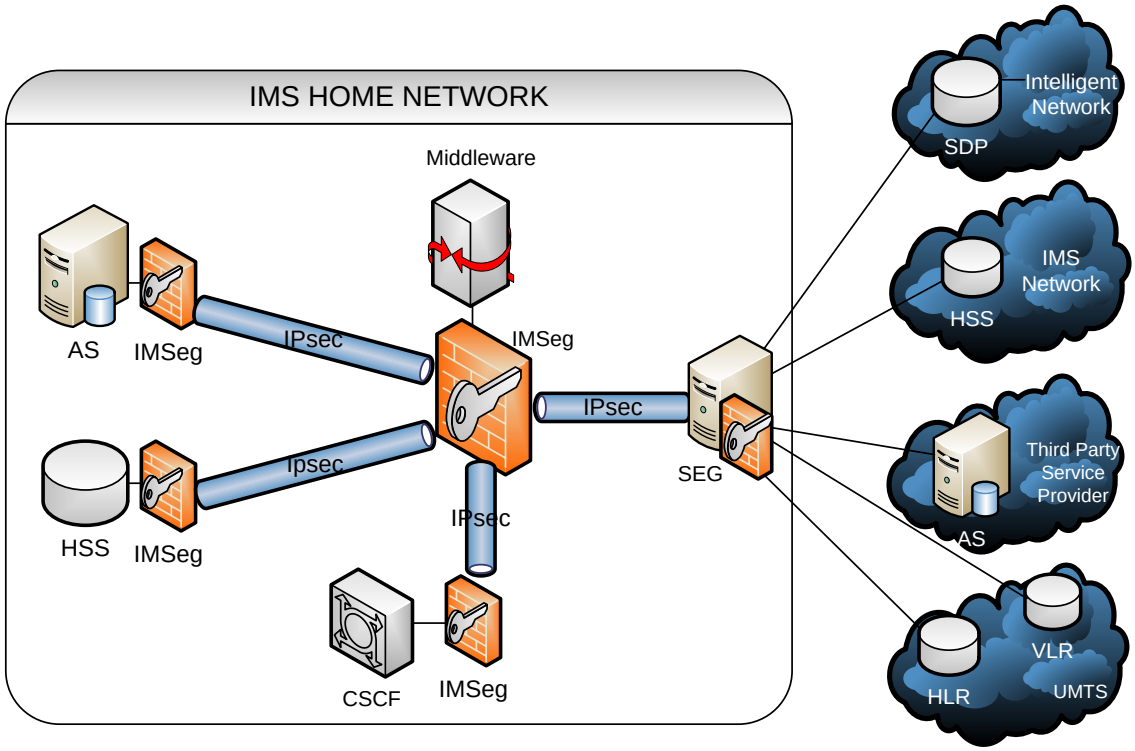

Figura 12. Incorporación del Mecanismo de Seguridad en una red IMS con replicación mediante Middleware

3.6. Prototipo para la validación del mecanismo propuesto

Después de llegar a una idea clara de la solución, se procedió a realizar el diseño de un prototipo para la validación del mecanismo de seguridad propuesto. Además se hizo una exploración, análisis y selección de posibles herramientas a ser usadas en dicho prototipo.

3.6.1. Diseño

De acuerdo a lo visto anteriormente, entre los módulos de la Arquitectura de Servicios Nativos de IMS se comparte cierta información de usuario que puede ser estática o puede variar durante una sesión establecida. Por lo tanto, para realizar la implementación del

prototipo dentro de esta arquitectura se debe seleccionar un tipo de información dinámica y que pueda ser modificada fácilmente con el fin de verificar que el intercambio de esta se realice de manera segura.

Para efectos prácticos la sincronización de información entre bases de datos representa una excelente manera de verificar que el intercambio de datos de usuario se lleva a cabo con las medidas de seguridad necesarias ya que este método exige comunicaciones entre varios nodos de la red al mismo tiempo.

Entonces, se determinó probar el mecanismo propuesto, sincronizando mediante un Middleware los nodos que comparten la información del *Repository Data* para un servicio dado dentro del IMS. Esta información es de gran importancia para la prestación de servicios personalizados, ya que los proveedores de servicio pueden ingresar en estos campos cualquier tipo de dato relacionado con los servicios que un usuario en particular consume. Los nodos que comparten estos datos son el HSS y los AS que prestan el servicio tal como se especifica en [28].

Además de los elementos mencionados anteriormente, se incluye en el prototipo un SEG, el cual es la entrada al dominio seguro y actúa como pasarela entre la aplicación que genera la información a ser sincronizada y la plataforma de replicación.

Todos estos dispositivos deben implementar las herramientas y procedimientos de seguridad del mecanismo propuesto para garantizar que la comunicación salto a salto sea asegurada mediante una cadena de túneles IPsec.

De esta manera la arquitectura del prototipo para la validación del mecanismo de seguridad IMSeg se muestra en la Figura 13.

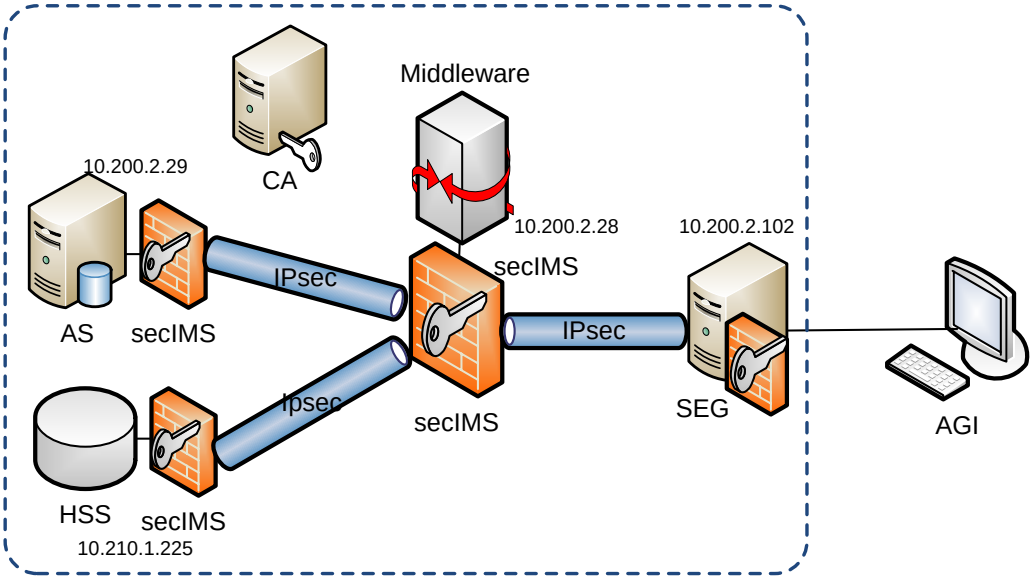

Figura 13. Prototipo para la validación del mecanismo de seguridad

3.6.2. Descripción del funcionamiento

A continuación se presenta una descripción del papel que juega cada uno de los módulos presentes en la arquitectura y de la forma como IMSeg interactúa con ellos.

- AS: El servidor de aplicaciones local, es el elemento a quien le interesa la información que será sincronizada. En este diseño se asume que en la misma máquina corren el AS y el motor de bases de datos al cual se pretende replicar la información de la tabla *repository data* del HSS. El AS tiene una conexión con la plataforma middleware mediante los controladores propios de esta.

  La implementación de IMSeg en el AS se realiza de la misma manera que en los otros nodos de la red, esta debe establecer una Asociación de Seguridad que garantice que la comunicación entre el AS y el middleware se lleve a cabo a través de un túnel seguro IPsec. El firewall local garantiza que únicamente exista comunicación entre el AS y el middleware mediante el puerto utilizado por el protocolo del motor de bases de datos del AS.

- HSS: Este es el contenedor principal de toda la información en la red IMS, al igual que en el caso del AS, el HSS se conecta a la plataforma *middleware* mediante los controladores que esta provea.

  Del mismo modo en el HSS debe implementarse el firewall local para que la conexión con el middleware sea solamente mediante el puerto del motor de bases de datos utilizado y las herramientas para establecer la Asociación de Seguridad y el túnel IPsec con el middleware.

- Plataforma *Middleware*: Proporciona la implementación de los protocolos de replicación para las bases de datos de una forma transparente. Como es el corazón de la red de intercambio de información, IMSeg debe ser instalado y configurado de forma cuidadosa, puesto que se establecen Asociaciones de Seguridad para la comunicación con las bases de datos AS y HSS además de los nodos de red que acceden a ellas como son el CSCF y el SEG, bloqueando también los puertos TCP/UDP innecesarios para las transacciones y administración de los servicios que presta.

- SEG: El Gateway de Seguridad proporciona el acceso al dominio IMS seguro. Todos los paquetes entrantes o salientes del dominio deben pasar a través de él. Tiene una conexión con el middleware por lo tanto debe establecer la respectiva Asociación de Seguridad y firewall local para garantizar los servicios de seguridad requeridos al interior del dominio.

  Para efectos de realizar pruebas y ser consistentes con la arquitectura planteada, en este nodo corre la aplicación *Gateway* desarrollada para el proyecto SincIMS, la cual recibe las actualizaciones de información provenientes de la AGI (mensajes HTTP) y las reenvía al middleware mediante el controlador que este provea, con el fin de replicar la información al HSS y al AS.

- Aplicación Generadora de Información (AGI): Aplicación desarrollada para el proyecto SincIMS, se decidió tenerlo en cuenta en este prototipo debido a su utilidad para realizar pruebas, puesto que es la encargada de generar la información que se pretende sea sincronizada entre el AS y el HSS. Es una representación de un servidor que genera información de los usuarios IMS ubicado fuera *Core* IMS, el cual representa al dominio seguro, por lo tanto esta aplicación no se conecta directamente con la plataforma *middleware* sino que lo hace mediante el *Gateway*, enviando las peticiones a través de mensajes HTTP.

  Este módulo se tiene en cuenta únicamente para tener una representación de un equipo que genera actualizaciones de información de usuario. Por consiguiente la conexión con el SEG no se encuentra asegurada, ya que gracias a la convergencia de redes el dispositivo podría tener uno de los diversos tipos de tecnologías de acceso al dominio IMS. Por lo tanto deben realizarse estudios individuales con el fin proveer seguridad en cada una de estas tecnologías, cosa que esta fuera del alcance de este proyecto.

- Autoridad Certificadora (CA): Este módulo representa a la entidad de confianza, encargada de emitir y revocar certificados digitales X.509 para cada uno de los equipos que implementarán autenticación basada en certificados con IPsec, la cual es necesaria para el establecimiento de una Asociación de Seguridad entre dos hosts, además es mucho más segura y confiable que la autenticación mediante claves pre compartidas.

  Cabe aclarar que en un entorno real se podría comprar dichos certificados a entidades acreditadas a nivel mundial como *VeriSign, GeoTrust, DigiCert o Certicamara* en el caso de Colombia. Sin embargo en una autenticación mediante IPsec es bastante seguro el hecho de poseer una entidad certificadora local de confianza ya que para establecer Asociaciones de Seguridad entre dos hosts la autenticación tiene que ser bidireccional, por lo tanto los certificados de una y otra parte deben ser emitidos por la misma entidad lo cual prácticamente anula la posibilidad de suplantarla y evita ataques de tipo *spoof* y *hombre en el medio*. A esto se suma el ahorro en trámites, costos y el hecho de poder renovar o cambiar los certificados en cualquier momento.

3.6.3. Elección de Herramientas para la implementación del mecanismo de seguridad.

Después de tener en claro el diseño del prototipo se realizó una exploración de las herramientas para la implementación de cada nodo de la red. En primera instancia se elije las herramientas para la red IMS de prueba, las cuales son independientes al mecanismo de seguridad. Luego se decidió cual sería el sistema operativo más idóneo para los nodos de la red. Por último se hizo una exploración de herramientas de seguridad que estén acordes con los requerimientos y se realiza la elección pertinente.

3.6.3.1. Herramientas de la red base:

En primera instancia las herramientas seleccionadas para la red sobre la cual se probará el mecanismo de seguridad serán las utilizadas en SincIMS:

- MIDDLEWARE: Este nodo hace uso de la herramienta *Sequoia*, un proyecto de Código abierto de la empresa *Continuent* [34]. Es una solución *middleware* muy completa capaz de manejar *clusters*, además de ofrecer mecanismos de balance de carga y de recuperación de fallos. Su implementación ha sido realizada 100% en lenguaje Java, lo cual garantiza su portabilidad a cualquier sistema que posea un JRE (*Java Runtime Environment*) superior al 1.4 así como la interoperabilidad con cualquier base de datos que provea un controlador JDBC (*Java Database Connectivity*). Entre sus principales ventajas se encuentra que para su implantación no es necesario hacer cambios sustanciales en las aplicaciones existentes, ya que únicamente basta con cambiar la URL (*Universal Resource Locator*) de conexión hacia la base de datos.

  El *core* del sistema está compuesto por controladores que implementan la tecnología RAIDb (*Redundant Array of Inexpensive Disks*), estos mantienen la sincronización del *clúster* haciendo uso de primitivas de *group communication.*

- *CORE* IMS: Open IMS *Core* [35], es una implementación del CSCF y HSS, realizada en el *Instituto Fraunhofer* de Alemania, la principal característica de esta iniciativa, como su nombre lo indica, es que está basada en software libre tal como OpenSER y MySQL y es usable bajo licencia GPLv2 (*General Public License*), la cual permite su libre modificación y distribución. Esta plataforma ha alcanzado suficiente madurez y es usada como cama de pruebas dentro del Open IMS Payground, un proyecto liderado por el grupo Fokus del instituto Fraunhofer, que ofrece servicios como implementación de soluciones IMS, consultoría, *benchmarking* entre otros.

  La implementación del Open IMS integra el CSCF basado en el servidor de aplicaciones SIP OpenSER además del HSS realizado en lenguaje Java y que incorpora el motor de bases de datos MySQL.

- SERVIDOR DE APLICACIONES: El Servidor SIP de BEA Weblogic [36] es un servidor convergente para aplicaciones Java EE (*Enterprise Edition*), IMS-SIP y SOA, lo que quiere decir que provee un contenedor con soporte integrado para los estándares Java, de Servicios Web e IMS. Posee una arquitectura sencilla de configurar y tiene la capacidad de utilizar eficientemente los recursos hardware disponibles.

- AGI: La Aplicación Generadora de Información, desarrollada para pruebas en el proyecto SincIMS consiste en una aplicación java, la cual pretende simular un servidor que genera información de los usuarios IMS. Esta aplicación puede ser configurada mediante una interfaz, siendo posible establecer parámetros como el número de usuarios para los que se está actualizando su información y la frecuencia con que estas actualizaciones ocurren [6].

- GATEWAY: Aplicación desarrollada en JAVA para el proyecto SincIMS. Su función es realizar la traducción de los mensajes HTTP enviados por la AGI en los correspondientes mensajes que son enviados al controlador SEQUOIA para realizar el proceso de sincronización de información [6]. Para el prototipo de IMSeg se despliega sobre un servidor de aplicaciones BEA Weblogic corriendo en el SEG.

3.6.3.2. Sistema Operativo:

Se decidió que todos los nodos del prototipo tengan instalado el sistema operativo Debian GNU/Linux 5 [37]. Algunas de las razones para elegirlo son las siguientes:

- Código Abierto: Ventaja gracias a la cual es posible realizar todo tipo de pruebas y modificaciones sin preocuparse por licencias ni costos.

- Robustez: Debian en su versión estable es una de las más robustas distribuciones de Linux, gracias a que su gran comunidad de desarrolladores invierten mucho esfuerzo en el mantenimiento permanente de paquetes. Además se necesita atravesar por un estricto proceso de revisiones para que un paquete llegue a ser estable.

- Seguridad: Las configuraciones por defecto de Debian poseen un alto grado de seguridad por lo cual es difícil que se abran agujeros de seguridad serios. Lo cual es una gran ventaja para un proyecto como el presente.

- El kernel estable de Linux (2.6), cuenta con una implementación de una pila IPsec nativa. Como soporta el protocolo estándar de gestión de claves, la pila IPsec de Linux puede utilizarse en conjunto con otros demonios gestores de claves.

- Gestión de paquetes: El sistema de gestor de paquetes de Debian es el más versátil y flexible de las distribuciones Linux, gracias a su facilidad de uso y potencia.

- Todas las aplicaciones mencionadas en el apartado anterior pueden instalarse en este sistema operativo.

3.6.3.3. Herramientas de seguridad

Debido a que se trabajará con un sistema operativo Linux, se exploraron las mejores opciones que sean compatibles con este sistema.

Herramientas para la implementación de IPsec en Linux: Gracias a que en los últimos años IPsec ha tomado gran popularidad entre los sistemas operativos basados en Unix, existen varias herramientas para implementarlo en Linux. A continuación se mencionan las más destacadas y posteriormente se hace la elección de la que mejor se adecúa a las necesidades del proyecto.

- Openswan [38]

  Openswan es una completa implementación de IPsec para los kernels 2.0, 2.2, 2.4 y 2,6 de Linux. Este proyecto comenzó como una parte del ahora abandonado proyecto FreeS/WAN, continuando bajo la licencia GNU (*General Public License*), sin embargo a diferencia de FreeS/WAN no es desarrollado exclusivamente para el sistema operativo Linux. Consiste en una pila IPsec de núcleo, junto con un demonio (*pluto*) y muchos scripts de *Shell*.

Desarrollado específicamente para construir VPN (*Virtual Private Network*) corporativas para proveer acceso remoto seguro. Entre sus ventajas principales está el hecho de que soporta certificados X.509 para la autenticación, se puede configurar en "Modo Agresivo", Además es capaz de interoperar con otros sistemas operativos como la familia BSD, Windows y Mac OSX.

Su mayor desventaja es el hecho de aun no cuenta con soporte para IKEv2, lo cual hace su configuración más compleja y le impide poseer nuevas características como el manejo sencillo de claves y *NAT-Traversal (Network Address Translation Traversal)*.

- StrongSwan [39]

  StrongSwan es una completa solución VPN de código abierto para el sistema operativo Linux. El diseño modular de las últimas versiones de strongSwan implementa completamente el protocolo IKEv2 así como Intercambio de Autenticación Múltiple.

  Es otro de los descendientes del proyecto FreeS/WAN, se mantiene de manera activa en el *Institute for Internet Technologies and Applications en Rapperswil*, Suiza por Andreas Steffen, profesor de Seguridad en Comunicaciones. El objetivo del proyecto strongSwan es fortalecer los mecanismos de autenticación usando una infraestructura de llaves publicas y certificados X.509. Opcionalmente se provee almacenamiento de llaves privadas en tarjetas inteligentes a través de la interfaz estandarizada PKCS (Public-Key Cryptography Standards) y soporta listas de revocación de certificados OCSP (*Online Certificate Status Protocol*) [40].

  Una característica única es el uso de atributos en certificados X.509 con el fin de implementar esquemas de control de acceso avanzado basado en la pertenencia a grupos. Otra de sus ventajas es su interoperabilidad con la mayoría de implementaciones IPsec, incluidos varios clientes para VPN Windows y Mac OSX.

- Racoon2 [41]

  Racoon2 es desarrollado y mantenido por el proyecto WIDE, un esfuerzo conjunto de varios investigadores provenientes de organizaciones japonesas como *Fujitsu, Toshiba* e *Hitachi* entre otras. Los cuales iniciaron con el proyecto KAME, que implementó soporte IPsec completo para NetBSD y FreeBSD. Su demonio de gestión de claves se llamó *racoon* con soporte para IKEv1 [42].

  Liberado bajo licencia estilo BSD, Racoon2 es un sistema muy completo para intercambiar e instalar parámetros de seguridad para IPsec, provee una implementación de un sistema de administración de llaves soportando múltiples protocolos para el intercambio de las mismas. Soporta los protocolos IKEv1, IKEv2 y KINK (*Kerberized Internet Negotiation of Keys*).

  Racoon fue desarrollado originalmente para FreeBSD y NetBSD, uno de los mayores avances de racoon2 es su integración al kernel 2.6 de Linux. Por su practicidad y robustez ha sido adoptado en diversos entornos empresariales convirtiéndose en una

de las implementaciones IPsec más populares, con una gran comunidad de usuarios y colaboradores en todo el mundo.

- OpenIKEv2 [43]

  OpenIKEv2 fue la primera implementación de código abierto del protocolo IKEv2. Desarrollada enteramente en el lenguaje C++ por el grupo ANTS de la Universidad de Murcia, España.

  Su principal aporte es el desarrollo de las librerías de código abierto: libopenikev2, que implementa las funcionalidades del *core* IKEv2 y libopenikev2_impl que provee algunas implementaciones de interfaces para libopenikev2. Estas librerías pueden usarse por cualquier programador para desarrollar implementaciones a la medida mediante la API de libopenikev2_impl, la cual permite utilizar el protocolo IKEv2 dentro de otras aplicaciones sin correr ningún proceso adicional. Esto conlleva una manera más elegante de utilizar el protocolo IKEv2 en varios escenarios avanzados que necesiten de este.

  La aplicación OpenIKEv2 utiliza las librerías Libopenikev2 y Libopenikev2_impl. Tiene características similares a otras implementaciones IKEv2 como racoon2 y strongSwan como: autenticación mediante claves compartidas y certificados, implementación de modo túnel y transporte, soporte para ipv6, manejo sencillo de claves, etc.

  Entre sus desventajas está el hecho de que la característica *NAT-Traversal* no se ha desarrollado completamente y sus implementaciones en sectores productivos son escasas, puesto que ha servido mucho más para impulsar nuevos desarrollos que han utilizado sus librerías, por tanto no cuenta con una gran comunidad de usuarios ni posee documentación abundante como es el caso de otros desarrollos.

Elección de herramienta IPsec: Después de analizar las herramientas para la implementación de IPsec se concluyó que las mejores opciones son strongSwan y Racoon2 debido a las características de robustez, soporte para IKEv2 e interoperabilidad. Por tanto se decidió utilizar strongSwan en el prototipo, debido a su capacidad de incorporar autenticación mediante certificados y a que incorpora herramientas de línea de comandos que permiten monitorear el estado de las Asociaciones de Seguridad y conexiones establecidas, además existe una comunidad de usuarios considerable en entornos de producción, por lo tanto la documentación de implementaciones exitosas es mayor.

Herramientas firewall

- Iptables [44]

  Iptables es un sistema de firewall desarrollado bajo el proyecto Netfilter/iptables integrado al kernel de Linux, que se ha extendido enormemente a partir del kernel 2.4 de este sistema operativo y está presente en prácticamente todas las distribuciones de Linux.

Está conformado por un conjunto de herramientas que le permiten al usuario enviar mensajes al kernel, el cual maneja todos los paquetes TCP/IP que entran o salen del sistema, esto permite a iptables decirle al kernel qué debe hacer con cada uno de los paquetes basándose en las características de un paquete en particular.

Iptables permite al administrador del sistema definir reglas acerca de qué hacer con los paquetes de red. Las reglas se agrupan en cadenas: cada cadena es una lista ordenada de reglas. Las cadenas se agrupan en tablas: cada tabla está asociada con un tipo diferente de procesamiento de paquetes.

Entonces, iptables es una forma de indicarle al kernel algunas cosas que debe hacer con cada paquete, dependiendo de la regla en la cual es identificado el paquete y de donde provenga será posible: aceptar/negar su salida, aceptar/negar su entrada o aceptar/negar su reenvío.

Además iptables permite realizar traducción de direcciones de red (NAT) para IPv4 y mantener registros de log. Todas estas razones lo han convertido en la herramienta estándar de todas las distribuciones modernas de GNU/Linux, además es la base de las herramientas firewall para escritorio como: *Firestarter, Bastion-firewall, Firewall Builder*, las cuales permiten configurar reglas con iptables mediante una interfaz gráfica.

Para la implementación del prototipo se decidió utilizar scripts de Shell iptables, los cuales son una serie de órdenes de línea de comandos que van ejecutando las reglas del firewall. Este tipo de configuración aunque es compleja, brinda mayor control sobre las reglas que se aplican a cada paquete que entra o sale del sistema. Además debido a que se está trabajando en un entorno de servidores Linux, generalmente estos no cuentan con entorno de escritorio, por tanto no es posible instalar herramientas con interfaz gráfica.

- TCP Wrappers [45]

  Es una herramienta libre muy útil para el filtrado de paquetes IP en sistemas operativos tipo UNIX, escrita por Wieste Venema de la *Universidad Tecnológica de Eindhoven*, Países Bajos. Provee control específico sobre algunos servicios de red como SSH o TELNET, decidiendo cuales hosts pueden acceder a ellos. También hace uso del la herramienta *syslog* para guardar reportes del uso local de la red. Su mayor ventaja es la sencillez de configuración, ya que solo hace falta añadir reglas simples tipo ACL (*Access List Rules*) en archivos de configuración.

  En el prototipo del proyecto se usa esta herramienta en conjunto con Iptables para añadir una capa adicional de protección a servicios como SSH con el fin de asegurar la administración remota de los servidores y generar logs para el monitoreo de la red.

Herramienta para la Autoridad Certificadora

OpenSSL [46]

Este proyecto es un esfuerzo colaborativo para desarrollar un conjunto de

herramientas que implementen los protocolos SSL y TLS de calidad comercial, robusto y de código abierto, así como una potente librería criptográfica de propósito general.

El proyecto es desarrollado y mantenido por una comunidad mundial de voluntarios y se ha convertido en el más popular de las implementaciones que implementan SSL, siendo base de muchos proyectos de código abierto que utilizan sus librerías como: *apache-ssl, bind, openssh, sendmail, javassl, samba, postfix*, entre muchos otros [47].

OpenSSL incorpora varios scripts que permiten crear una Autoridad Certificadora auto-firmada, por lo tanto es ideal para este modulo del prototipo.

CAPÍTULO 4: IMPLEMENTACIÓN DEL PROTOTIPO DE SEGURIDAD PROPUESTO

4.1. Introducción

En el capítulo 3 se llegó al diseño y elección de herramientas para el prototipo del mecanismo de seguridad. El presente capítulo parte del anterior tomando los elementos que allí se describen y mostrando los pasos más sobresalientes en la configuración y puesta a punto del prototipo planteado.

Para llevar a cabo esta implementación en principio se mencionan algunos ajustes a la instalación de la red base IMS sobre la cual se probará el mecanismo, posteriormente se describen los procesos para la configuración de seguridad.

Para esto inicialmente se somete a los servidores a algunos prácticas de *endurecimiento*, luego se analiza el escenario para poder crear los scripts de firewall asegurando que el intercambio de información entre los equipos sea únicamente el necesario para el buen funcionamiento de la red IMS y posteriormente se instalan y configuran las herramientas IPsec para minimizar la inseguridad en la comunicación entre los nodos.

4.2. Configuración de la Red Base

Los procesos de instalación de las herramientas de la red base se presentan con más detalle en el trabajo SincIMS [30], de esta forma en las secciones siguientes sólo se realiza una descripción de los pasos de configuración necesarios para el correcto funcionamiento de cada módulo de la arquitectura presentada en 3.5.

4.2.1. Prerrequisitos

La topología de red del prototipo requiere al menos cuatro hosts, se contó con dos equipos para la realización del proyecto, por lo tanto hubo la necesidad de instalar dos máquinas virtuales que si bien comparten los recursos hardware de los anfitriones, son estaciones lógicamente individuales con asignación propia de disco duro, memoria RAM y tarjeta de red.

Para tener claridad en los ejemplos de configuración que se mostrarán en las siguientes secciones, la Tabla 1 muestra una breve descripción de las funcionalidades de los equipos en la red base y las herramientas necesarias:

Tabla 1. Descripción de los equipos utilizados en el prototipo

Nombre del equipo	Dirección IP	Nodo de la red	Utilidades de la red base
fenix	10.210.1.225	HSS, AGI	Open IMS Core Rev. 570, AGI, Estación de trabajo del administrador de la red
xeratul	10.200.2.28	Middleware	Sequoia V2.10
zealot	10.200.2.102	SEG	Bea Weblogic V3.1, Gateway
dragoon	10.200.2.29	AS	MySQL Server 5.05, Bea Weblogic V3.1

Para realizar la configuración y despliegue de las aplicaciones del prototipo se debieron tener instaladas las herramientas mencionadas en la tabla anterior.

Por comodidad en los ejemplos de configuración que se verán se tienen en cuenta los nombres de los equipos (*fenix, xeratul, zealot y dragoon*) más que sus direcciones IP.

En cada host se instaló el sistema operativo Debían GNU/Linux 5.0, la más reciente versión estable de esta distribución.

4.2.2. Sequoia

La configuración de Sequoia (instalado en *xeratul*) para el caso particular del prototipo se realiza modificando los archivos de configuración controller.xml y mysql.xml.

En el archivo controller.xml lo más importante es incluir la dirección IP de la interfaz donde

```
<Controller ipAddress="10.200.2.28" port="25322">
```

el controlador escuchará las peticiones:

El archivo mysql.xml corresponde a la configuración de la base de datos virtual, en esta se definen entre otros parámetros los datos de conexión a las bases de datos reales que pertenecen clúster, las características de replicación y recuperación en caso de fallos por ejemplo:

```
<DatabaseBackend name="as" driver="com.mysql.jdbc.Driver"
   url="jdbc:mysql://10.200.2.29:3306/user_data"
      driverPath="/usr/share/java/mysql-connector-java-5.1.6.jar"
      connectionTestStatement="select 1">
      <ConnectionManager          vLogin="user"          rLogin="userd"
      rPassword="contraseña-segura">
            <VariablePoolConnectionManager initPoolSize="40"/>
      </ConnectionManager>
</DatabaseBackend>
```

Aquí se está definiendo la conexión para el usuario userd al *backend* "as" el cual corresponde a una base de datos MySQL llamada user_data alojada en el equipo con dirección IP 10.200.2.29.

4.2.3. Open IMS Core

Antes de iniciar el FHoSS (FOKUS HSS) Es necesario configurarlo para que el acceso a su base de datos (MySQL) sea controlado mediante el Middleware Sequoia. Teniendo en cuenta que el desarrollo del FHoSS utiliza el patrón de acceso a datos de Hibernate, basta con añadir las siguientes líneas en el inicio del archivo /opt/OpenIMSCore/FHoSS/deploy/hibernate.properties:

```
## Sequoia
hibernate.dialect                 org.hibernate.dialect.MySQLDialect
hibernate.connection.driver_class org.continuent.sequoia.driver.Driver
hibernate.connection.username     user
hibernate.connection.password     contraseña-segura
hibernate.connection.url        jdbc:sequoia://10.210.1.225/repository
```

Una vez realizado esto, se debe proporcionar al Open IMS Core el controlador de Sequoia para que a través de este realice las conexiones a su base de datos. Para realizar esta tarea es necesario copiar el controlador de Sequoia en la carpeta /opt/OpenIMSCore/FHoSS/lib de la instalación del OpenIMS Core, ya que aquí es donde por defecto el va a buscar cualquier librería externa.

Por último se debe permitir el acceso externo a la base de datos, esto con el fin de garantizarle al controlador SEQUOIA un adecuado manejo del *backend* denominado HSS.

Para realizar esto es necesario modificar la configuración que por defecto utiliza MySQL la cual se encuentra en el archivo /etc/mysql/my.cfn y cambiar `BindAddress` para que quede así:

```
BindAddress 10.200.2.28
```

Esta última es la dirección IP correspondiente a la interfaz a la cual se conecta el controlador. Una vez realizado esto basta con reiniciar el servicio de MySQL e iniciar el FHoSS.

4.2.4. Aplicación Gateway

Esta aplicación simplemente traduce los mensajes HTTP enviados por la AGI en mensajes Sequoia por tanto antes de desplegarla en el servidor de aplicaciones se deben cambiar los datos de conexión con el controlador Sequoia. Para el caso del prototipo estos datos son:

Dirección IP del controlador: 10.200.2.28
Nombre de la base de datos virtual: repository
Nombre de usuario: user
Contraseña: contraseña-segura

4.2.5. Aplicación AGI

La aplicación AGI sólo debe modificarse para cambiar los datos de conexión con Gateway de la siguiente manera:

Dirección IP del Gateway: 10.200.2.102
Nombre de la aplicación Gateway: SEG

4.3. Configuración de seguridad

4.3.1. Endurecimiento de Servidores Linux

Antes de iniciar con configuraciones de Iptables e IPsec, es muy importante implementar algunas prácticas de seguridad en el entorno de trabajo de servidores GNU/Linux Debian, esto se conoce como Endurecimiento de Servidores, pues aunque el intercambio de información entre los nodos sea seguro, si un intruso llega a encontrar y explotar un fallo de seguridad en un servidor, esto pone en riesgo evidente tanto los datos que almacena ese nodo como la información que circula a través de él.

Existen muchas prácticas de seguridad, así como varios niveles de endurecimiento para un servidor Debian Linux, por tanto en este apartado se mencionan algunos de los procedimientos más relevantes llevados a cabo en cada uno de los nodos del prototipo [48].

4.3.1.1. Actualización del sistema

Actualizar el sistema de forma periódica evita que el servidor se vea afectado por *bugs* conocidos y permite utilizar nuevas características y mejoras de los paquetes instalados. En distribuciones basadas en Debian es necesario contar con una adecuada configuración de repositorios, lo cual se hace editando el archivo /etc/apt/sources.list. Una vez hecho esto se actualiza el sistema mediante los comandos mostrados en las Figuras 14 y 15:

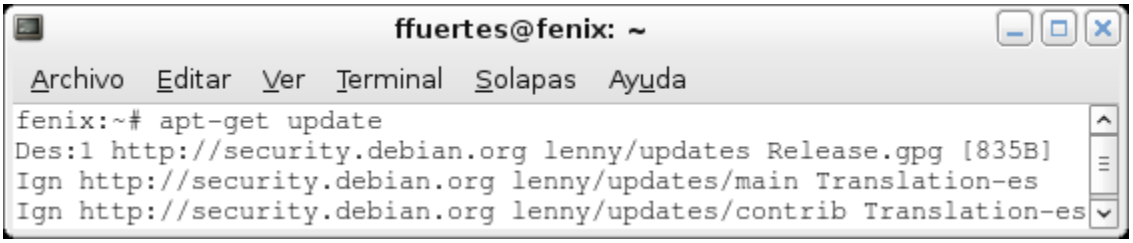

Figura 14. Actualización de la base de datos local de paquetes

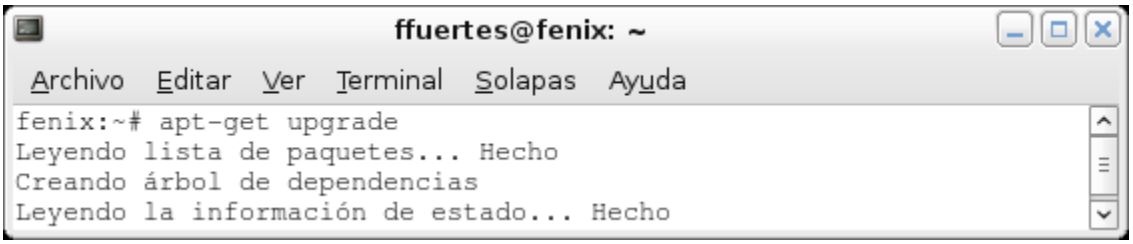

Figura 15. Actualización de paquetes instalados

4.3.1.2. Evitar suministrar Información del servidor

Existen algunos archivos que brindan información sistema cuando un usuario inicia sesión por medio de SSH, por tanto es necesario borrar, comentar o modificar su contenido para evitar suministrar datos del sistema. En el prototipo se borraron todas las líneas de los siguientes archivos:

/etc/motd: mensaje de inicio de sesión.
/etc/motd.tail: mensaje de inicio de sesión.
/etc/issue: mensaje anterior a la solicitud de login.

4.3.1.3. Deshabilitar Servicios Innecesarios

En un servidor debe instalarse lo absolutamente necesario y eliminar o deshabilitar procesos o servicios que no sean fundamentales para el correcto funcionamiento del mismo, en Debian el súper demonio *inetd* puede iniciar varios servicios a la vez como POP, IMAP, FTP, etc. Por lo tanto en la instalación estándar se deben comentar todas las líneas del archivo de configuración de este demonio ubicado en: /etc/inetd.conf.

Existen otros servicios instalados con el sistema base, que no se necesitan en el servidor

estándar pero que se inician automáticamente. El directorio /etc/rc2.d/ contiene los enlaces simbólicos hacia los scripts que inician dichos servicios, si el nombre de uno de estos enlaces comienza con la letra “S” el servicio se iniciará con el sistema, si comienza con la letra “K” deberá ser iniciado manualmente. Por lo tanto se deben cambiar los nombres de servicios como *cups* (*Common Unix Printing System*) o *nfs* (*Network File System*) como se muestra en la Figura 16.

Figura 16. Deshabilitar el inicio por defecto del servicio *cups*

4.3.1.4. Asegurar la administración remota

La forma más conveniente de administrar remotamente un servidor Linux es utilizar SSHv2, pero existen varias vulnerabilidades que pueden hacerlo inseguro, por tanto después de instalar este servicio es importante realizar algunos ajustes. En primer lugar modificar ciertas líneas del archivo de configuración del servicio /etc/ssh/sshd_config de la

```
Port 2022
#Evitar los ataques con scripts al puerto por defecto.

PermitRootLogin no
#Obliga a tener una cuenta de usuario local y a entrar en ella, antes
#de acceder como root (Administrador del sistema)

AllowUsers ffuertes
#El único usuario autorizado para iniciar sesión remotamente es
#ffuertes

X11Forwarding no
#Un servidor generalmente no debe tener instalado interfaces de
#ventanas, por lo tanto esto no se requiere.
```

siguiente forma:

Como se mencionó anteriormente la herramienta TCP Wrappers permite filtrar el acceso de red a ciertos servicios, en principio puede realizarse una configuración estricta donde se niegue todo acceso a la maquina menos el de SSH proveniente de la IP del equipo del administrador del servidor. Dependiendo de los servicios instalados, esta restricción puede modificarse posteriormente para tener acceso a los puertos necesarios desde las ubicaciones autorizadas para una determinada conexión.

Los archivos de configuración de TCP Wrappers son /etc/hosts.allow en el cual se configuran las ACL con los servicios y hosts permitidos y /etc/hosts.deny donde están las ACL que niegan el acceso. Durante la instalación de cada nodo del prototipo se establece que tenga acceso por SSH únicamente el equipo del administrador. Para esto se agregan las siguientes líneas en los respectivos archivos de configuración:

```
sshd : 10.210.1.225 : spawn /bin/echo `/bin/date` acceso satisfactorio
desde %h usuario %u >> /var/log/ssh.log : ALLOW
```

En /etc/hosts.allow:

Con esta línea se permite el acceso por SSH al host con dirección IP 10.210.1.225 y se guarda un registro con los datos de esta conexión en /var/log/ssh.log.

```
sshd : ALL : spawn /bin/echo `/bin/date` intento de acceso no
autorizado desde %h >> /var/log/ssh.log : DENY
```

En /etc/hosts.deny:

Con esta línea se niega todo acceso por SSH al equipo y se guarda un registro de los intentos de inicio de sesión en /var/log/ssh.log.

4.3.1.5. Endurecimiento del kernel

Proteger al kernel de Linux contra ataques conocidos es una tarea que impone un alto nivel de seguridad en cualquier servidor, esto se consigue compilando un kernel a la medida, habilitando en este proceso diferentes opciones de seguridad avanzadas. En el anexo A se detallan los pasos para el endurecimiento del Kernel de Debian utilizando el parche Grsecurity [49]; este procedimiento se siguió en los cuatro hosts como se puede corroborar en el mismo anexo.

Durante la implementación del prototipo de IMSeg se tuvieron en cuenta otras tareas de seguridad como la administración adecuada de permisos en carpetas y archivos del sistema, restricción de shell de usuarios, manejo de contraseñas seguras y un particionado adecuado. Algunos otros procedimientos como *enjaulamiento* de usuarios y procesos, manejo de una infraestructura de logs detallada, implementación de sistemas de backups, etc. son recomendados para mantener servidores lo menos inseguros posible.

4.3.2. Configuración del firewall local

La configuración del firewall en cada nodo se realiza por medio de scripts tipo bash que ejecutan líneas con de comando *Iptables* para filtrar paquetes IPv4 de acuerdo a la

planeación de las reglas o políticas necesarias del firewall en cada equipo. En la siguiente sección se establecen dichas políticas dependiendo de los servicios que presta cada uno de los host pertenecientes al prototipo de IMSeg.

La Figura 17. muestra en detalle a cada nodo del prototipo con los puertos TCP o UDP necesarios para brindar los servicios de la red IMS. También se especifican las conexiones que se establecen entre estos indicando con las flechas el sentido de cada conexión y el puerto al que se accede (Por ejemplo: el host que corre el AGI se conecta al puerto 7001 del host SEG).

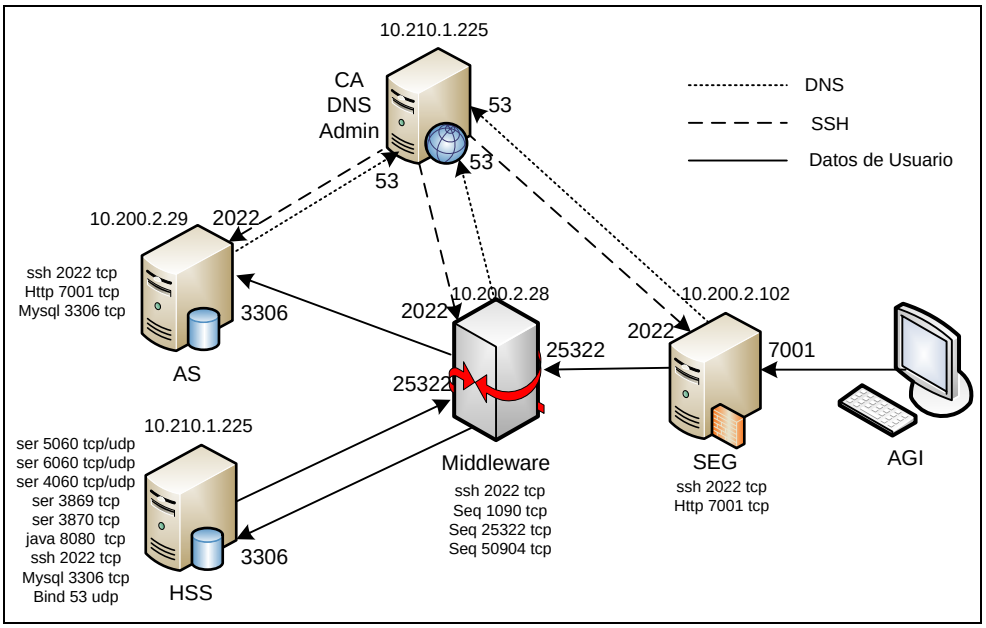

Figura 17. Conexiones detalladas entre nodos

Como se aprecia en la Figura anterior el host con dirección IP 10.210.1.225 *(fenix)* corre tanto OpenIMS Core (donde se encuentra el nodo HSS), como la aplicación AGI, además es el equipo central de administración de la red (Admin) el cual tiene privilegios de ingresar mediante SSH a los otros tres equipos. Debido a que Open IMS Core necesita instalar un servidor DNS local, por practicidad este servicio también está disponible para los otros nodos de la red.

Para la Autoridad de Certificación se usó un equipo externo (*akira*), debido a que los certificados X.509 deben estar firmados por una entidad que sea diferente a los host que implementen los túneles IPsec. Estos se copiaron a *fenix* ya que este tiene acceso a los demás equipos mediante el protocolo SCP (*Secure Copy*), el cual hace uso de SSH para asegurar la conexión, desde ahí se distribuyen mediante el mismo protocolo a los demás host.

Las conexiones de los protocolos ESP e IKE no se muestran en la figura pero serán necesarias cuando se establezcan los túneles IPsec como se verá en la sección 4.3.3. Permitir por defecto ESP desde y hacia cualquier dirección no genera inseguridad, ya que este se utiliza únicamente en conexiones cifradas establecidas mediante un riguroso proceso de autenticación llevado a cabo por IKE (puertos 500 udp y 4500 tcp/udp), por lo

tanto, además de ESP, se debe permitir estos puertos de entrada y salida en todos los equipos.

Cabe aclarar que en la Figura 17. únicamente se tienen en cuenta los servicios necesarios para el prototipo, en un ambiente real se pueden tener más servicios habilitados y conexiones más complejas, todo depende de las necesidades y capacidades del proveedor. Para el análisis de la seguridad en el intercambio de datos de usuario es suficiente con lo mostrado en la figura ya que se establecen las conexiones básicas que se tendrían en un entorno de producción.

4.3.2.1. Planeación de políticas de firewall para los nodos

Una vez establecidos los servicios y el sentido de la comunicación entre los nodos, es posible plantear las políticas del firewall para cada equipo. Un firewall seguro se consigue si la política por defecto es DENEGAR, de este modo únicamente se aceptan conexiones con equipos permitidos en puertos específicos. Las políticas generales para cada host en estado operativo quedarían del siguiente modo:

dragoon (AS) 10.200.2.29:

- Política por defecto DENEGAR.
- Aceptar nuevas conexiones desde 10.200.2.28 en el puerto local 3306 tcp
- Aceptar nuevas conexiones desde 10.210.1.225 en el puerto local 2022 tcp
- Permitir iniciar una conexión hacia 10.210.1.225 al puerto destino 53 udp
- Permitir el ping desde 10.210.1.225
- Aceptar nuevas conexiones de IKE y ESP
- Permitir iniciar nuevas conexiones IKE y ESP

xeratul (Middleware) 10.200.2.28:

- Política por defecto DENEGAR
- Aceptar nuevas conexiones desde 10.210.1.225 en el puerto local 2022 tcp
- Aceptar nuevas conexiones desde 10.200.2.102 en el puerto local 25322 tcp
- Aceptar nuevas conexiones desde 10.210.1.225 en el puerto local 25322 tcp
- Permitir iniciar una conexión hacia 10.200.2.29 al puerto destino 3306 tcp
- Permitir iniciar una conexión hacia 10.210.1.225 al puerto destino 3306 tcp
- Permitir iniciar una conexión hacia 10.210.1.225 al puerto destino 53 udp
- Permitir el ping desde 10.210.1.225
- Aceptar nuevas conexiones de IKE y ESP
- Permitir iniciar nuevas conexiones IKE y ESP

zealot (SEG) 10.200.2.102:

- Política por defecto DENEGAR
- Aceptar nuevas conexiones desde 10.210.1.225 en el puerto local 2022 tcp
- Aceptar nuevas conexiones desde 10.210.1.225 en el puerto local 7001 tcp
- Permitir iniciar una conexión hacia 10.200.2.28 al puerto destino 25322 tcp
- Permitir iniciar una conexión hacia 10.210.1.225 al puerto destino 53 udp
- Permitir el ping desde 10.210.1.225

- Aceptar nuevas conexiones de IKE y ESP
- Permitir iniciar nuevas conexiones IKE y ESP

fenix (HSS, Admin, AGI, DNS) 10.210.1.225:

- Política por defecto DENEGAR
- Aceptar nuevas conexiones desde 10.200.2.28 en el puerto local 3306 tcp
- Permitir iniciar una conexión hacia 10.200.2.28 al puerto destino 2022 tcp
- Permitir iniciar una conexión hacia 10.200.2.29 al puerto destino 2022 tcp
- Permitir iniciar una conexión hacia 10.200.2.102 al puerto destino 2022 tcp
- Permitir iniciar una conexión hacia 10.200.2.28 al puerto destino 25322 tcp
- Permitir iniciar una conexión hacia 10.200.2.102 al puerto destino 7001 tcp
- Aceptar nuevas conexiones desde 10.200.2.28 en el puerto local 53 udp
- Aceptar nuevas conexiones desde 10.200.2.29 en el puerto local 53 udp
- Aceptar nuevas conexiones desde 10.200.2.102 en el puerto local 53 udp
- Permitir realizar ping hacia 10.200.2.28
- Permitir realizar ping hacia 10.200.2.29
- Permitir realizar ping hacia 10.200.2.102
- Aceptar nuevas conexiones de IKE y ESP
- Permitir iniciar nuevas conexiones IKE y ESP

Nota: En estado de instalación o mantenimiento de los servidores pueden existir necesidades de conexión a otros equipos como puerta de enlace o proxy para actualización de repositorios, instalación de paquetes, etc.

4.3.2.2. Configuración de Iptables

Un script de Iptables consiste en una serie de reglas ordenadas las cuales examinan si un paquete que entra o sale del host coincide con ciertas características, si es así le dicen al kernel que le aplique una acción determinada por la regla, la cual puede ser: aceptar (lo deje pasar), denegar (lo descarte), rechazar, marcar o modificar.

Existen 3 tipos de tablas en Iptables: *nat, filter y mangle*, estas contienen ciertas cadenas que a su vez contienen las reglas que se crean mediante el comando `iptables`.

Las reglas para los equipos del prototipo utilizan solamente la tabla *filter* pues esta permite filtrar paquetes. Las tablas *nat* (que permite cambiar direcciones IP fuente y destino) y *mangle* (que implementa algunas características de modificación de paquetes) no son necesarias para aplicar las políticas definidas en el apartado anterior.

Dentro de la tabla *filter* se tienen las cadenas INPUT, OUTPUT y FORWARD, de estas se usan en el prototipo las dos primeras ya que dentro de ellas se crean las reglas que permiten o niegan el ingreso o la salida de paquetes los cuales tienen como fuente o destino el propio equipo. La cadena FORWARD examina paquetes que pasan a través del host pero no van dirigidos a este, por tanto no es necesaria.

Como elementos adicionales en los scripts se incluyen reglas `arp` para generar entradas estáticas en la tabla ARP de cada equipo con las direcciones IP y MAC conocidas, con el fin de evitar ataques tipo Hombre en el Medio. Además, se incluyen reglas para registrar

en logs propios de Iptables, todas las interacciones de red en los equipos, tanto las permitidas (etiquetadas como `IMSeg_ACEPTADO`), como las bloqueadas o rechazadas (etiquetadas como `IMSeg_RECHAZADO`).

A manera de ejemplo, el script del firewall para el equipo *xeratul* según las políticas definidas anteriormente, utilizando la versión de Iptables 1.4.2 (instalada por defecto en Debian 5) es el siguiente:

```
#!/bin/sh

## Firewall local de xeratul

#Agregamos las entradas estáticas en la tabla ARP
arp -s 10.200.2.29 00:0c:29:90:4b:65
arp -s 10.210.1.225 00:12:3f:6f:a2:b4
arp -s 10.200.2.102 00:50:da:c6:1e:fe

# Limpiar reglas anteriores
iptables -F
iptables -X
iptables -Z
iptables -t nat -F

## Establecemos política por defecto: DROP!!!
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

## Definimos la cadena para registrar el tráfico aceptado
iptables -N LOGACCEPT
iptables -A LOGACCEPT -j LOG --log-level debug --log-prefix
"IMSeg_ACEPTADO "
iptables -A LOGACCEPT -j ACCEPT

## Definimos la cadena para registrar el tráfico rechazado
iptables -N LOGDROP
iptables -A LOGDROP -j LOG --log-level debug --log-prefix
"IMSeg_RECHAZADO "
iptables -A LOGDROP -j DROP

#Permitir conexiones de la misma maquina
iptables -t filter -A INPUT -i lo -j ACCEPT
#Permitir conexiones establecidas de entrada y salida
iptables -t filter -A INPUT -m state --state RELATED,ESTABLISHED -j
ACCEPT
iptables -t filter -A OUTPUT -m state --state RELATED,ESTABLISHED -j
ACCEPT

#Permitir ESP para conexiones IPsec Establecidas
iptables –t filter –A INPUT –p ESP –j ACCEPT
```

```
iptables –t filter –A OUTPUT –p ESP –j ACCEPT

#Permitir IKE con dragoon
iptables –t filter –A INPUT –s 10.200.2.29 –p udp –m multiport –dport
500,4500 –m state --state NEW –j LOGACCEPT

iptables –t filter –A OUTPUT –d 10.200.2.29 –p udp –m multiport –sport
500,4500 –m state --state NEW –j LOGACCEPT

iptables –t filter –A INPUT –s 10.200.2.29 –p tcp –dport 4500 –m state
--state NEW –j ACCEPT

iptables –t filter –A OUTPUT –d 10.200.2.29 –p udp –sport 4500 –m state
--state NEW –j LOGACCEPT

#Permitir Acceso ssh desde fenix
iptables -t filter -A INPUT -s 10.210.1.225 -p tcp --dport 2022 -m
state --state NEW -j LOGACCEPT

#Permitir Acceso a sequoia desde fenix
iptables -t filter -A INPUT -s 10.210.1.225 -p tcp --dport 25322 -m
state --state NEW -j LOGACCEPT

#Permitir acceso a sequoia desde zealot
iptables -t filter -A INPUT -s 10.200.2.102 -p tcp --dport 25322 -m
state --state NEW -j LOGACCEPT

#Permitir iniciar conexión a MySQL en dragoon
iptables -t filter -A OUTPUT -d 10.200.2.29 -p tcp --dport 3063 -m
state --state NEW -j LOGACCEPT

#Permitir iniciar conexión a MySQL en fenix
iptables -t filter -A OUTPUT -d 10.200.2.29 -p tcp --dport 3063 -m
state --state NEW -j LOGACCEPT

#Permitir ping desde fenix
iptables -t filter -A INPUT -s 10.210.1.225 -p ICMP -m state --state
NEW -j LOGACCEPT

#Permitir hacer consultas al servicio DNS
iptables -t filter -A OUTPUT -d 10.210.1.225 -p udp --dport 53 -m state
--state NEW -j LOGACCEPT

#Finalmente Rechazar y registrar todo el trafico restante
iptables -A INPUT -j LOGDROP

echo -n Reglas del Firewall Aplicadas........                    [OK]
echo ""

# Fin del script
```

La creación de los scripts para los demás host del prototipo es similar y depende de las políticas requeridas para cada cual.

La Figura 18. muestra la ejecución del script en el equipo y la verificación de las reglas aplicadas. Como puede verse, la política por defecto para las cadenas INPUT, OUTPUT y FORWARD es DROP y solamente se permiten los paquetes que se han habilitado en las reglas del script. Por lo tanto ni siquiera el comando *ping* desde el equipo es permitido como lo muestra la Figura 19. .

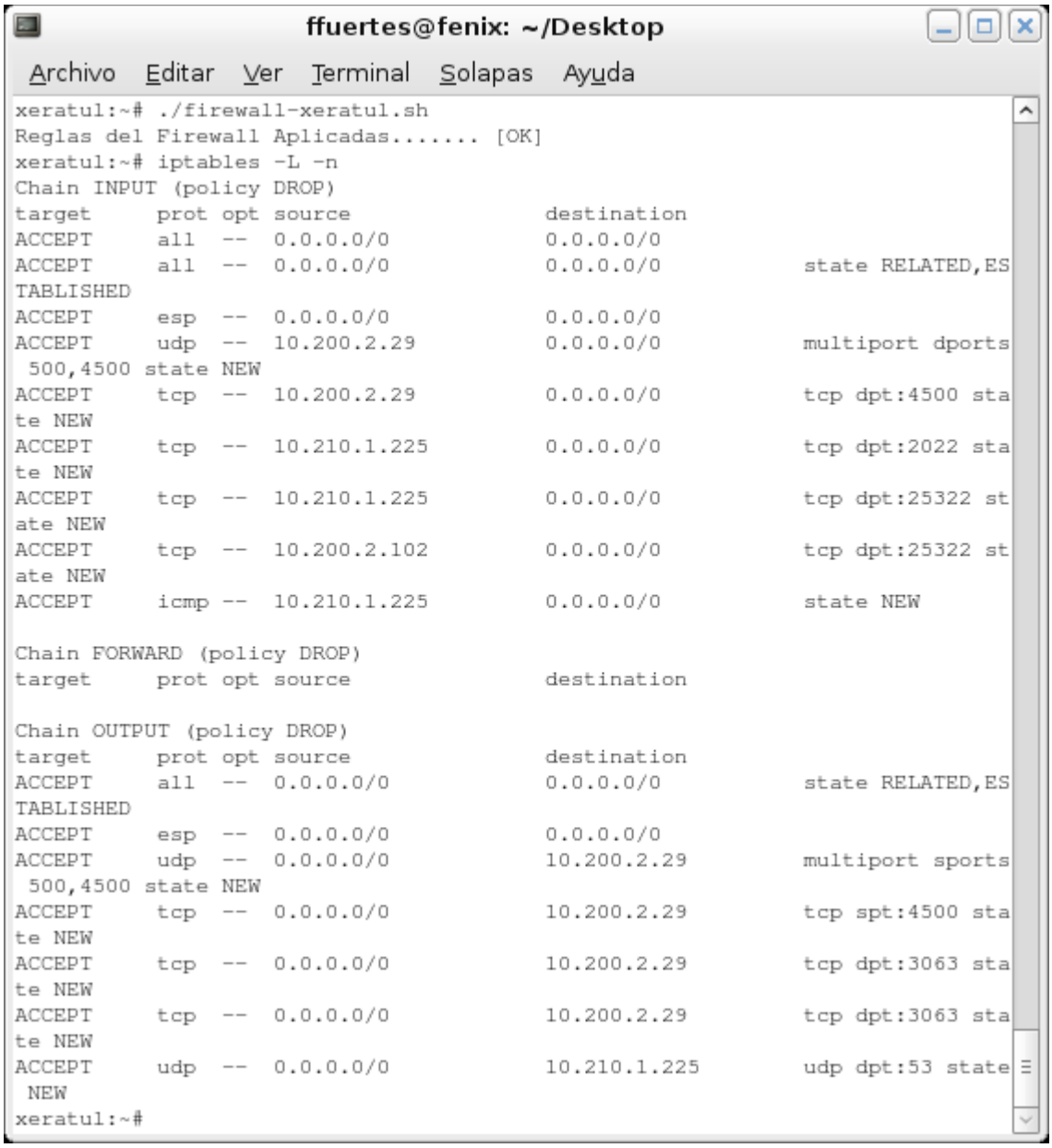

Figura 18. Ejecución del script de Iptables y verificación de reglas.

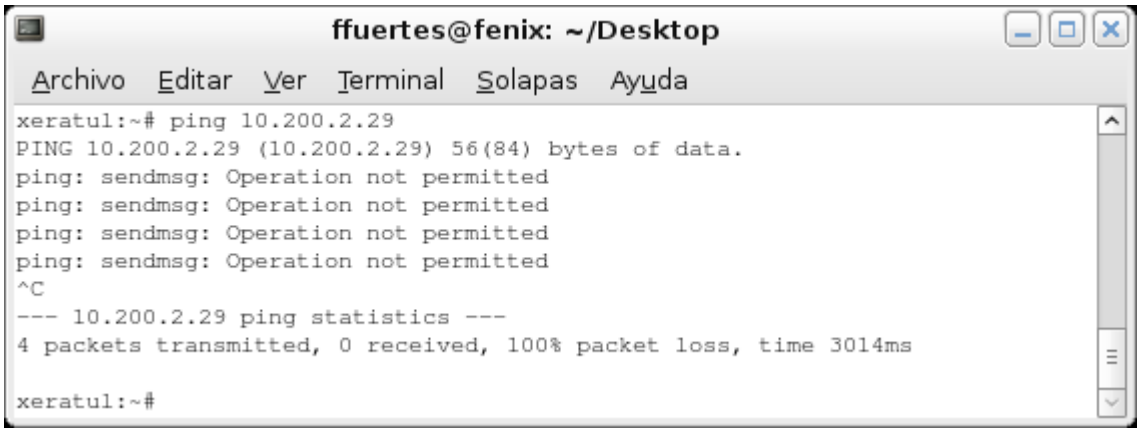

Figura 19. Restricción del comando *ping*

Una vez se prueba que las reglas del script son las correctas es necesario hacer que este se ejecute cada vez que inicie el sistema operativo del equipo. Figura 20. muestra como realizar este procedimiento utilizando el comando `update-rc.d` con el script de nombre *firewall-xeratul.sh.*

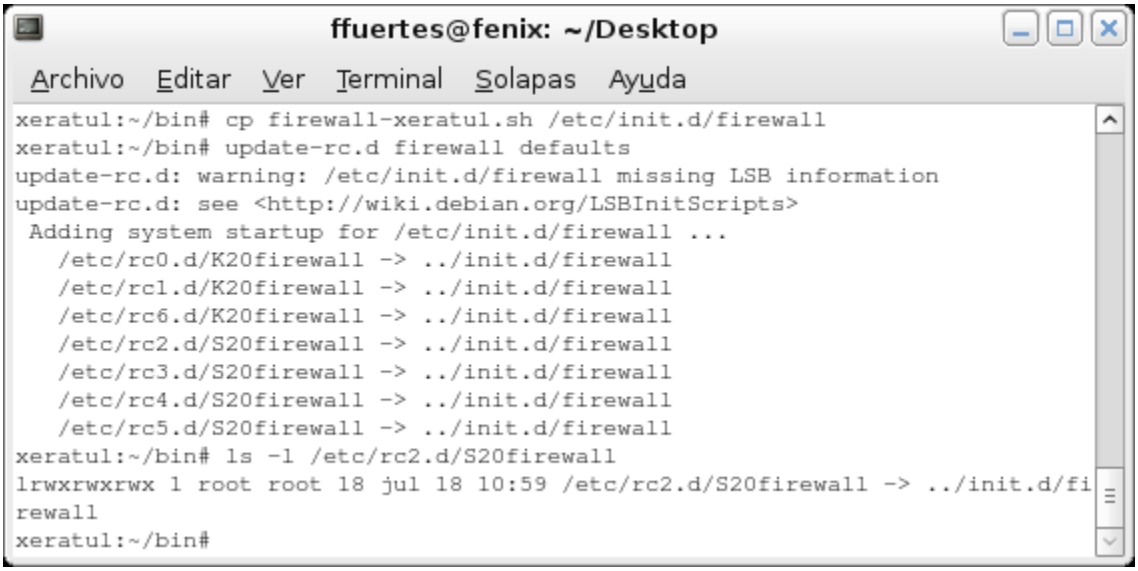

Figura 20. Proceso para iniciar firewall automáticamente en el arranque del sistema

4.3.3. Configuración de IPsec

4.3.3.1. Definición de Políticas de Seguridad IPsec

Antes de configurar un túnel IPsec es necesario determinar ciertas políticas que determinan el tratamiento de todo el tráfico IP entrante o saliente del host. Debido a que la pila IPsec opera en la capa de red (IP), el tráfico entrante se examina antes de pasar por el firewall Iptables y el saliente después de pasar por el mismo. De esta manera, las Políticas de Seguridad determinan si un paquete entrante o saliente puede pasar sin la

protección IPsec o de lo contrario se especifica los servicios de seguridad proporcionados, los protocolos a emplear, los algoritmos que se utilizan, etc.[50].

Como se mencionó en 3.5, la especificación TS 33.210 del 3GPP recomienda para las implementaciones IPsec utilizar modo túnel junto con el protocolo ESP. Estos se emplean en el prototipo, además se hace uso de autenticación mediante IKEv2 y certificados digitales X.509.

Igualmente se utiliza un cifrado fuerte con AES (*Advanced Encryption Standard*) en modo CBC (*Cipher-Block Chaining*) con claves de 256-bits y SHA2-512 (*Secure Hash Algorithm*) como función de hash. Todos estos elementos hacen que los servicios de seguridad proporcionados en cada túnel IPsec (autenticación, integridad, confidencialidad, no repudio) sean bastante confiables ya que se utilizan los algoritmos y protocolos recomendados por el 3GPP y por muchos expertos en seguridad que consideran muy difícil vulnerarlos con la tecnología y técnicas actuales.

Debido a que en el prototipo se implementa Iptables, el tráfico entre los nodos consiste únicamente en el permitido por las reglas del firewall aplicadas, esto simplifica el establecimiento de las Políticas de Seguridad, ya que se determina aplicar los servicios IPsec para todos los paquetes que se intercambien entre los hosts.

La Figura 21. muestra las conexiones IPsec entre los host necesarias para el buen funcionamiento del prototipo:

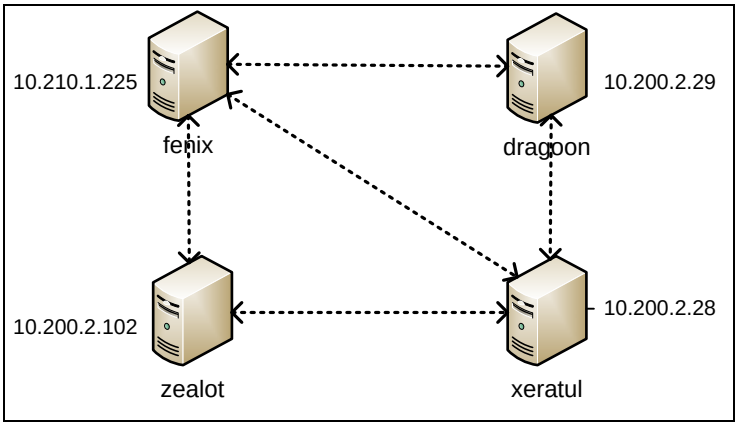

Figura 21. Conexiones IPsec entre hosts

Teniendo en cuenta las consideraciones anteriores y según la Figura 21. , las Políticas para cada equipo se reducen a las siguientes:

dragoon (AS) 10.200.2.29:

- Aplicar IPsec entre *dragoon* y *fenix* para todo el tráfico
- Aplicar IPsec entre *dragoon* y *xeratul* para todo el tráfico

xeratul (Middleware) 10.200.2.28:

- Aplicar IPsec entre *xeratul* y *fenix* para todo el tráfico

- Aplicar IPsec entre *xeratul* y *dragoon* para todo el tráfico
- Aplicar IPsec entre *xeratul* y *zealot* para todo el tráfico

zealot (SEG) 10.200.2.102:

- Aplicar IPsec entre *zealot* y *fenix* para todo el tráfico
- Aplicar IPsec entre *zealot* y *xeratul* para todo el tráfico

fenix (HSS, Admin, AGI, DNS) 10.210.1.225:

- Aplicar IPsec entre *fenix* y *xeratul* para todo el tráfico
- Aplicar IPsec entre *fenix* y *dragoon* para todo el tráfico
- Aplicar IPsec entre *fenix* y *zealot* para todo el tráfico

4.3.3.2. Instalación y descripción de componentes básicos

Para la implementación de los túneles IPsec se utiliza la herramienta strongSwan, esta se encuentra en los repositorios oficiales de Debian, lo cual facilita la instalación que se reduce al siguiente comando:

```
apt-get install strongswan
```

Este paquete utiliza a los demonios *charon y pluto* para el manejo de Asociaciones de Seguridad, los cuales a su vez puede ser administrado mediante el comando `ipsec`, que entre otras cosas se encarga de iniciar y detener el servicio IPsec en la máquina así como dar la orden para establecer o terminar una conexión cifrada.

Una vez instalado el paquete se procede a configurar en los hosts las Asociaciones de Seguridad necesarias para cada túnel. Los archivos más importantes en la configuración son:

/etc/strongswan.conf: Donde se configuran *plug-ins* y módulos adicionales que se cargan al iniciar el servicio.

/etc/ipsec.conf: Es el archivo más importante en la configuración ya que en este se configuran los parámetros para establecer las Asociaciones de Seguridad y Políticas de Seguridad IPsec.

/etc/ipsec.secrets: En este archivo se guardan las claves pre-compartidas y la ruta de los certificados X.509 con su respectiva contraseña (si la tiene) para el establecimiento de cada AS. Por lo tanto es muy importante que el dueño y grupo a los cuales pertenece este archivo sean *root,* además solamente debe tener permiso de lectura y ejecución para este usuario.

Directorios importantes:

/etc/ipsec.d/: Su función es almacenar archivos utilizados para la autenticación entre los pares que intentan establecer túneles IPsec como certificados, claves privadas, listas negras, etc.

/etc/ipsec.d/certs/: Almacena los certificados ya sea del propio host o de un equipo con el cual se establezca un túnel.

/etc/ipsec.d/cacerts/: Almacena los certificados de CA acreditadas. Si un certificado de host está firmado por una CA que tenga su certificado en este directorio, entonces el certificado de dicho host se considera válido.

/etc/ipsec.d/crls/: En este directorio se encuentran las Listas de Revocación de Certificados, en estas se almacenan los números seriales de certificados que por una u otra razón se han considerado inválidos por alguna CA.

/etc/ipsec.d/private/: Almacena la llave privada del host. Al igual que el archivo ipsec.secrets este directorio y su contenido deben tener los permisos adecuados para que únicamente sea el usuario *root* quien pueda acceder a ellos.

La Figura 22. muestra la manera de establecer permisos restrictivos para el archivo ipsec.secrets. De la misma manera se debe proceder con el directorio /etc/ipsec.d/private/ y las claves privadas que este contenga.

Figura 22. Establecer permisos adecuados al archivo ipsec.secrets

4.3.3.3. Localización de certificados

Antes de iniciar con la configuración, es necesario que cada equipo cuente con su certificado de host firmado por la CA correspondiente y su llave privada, así como con el certificado auto-firmado de la CA, ya que estos elementos son esenciales para el proceso de autenticación entre pares. En el anexo B se describe el procedimiento para la creación de una CA usando OpenSSL y la petición y firma de certificados para los host del prototipo.

Una vez la CA tenga los certificados, estos se copian al equipo con privilegios de ingreso a los demás nodos de la red *(fenix)* junto con el certificado de la CA y desde ahí se envían a los directorios apropiados de cada nodo mediante SCP como lo muestra la Figura 23. :

Figura 23. Copia de certificados desde *fenix* hacia *zealot*

En la gráfica anterior se observa que se copian de manera segura los archivos desde *fenix* hacia *zealot.* Procediendo de la misma manera para los demás hosts se tiene listo el ambiente para establecer los túneles IPsec con autenticación mediante certificados.

4.3.3.4. Configuración de strongSwan

La configuración de un túnel tanto en strongSwan como en otras implementaciones de IPsec debe ser simétrica, es decir se deben habilitar opciones inversas en los mismos archivos de configuración en cada host. Para entender esto se tomará como ejemplo de configuración del túnel entre los equipos *dragoon* y *xeratul.* Las configuraciones para los demás túneles y hosts se realizan de manera similar.

Archivo ipsec.secrets:

En este archivo se guarda el nombre de la clave privada del host, (esto si la clave se encuentra en el directorio /etc/ipsec.d/private) o la ruta completa a la misma, al igual que la contraseña con la cual se creó el certificado para el equipo. La única línea del archivo se ve así en *dragoon*:

```
: RSA dragoonKey.pem contraseña-segura
```

En los demás equipos se configura de la misma manera.

Archivo ipsec.conf:

Es el archivo más importante ya que aquí se configuran todos los parámetros de las asociaciones de seguridad. Se divide en dos secciones, la primera (config setup), donde se establecen parámetros generales del servicio como el inicio de los demonios, el intervalo de reenvío de certificados, el inicio en modo depuración, etc.

Para el prototipo únicamente se estableció el inicio automático de los dos demonios de strongSwan (*pluto* y *charon*) en todos los equipos:

```
config setup
      charonstart=yes
      plutostart=yes
```

La segunda sección define los parámetros para cada Asociación de Seguridad que se vaya a configurar, cada conexión se define con la palabra `conn`, generalmente se establece una por defecto de la cual se tomarán atributos para las demás conexiones. Esta se ve así en *dragoon*:

```
conn %default
      ikelifetime=60
      keylife=20
      rekeymargin=3m
      keyingtries=1
      left=10.200.2.29
      keyexchange=ikev2
```

En los anteriores parámetros se definen los tiempos de vida de una Asociación de Seguridad, además el parámetro `left` se refiere al propio host, en este caso la dirección IP. También se establece que el protocolo de intercambio de claves será IKEv2.

Las siguientes entradas `conn` se refieren a conexiones con otros equipos. El siguiente ejemplo muestra la Asociación de Seguridad que se realizará con el host *xeratul* definida

```
conn xeratul
      leftcert=dragoonCert.pem
      leftid="C=CO, ST=Cauca, L=Popayan, O=Unicauca,    OU=IMSeg,
         CN=dragoon.unicauca.edu.co,E=admin@imseg.unicauca.edu.co"
      right=10.200.2.28
      rightid="C=CO, ST=Cauca, L=Popayan, O=Unicauca,    OU=IMSeg,
         CN=xeratul.unicauca.edu.co,E=admin@imseg.unicauca.edu.co"
      ike=aes256-sha2_512-modp8192
      esp=aes256-sha2_512
      auto=add
```

en el archivo ipsec.conf de *dragoon:*

Como puede verse, el nombre de esta conexión es *xeratul,* se definen los atributos de autenticación del host local (*leftcert* y *leftid*) y del externo (*right* y *rightid*) presentes en los respectivos certificados. También se determinan los algoritmos de cifrado y hash (aes256 sha2_512) tanto para IKE como para ESP y por último la opción auto le indica a strongSwan que prepare la conexión cuando se arranque el servicio IPsec pero no la ponga en funcionamiento todavía, ya que esta será iniciada posteriormente de manera manual.

De la misma manera, en *xeratul* se debe tener una conexión simétrica a la anterior en el archivo ipsec.conf así:

```
conn dragoon
      leftcert=xeratulCert.pem
      leftid=”C=CO, ST=Cauca, L=Popayan, O=Unicauca,    OU=IMSeg,
         CN=xeratul.unicauca.edu.co,E=admin@imseg.unicauca.edu.co”
      right=10.200.2.29
      rightid=”C=CO, ST=Cauca, L=Popayan, O=Unicauca,    OU=IMSeg,
         CN=dragoon.unicauca.edu.co,E=admin@imseg.unicauca.edu.co”
      ike=aes256-sha2_512-modp8192
      esp=aes256-sha2_512
      auto=add
```

Estas configuraciones garantizan en primer lugar la autenticación de las partes, ya que en las dos fases de IKE se intercambian y validan certificados en ambos lados de la comunicación, los cuales deben corresponder a la configuración de cada AS y ser emitidos por una CA de confianza. En segundo lugar y si el proceso previo es exitoso, se establecen los parámetros de cifrado y se intercambian las claves simétricas para poder iniciar la comunicación segura.

De esta forma el archivo de configuración completo para *xeratul* según las Políticas de Seguridad definidas anteriormente queda de la siguiente manera:

```
# ipsec.conf - strongSwan IPsec configuration file
# basic configuration

config setup
      charonstart=yes
      plutostart=yes

# AS connections
conn %default
      ikelifetime=60
      keylife=20
      rekeymargin=3m
      keyingtries=1
      left=10.200.2.28
      keyexchange=ikev2

conn dragoon
      leftcert=xeratulCert.pem
      leftid=”C=CO, ST=Cauca, L=Popayan, O=Unicauca,    OU=IMSeg,
         CN=xeratul.unicauca.edu.co,E=admin@imseg.unicauca.edu.co”
      right=10.200.2.29
```

```
      rightid="C=CO, ST=Cauca, L=Popayan, O=Unicauca,    OU=IMSeg,
         CN=dragoon.unicauca.edu.co,E=admin@imseg.unicauca.edu.co"
      ike=aes256-sha2_512-modp8192
      esp=aes256-sha2_512
      auto=add

conn fenix
      leftcert=xeratulCert.pem
      leftid="C=CO, ST=Cauca, L=Popayan, O=Unicauca,    OU=IMSeg,
         CN=xeratul.unicauca.edu.co,E=admin@imseg.unicauca.edu.co"
      right=10.210.1.225
      rightid="C=CO, ST=Cauca, L=Popayan, O=Unicauca,    OU=IMSeg,
         CN=fenix.unicauca.edu.co,E=admin@imseg.unicauca.edu.co"
      ike=aes256-sha2_512-modp8192
      esp=aes256-sha2_512
      auto=add

conn dragoon
      leftcert=xeratulCert.pem
      leftid="C=CO, ST=Cauca, L=Popayan, O=Unicauca,    OU=IMSeg,
         CN=xeratul.unicauca.edu.co,E=admin@imseg.unicauca.edu.co"
      right=10.200.2.102
      rightid="C=CO, ST=Cauca, L=Popayan, O=Unicauca,    OU=IMSeg,
         CN=zealot.unicauca.edu.co,E=admin@imseg.unicauca.edu.co"
      ike=aes256-sha2_512-modp8192
      esp=aes256-sha2_512
      auto=add
```

El archivo strongswan.conf se deja tal cual como lo instala el paquete ya que no se necesita cargar módulos adicionales en las configuraciones.

Una vez realizados estos cambios en los archivos de configuración de ambos hosts, se tiene listo el túnel, el cual se puede iniciar desde uno u otro lado de la conexión mediante el siguiente comando:

```
ipsec up nombre-de-la-conexión
```

La Figura 24. muestra el inicio del túnel IPsec entre *dragoon* y *xeratul,* en esta gráfica se muestran todos los mensajes provenientes de la autenticación mutua entre los pares; se detalla el proceso de intercambio de certificados y su validación de parte y parte hasta establecer la comunicación segura satisfactoriamente.

La Figura 25. muestra la verificación las conexiones IPsec establecidas utilizando el comando `ipsec status`. Como puede verse, existen dos Asociaciones de Seguridad en el host dragoon: una con xeratul y otra con *fenix*, las dos en modo túnel y utilizando el protocolo ESP.

Una vez las configuraciones de strongSwan están listas en cada host, la ultima parte de la

implementación consiste en iniciar todas las AS de manera ordenada, por tanto es necesario determinar el proceso de arranque del servicio IPsec, ya que si todos los nodos comienzan las conexiones seguras configuradas por defecto, se estaría intentando iniciar dos veces cada conexión, lo cual aparte de ser innecesario, podría causar inconsistencias si se intenta establecer una comunicación al mismo tiempo desde dos equipos distintos. Por lo tanto una vez todos los host estén encendidos debe existir una secuencia de establecimiento de conexiones, que asegura que cada AS configurada se inicie desde un solo equipo.

Debido a que cualquiera de las partes de un túnel IPsec puede iniciar o terminar una AS, es posible elegir una secuencia arbitraria para el establecimiento de las conexiones. En el prototipo se decidió que el orden fuera el siguiente:

fenix: - Inicia AS con *xeratul*

xeratul: - Inicia AS con *dragoon*

dragoon: - Inicia AS con *fenix*

zealot: - Inicia AS con *xeratul*
- Inicia AS con *fenix*

Figura 24. Inicio de una conexión IPsec.

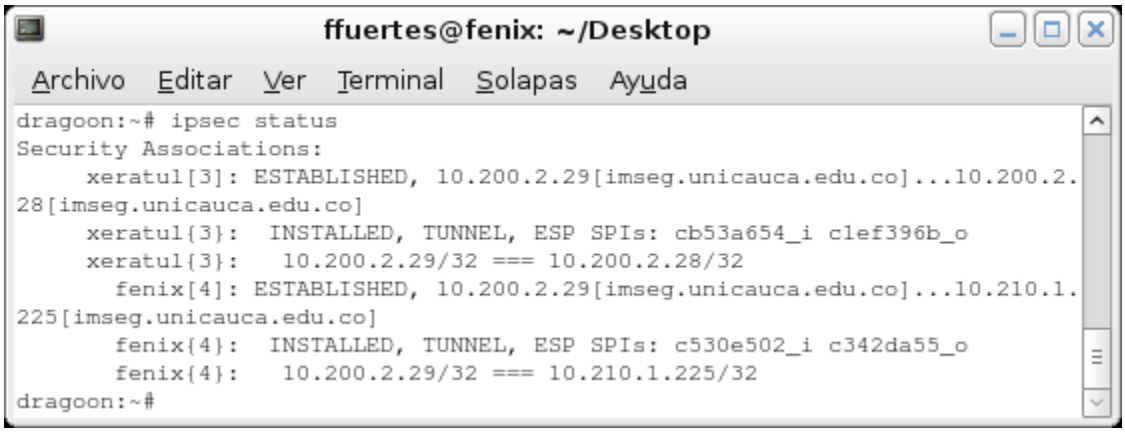

Figura 25. Verificación de las conexiones IPsec establecidas

Por comodidad en cada host se crea un script en *bash* con el comando para dar inicio a la comunicación segura que le corresponde. A manera de ejemplo, el script para establecer los túneles definidos para *zealot* es el siguiente:

```
#!/bin/bash
#Script de inicio de conexiones Ipsec

#Mirar si existen conexiones
echo Las conexiones actuales son:
ipsec status

#Iniciar AS xeratul
ipsec up xeratul > /dev/null
sleep 1

#Iniciar AS fenix
ipsec up fenix > /dev/null
sleep 1

#Verificar que se ha iniciado los túneles
echo Ahora se han establecido las siguientes conexiones:
ipsec status
```

La Figura 26. muestra la ejecución satisfactoria de este script:

Figura 26. Ejecución del script de inicio de conexiones IPsec

CAPÍTULO 5: EVALUACIÓN DE SEGURIDAD Y RENDIMIENTO DEL MECANISMO PROPUESTO

5.1. Introducción

En anteriores capítulos se llegó al diseño e implementación del mecanismo de seguridad para el intercambio de datos en IMS. En el presente capítulo se toma el prototipo implementado para realizar una validación del mismo en cuanto al nivel de seguridad que proporciona y al rendimiento de la red cuando se incorpora el mecanismo propuesto en un modelo de sincronización de datos de usuario en IMS sin elementos de seguridad.

En la primera parte, se utiliza una conocida metodología de testeo de seguridad en el campo de las TI (Tecnologías de Información), para poner a prueba la seguridad del mecanismo mediante un test de intrusión exhaustivo.

Posteriormente, se realiza la evaluación de rendimiento utilizando métodos similares a los llevados a cabo en SincIMS, con el fin de tener elementos de comparación entre los dos trabajos.

5.2. Evaluación de seguridad

Para llevar a cabo este proceso se siguió el manual de la metodología OSSTMM 3 [7] (*Open-Source Security Testing Methodology Manual*, versión 3), con el propósito de evaluar el desempeño del prototipo contra las amenazas de seguridad más comunes.

Dado el enfoque de OSSTMM para el análisis de seguridad en el amplio campo de TI, es necesario tomar de este, únicamente los procedimientos aplicables al entorno de trabajo, en este caso la evaluación del prototipo de seguridad, por tanto se utilizó el *Canal COMSEC* que comprende la seguridad en redes de datos y telecomunicaciones.

A su vez, el plan de pruebas utilizado, fue adaptado de los criterios para la realización de un plan de pruebas de seguridad presentados en el trabajo de grado: *"Criterios Para Establecer Políticas De Seguridad De La Información Y Plan De contingencia, Caso de Estudio El Centro de Datos de La Universidad del Cauca"* [51], desarrollado en la FIET por Carolina Guevara Campo y Fabián Andrés Mera, dirigido por el ingeniero Siler Amador, en el cual se sigue la metodología OSSTMM para llevar a cabo un test de intrusión dentro del plan de pruebas. El flujo de las actividades del plan adaptado para el presente trabajo se muestra en la Figura 27.

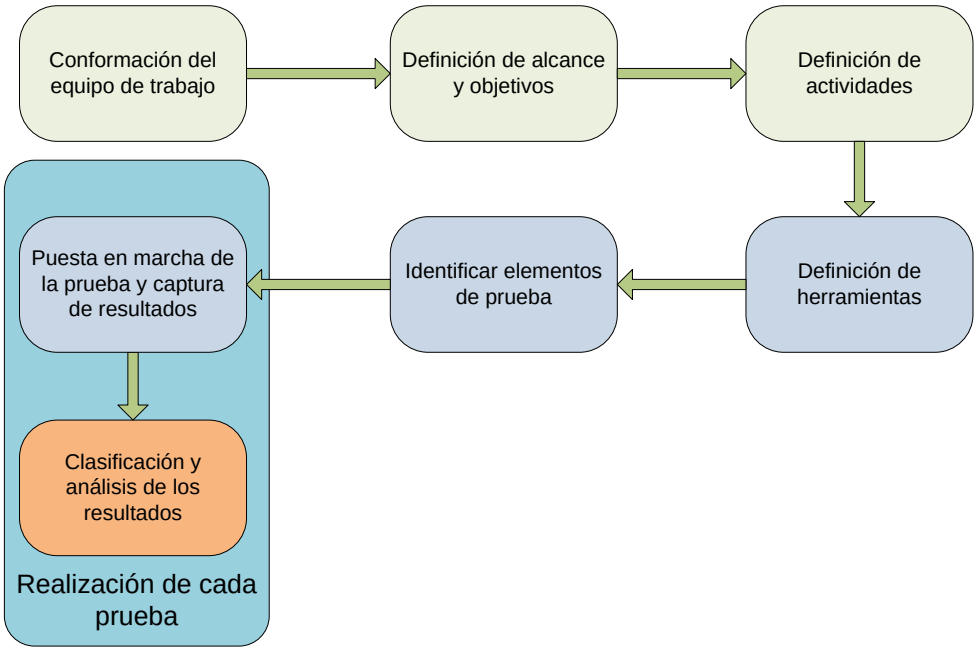

Figura 27. Flujo de Actividades en el Plan de Pruebas de Seguridad

Las actividades mostradas en la Figura 27, referentes al plan para la evaluación de IMSeg son las siguientes:

1. Conformación del equipo de trabajo para realizar, monitorear y revisar los resultados de las pruebas.
2. Definición del límite y alcance de las pruebas, donde se delimita los objetivos propuestos.
3. Definición general de las actividades a realizar en el proceso de evaluación.
4. Definición de las herramientas necesarias para llevar a cabo las pruebas.
5. Identificación de los equipos y elementos a los cuales se realizaran las pruebas.
6. Realización de las pruebas y consignación de los respectivos resultados.
7. Después de cada prueba se clasificación y analizan los resultados obtenidos.

5.2.1. Conformación del Equipo de trabajo

- 2 Desarrolladores: Estudiantes de trabajo de grado. Encargados de plantear, definir, llevar a cabo y analizar las pruebas.
- 1 Asesor: Directora del trabajo de grado. Encargada de revisar el plan de pruebas y verificar el cumplimiento de las mismas.

5.2.2. Definición del Alcance y Objetivos

El objetivo de esta evaluación es comprobar que la implementación del prototipo propuesto para IMSeg cumpla con los requerimientos de seguridad necesarios en una red

en la cual se intercambian datos de usuario entre servidores, en este caso en el ámbito de IMS.

Dado que se evalúa el desempeño de un prototipo, las pruebas se centrarán exclusivamente en la funcionalidad del mismo y no se tendrán en cuenta aspectos externos que se sugieren en la metodología OSSTMM, como la revisión de políticas de la empresa, auditoria al personal operario, revisión de hardware, calidad de la red, etc.

Como se está tratando con un sistema que no está en producción, para llevar a cabo las pruebas se simularán situaciones anormales, en algunas de ellas incluso se le darán privilegios al auditor sobre elementos que en condiciones normales serian confidenciales, esto con el fin de tener una comprobación más exhaustiva de la seguridad del prototipo.

En OSSTMM se definen varios tipos de test, en este caso se llevará a cabo un test tipo *Tándem* o *Caja de Cristal*, en el cual el auditor al ser parte del equipo que implementó el prototipo de seguridad, posee conocimiento previo integral de los elementos o del entorno a ser testeados.

5.2.3. Definición de las actividades a realizar

Las siguientes actividades se tomaron del manual OSSTMM y se adaptaron al presente proyecto, estas representan el tipo de pruebas a realizar. Cabe mencionar que dentro de cada actividad pueden existir una o más pruebas distintas.

1. Actualización: Revisión de las versiones de software de las aplicaciones requeridas para las operaciones normales en la red, instaladas en cada equipo.
2. Visibilidad: Revisión de la visibilidad de los equipos objetivo desde el exterior a la red prototipo.
3. Acceso: Revisión de los métodos de autenticación entre los equipos buscando posibles fallos de seguridad.
4. Confianza: Verificar si es posible acceder a los recursos del sistema sin autenticación previa o falseando la identidad de los equipos.
5. No repudio: Verificar que las aplicaciones involucradas en el prototipo guarden registros de sus interacciones con otras entidades.
6. Confidencialidad e Integridad: Verificar el correcto funcionamiento del método para proteger la confidencialidad e integridad de la información intercambiada entre dos partes del sistema.
7. Alertas y logs: Verificar el uso de sistemas de logs o alarmas que informen sobre situaciones sospechosas o actividades perjudiciales sobre la red.
8. Validación de supervivencia: determinar y medir la resistencia de los equipos de la red hacia situaciones que causen que el sistema deje de funcionar en su totalidad o en parte debido a ataques tipo DoS.

5.2.4. Definición de Herramientas

Las herramientas utilizadas para llevar a cabo las pruebas mencionadas en 5.2.3 son:

- Wireshark [52]: es un analizador de protocolos de red, el cual permite configurar la tarjeta de red de un computador en modo "promiscuo" para capturar todos los paquetes que circulen en el mismo segmento de red para identificar y analizar este tráfico de manera detallada.

- John the Ripper [53]: Herramienta criptográfica utilizada para descifrar contraseñas principalmente sobre sistemas Unix. Es muy popular tanto en el ambiente hacker como en el de administradores de sistemas para detectar la debilidad de las contraseñas de usuario.

- Tcpdump [54]: Herramienta de línea de comandos para sistemas operativos tipo Unix, la cual permite capturar, mostrar y analizar el tráfico que circula por la red en tiempo real y aplicando filtros de paquetes de forma similar a Wireshark.

- Nmap [55]: Herramienta para exploración de redes y auditoria de seguridad. Entre sus utilidades se encuentran descubrir equipos activos en una red, igualmente determinar servicios, puertos abiertos, sistema operativo, versiones de software, entre otras, sobre una máquina remota.

- Paxtest [56]: Herramienta que evalúa si una maquina Linux es vulnerable a algunos de los ataques más comunes sobre el kernel instalado. Por ejemplo, la ejecución de código malicioso para escalar de privilegios.

- Ettercap [57]: Es un *sniffer* para redes LAN que se interconectan mediante *HUBs* o *Switches*, el cual se especializa en ataques tipo Hombre en el Medio. Puede ejecutarse en modo Interactivo o No-interactivo, soporta diversos protocolos y es capaz de inyectar datos e una conexión establecida y sincronizada.

5.2.5. Identificar elementos de prueba

Las pruebas se realizarán sobre la red prototipo implementada según lo expuesto en el capítulo 4. Dependiendo del tipo de prueba se examina a cada equipo localmente (como usuario root) o exteriormente desde el equipo del auditor que se encuentra en el mismo segmento de red. La Figura 28 muestra la red para los laboratorios de pruebas.

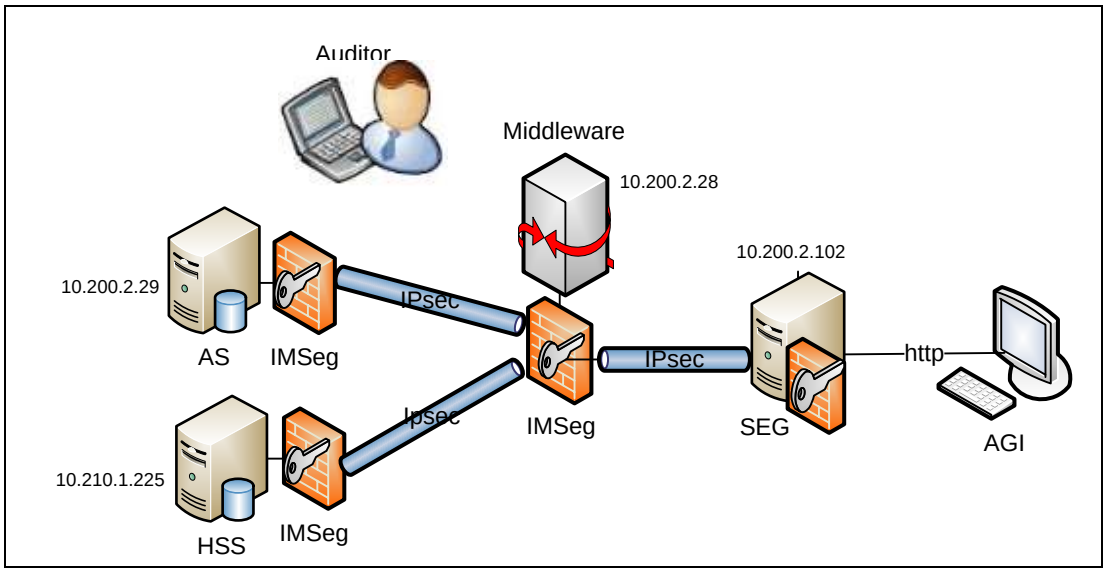

Figura 28. Red para los laboratorios de pruebas de seguridad

La Tabla 2 muestra la descripción de las principales características de los cuatro equipos que van a ser evaluados, así como las del equipo auditor.

Tabla 2. Descripción de los equipos involucrados en las pruebas

Nombre del equipo	Dirección IP	Utilidad en la Red	Herramientas Instaladas
fenix	10.210.1.225	HSS, AGI, Estación de trabajo del administrador de la red	Open IMS Core, AGI, Iptables, strongSwan, OpenSSH
xeratul	10.200.2.28	Middleware	Sequoia, Iptables, strongSwan, OpenSSH
zealot	10.200.2.102	SEG	Bea Weblogic, Iptables, strongSwan, Gateway, OpenSSH
dragoon	10.200.2.29	AS	MySQL Server, Bea Weblogic, Iptables, strongSwan, OpenSSH
koopa	10.210.1.167 ó 10.200.2.167	Equipo Auditor	Nmap, hping3, John the Ripper, Wireshark

5.2.6. Realización de las pruebas y captura de resultados

La descripción detallada de cada una de las pruebas de seguridad realizadas se puede encontrar en el Anexo C. En esta sección se presentan los resultados de las mismas en forma de tablas.

5.2.6.1. Actualización

En la Tabla 3 se resumen los resultados de las pruebas de actualización realizadas a los cuatro hosts. El guion (-) significa que en ese equipo no se encuentra instalado el software de la columna azul de la izquierda, la palabra "Sí" significa que cumple con el requerimiento.

Tabla 3. Resultados de las pruebas de Actualización en los cuatro hosts

Software Actualizado	fenix	zealot	xeratul	dragoon
Debian 5.0.2. Lenny	Sí	Sí	Sí	Sí
Open IMS Core (Octubre 2008)	Sí	-	-	-
Java 1.6.12	Sí	Sí	Sí	-
MySQL 5.0.5	Sí	-	-	Sí
Sequoia 2.10.10	-	-	Sí	-
Bea Weblogic SIP Server 3.1	-	Sí	-	-
strongSwan 4.2.4	Sí	Sí	Sí	Sí
Iptables 1.4.2.6	Sí	Sí	Sí	Sí
Kernel de Linux 2.6.26-2	2.6.27.10grsec	2.6.27.10grsec	2.6.27.10grsec	2.6.27.10grsec
OpenSSH-server 5.1	Sí	Sí	Sí	Sí

5.2.6.2. Visibilidad

En estas pruebas se utilizaron las herramientas Ping y Nmap para determinar si los equipos de la red prototipo son o no alcanzables o visibles desde el exterior. Se realizaron pruebas con y sin el firewall local en cada equipo. Incluso con opciones avanzadas de Nmap, se determinó que el firewall local de IMSeg hace que no sea posible alcanzar a los host si no se encuentran reglas permitiendo este acceso. Los resultados de las pruebas se muestran en la Tabla 4.

Tabla 4. Resultados de las pruebas de visibilidad en los cuatro hosts

Prueba	fenix	zealot	dragoon	xeratul
Ping sin firewall	visible	visible	visible	visible
Nmap sin firewall	visible	visible	visible	visible
Ping con firewall	No visible	No visible	No visible	No visible
Nmap con firewall	No visible	No visible	No visible	No visible

5.2.6.3. Acceso

En estas pruebas se buscan posibles configuraciones inseguras de los métodos de acceso a los equipos: usuarios con Shell privilegiada, configuración de SSH, fortaleza de las contraseñas de usuario y vulnerabilidad del Kernel. En la tabla 5 se muestran los resultados de las pruebas.

Tabla 5. Resultados de las pruebas de acceso en los cuatro hosts

Prueba	fenix	zealot	dragoon	xeratul
Usuarios con Shell privilegiada	root, ffuertes	root, ffuertes	root, tesis	root, tesis
Acceso de root por SSH habilitado	No	No	No	No
Puerto SSH por defecto habilitado	No	No	No	No
Equipos con Acceso SSH a este host	Direcciones IP conocidas	fenix	fenix	fenix
John the Ripper descifró contraseñas de usuario	No	No	No	No
Vulnerabilidades encontradas en el kernel antes de endurecimiento	19	19	19	19
Vulnerabilidades encontradas en el kernel después del endurecimiento	3	3	3	3

5.2.6.4. Confianza

Para esta actividad se realizó una prueba de Envenenamiento ARP o ataque de Hombre en el Medio, en la cual el equipo auditor envía repetidos mensajes ARP a dos de los host del prototipo con el fin de que todos los mensajes que intercambien estos dos equipos pasen a través de él y este pueda visualizarlos claramente. Se hizo la prueba con y sin el mecanismo de seguridad habilitado, los resultados se muestran en la Tabla 6.

Tabla 6. Resultados de las pruebas de verificación de confianza

Prueba	Sin IMSeg	Con IMSeg
Envenenamiento ARP	Hombre en el Medio exitoso	Hombre en el Medio No exitoso

5.2.6.5. Verificar mecanismos de No Repudio

En esta actividad, se realizaron varias pruebas para determinar si los logs las aplicaciones de seguridad de IMSeg: strongSwan, Iptables y TCP Wrappers, registran satisfactoriamente la identificación de los equipos que han interactuado de alguna manera

con ellas, guardando estos datos con fecha y hora como mecanismo de No Repudio. Los resultados de estas pruebas fueron satisfactorios y se muestran en la Tabla 7.

Tabla 7. Resultados de las pruebas de verificación de mecanismos de No Repudio

Logs de aplicaciones	Inicio de sesión	Terminación de sesión	Identificación mediante IP	Fecha y hora de la acción
Logs de strongSwan	Registra	Registra	Registra	Registra
Logs de Iptables	Registra	Registra	Registra	Registra
Logs de TCP Wrappers	Registra	No Registra	Registra	Registra

5.2.6.6. Confidencialidad e Integridad:

En esta actividad, se realizaron pruebas para determinar si la confidencialidad e integridad de la información intercambiada entre dos hosts del prototipo se ve comprometida utilizando técnicas de espionaje de red.

En primer lugar se deshabilitó IMSeg, con esto se logró capturar y visualizar claramente los datos intercambiados entre dos equipos del prototipo correspondientes a una consulta SQL sobre una base de datos. Posteriormente se realizó el mismo experimento habilitando únicamente la herramienta strongSwan, puesto que el firewall impide realizar ataques de Hombre en el Medio (ver 5.2.6.4 Verificación de confianza) y se pretendía probar si los túneles IPsec proporcionaban los servicios de Confidencialidad e Integridad por si solos; el resultado fue la captura de paquetes cifrados por ESP, protocolo utilizado por IPsec, lo cual confirmó el buen funcionamiento de este modulo de IMSeg. Los resultados de las pruebas se resumen en la Tabla 8.

Tabla 8. Resultados de las pruebas de confidencialidad e integridad

Prueba	Sin IMSeg	Con IMSeg
Captura de datos intercambiados en texto plano	Sí	No
Posibilidad de modificación de los datos capturados	Sí	No

5.2.6.7. Alertas y logs

Con estas pruebas se busca verificar que los sistemas logs de las diferentes aplicaciones del prototipo registren posibles actividades maliciosas y las etiqueten como tal. En general estas actividades serán intentos de conexión o acceso a las aplicaciones desde ubicaciones no autorizadas, las cuales son detectadas satisfactoriamente como lo

muestra la Tabla 9. Del mismo modo, como se ve en la tabla, se verificó que el único usuario con privilegios sobre estos archivos tan importantes sea el usuario root.

Tabla 9. Resultados de las pruebas de alertas y logs

Logs de aplicaciones	Intento de ARP Spoof	Identificación mediante IP	Fecha y hora de la acción	Intento de conexión o acceso no autorizada	Usuarios con privilegios sobre el log
Logs de strongSwan	No Registra	Registra	Registra	Registra como AUTH_FAILED	root
Logs de Iptables	No Registra	Registra	Registra	Registra como IMSeg_RECHAZADO	root
Logs de TCP Wrappers	No Registra	Registra	Registra	Registra como Acceso no autorizado	root
Logs de ARP Alert	Registra como ip_change	Registra dirección MAC e IP	Registra	No Registra	root

5.2.6.8. Validación de supervivencia

El objetivo de esta actividad es probar la resistencia de los equipos de la red hacia situaciones que pretendan causar que el sistema en su totalidad o en parte, deje de funcionar con normalidad, debido a la recepción de parámetros excesivos. Esto se conoce comúnmente como DoS.

Sin embargo, dado de que los elementos del mecanismo de seguridad hacen que los hosts del prototipo ignoren los intentos de conexión de equipos que no sean autorizados, (ver 5.2.6.2 Visibilidad, 5.2.6.3 Acceso, 5.2.6.4 Confianza, 5.2.6.7 Alertas y logs) los diferentes ataques de denegación de servicio también son ignorados, sin embargo es posible probar la resistencia del prototipo cuando se carga la red con trafico autorizado pero excesivo, lo cual se lleva a cabo en la evaluación de rendimiento (siguiente sección), donde se hacen pruebas hasta con 800 peticiones de actualizaciones de usuario por segundo sin que el sistema deje de funcionar.

5.3. Evaluación de rendimiento

Las pruebas de rendimiento se realizan con el fin de verificar los efectos causados por la incorporación del mecanismo de seguridad IMSeg sobre el desempeño de la red. Para esto se plantean escenarios y pruebas similares a las llevadas a cabo en SecIMS **¡Error! No se encuentra el origen de la referencia.**[6], con el fin de tener un punto de referencia para la validación de los resultados. Así, se va a simular un servicio de la Arquitectura de Servicios Nativos IMS, el cual genera y requiere información de usuario del *Repository Data.*

El servicio simulado es el de localización de un usuario móvil, en el cual el equipo del usuario actualiza su ubicación periódicamente tanto al HSS como al AS. Por ejemplo, es de gran utilidad si la persona se encuentra suscrita a un servicio de localización de sitios

de interés en una ciudad que se encuentra visitando. El AS debe conocer las coordenadas del usuario para poder enviarle la ubicación de sitios cercanos, como restaurantes, tiendas, bancos, etc.

EL AS se encarga de actualizar la ubicación del usuario en el mapa de acuerdo a la información suministrada por una Aplicación Generadora de Información (AGI) descrita en la sección 3.6.2 que simula un servidor de localización. Estas coordenadas se guardan en la base de datos del HSS (específicamente en el *Repository Data*) y en la base de datos del AS y son actualizadas mediante mecanismos de sincronización implementados en la plataforma *Middleware*.

Para tener un punto de referencia y evaluar los resultados se plantearon dos escenarios; en el primero se usó la red base sin utilizar seguridad en la comunicación ni en los servidores y para el segundo se usó el mecanismo propuesto.

5.3.1. Escenario 1

La Figura 29 muestra la arquitectura de red para este escenario, en el cual se tienen los elementos de la red base sin ningún tipo de seguridad.

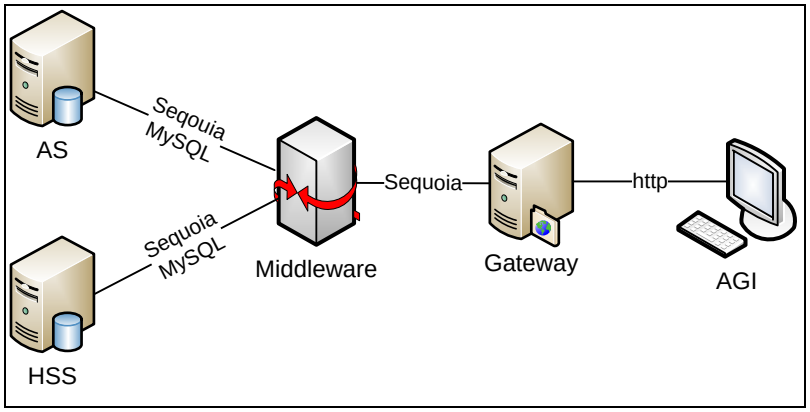

Figura 29. Escenario de pruebas 1

Los nodos de la red y sus funciones se describen en el Capítulo 3. El resumen de la operación de esta red es el siguiente: la AGI genera coordenadas que representan las ubicaciones de cierto número de usuarios IMS, estas deben llegar tanto al HSS como a la base de datos del AS, por lo tanto se envían como mensajes HTTP a través de una *Gateway* que se comunica con el *Middleware Sequoia*, el cual se encarga de actualizar paralelamente las dos bases de datos mencionadas.

5.3.2. Escenario 2

Este escenario consiste en la red base mostrada en el Escenario 1 a la cual se añaden los elementos de seguridad que componen IMSeg (Endurecimiento, firewall e IPsec), descritos en los capítulos 3 y 4. La Figura 30 muestra la arquitectura de esta red.

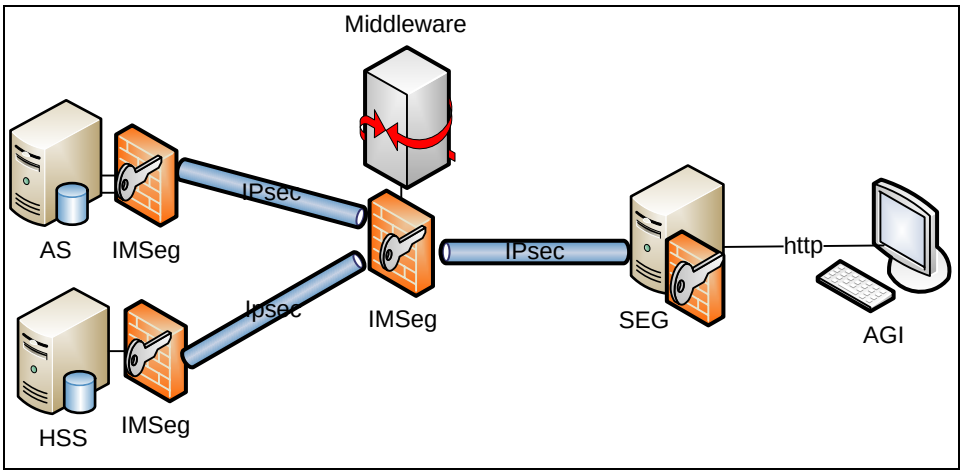

Figura 30. Escenario de pruebas 2

El funcionamiento básico de esta red es igual al del Escenario 1, pero se ha agregado seguridad en las comunicaciones y servidores por lo tanto se espera que la carga de tráfico para cada actualización sea mayor debido a los diferentes procesos de seguridad.

5.3.3. Características de las capturas realizadas

Con el fin de variar la carga de la red en cuanto a procesamiento y número de paquetes por segundo que se transportan por la misma, en cada escenario se realizaron varias capturas de los mensajes que se transfirieron entre los nodos que intervienen en el proceso de actualización. Para esto, en cada captura se mantuvo fijo el número de actualizaciones en 50, pero se variaron el número de usuarios para los cuales la AGI está actualizando la información de localización entre 10, 50 y 100, además el retardo entre actualizaciones se vario entre 500 ms, 250 ms y 125 ms. Combinando estas posibilidades se obtuvieron nueve capturas para cada escenario.

Así, la captura con menos carga para la red fue con 10 usuarios y con un retardo de 500 ms entre cada actualización. Por otra parte, la captura con 100 usuarios y un retardo de 125 ms, además de generar un tráfico bastante alto (800 peticiones de actualización de usuario por segundo) conlleva a que los equipos utilizados eleven su nivel de procesamiento, llegando al 98% de uso del procesador, incluso para el escenario 1 (sin hacer uso de seguridad), por lo tanto se considera la captura más exigente tanto para la red como para los equipos.

Para cada uno de los casos que se presentan, se utilizó el analizador de protocolos Wireshark para tomar estadísticas del tiempo que se tardó cada mecanismo en completar todas las actualizaciones y el número de Bytes que se generaron y se transportaron por la red.

Los parámetros extraídos de cada captura que sirven para generar una comparación del comportamiento entre los dos escenarios son los siguientes:

- Tiempo entre el primer y el último paquete (segs): es el tiempo total que se tarda el mecanismo en procesar todas las peticiones y enviar los mensajes a los nodos que correspondan.

- Paquetes: es el número total de paquetes generados por todos los nodos que intervienen en el mecanismo.

- Promedio Paquetes/seg: es el promedio de paquetes generados que se transportan por la red en un segundo.

- Promedio tamaño de paquetes (Bytes): es el promedio del tamaño de todos los paquetes generados.

- Bytes: es el número de Bytes total generados por todos los nodos que intervienen en el mecanismo.

- Promedio Bytes/seg: es el promedio de Bytes generados que se transportan por la red en un segundo.

- Promedio Mbit/seg: es el promedio de Mega bits de datos que se transportan por la red en un segundo, lo cual indica la carga promedio de la red en un segundo.

5.3.4. Resultados obtenidos

En las nueve capturas se extrajeron los parámetros mencionados anteriormente, los cuales se presentan en tablas comparativas entre los dos escenarios. De la misma manera se tomaron gráficas de tráfico (tiempo contra número de Bytes) que ayudan a visualizar la carga existente tanto en el enlace que inicia una actualización (entre los nodos AGI y Gateway), como en uno de los enlaces donde esta termina (entre los nodos Middleware y HSS). Esto con el fin de verificar si existe retardo entre la salida de los paquetes desde el nodo AGI y la llegada de los mismos hasta el HSS que es una de las bases de datos de destino.

- Captura 1

    - Número de actualizaciones: 50
    - Número de usuarios por actualización: 10
    - Retardo entre actualizaciones: 500 ms

Esta es la captura que representa menos carga para la red (20 usuarios por segundo), las Figuras 31 y 32 muestran la diferencia de tráfico entre los dos escenarios; en rojo se aprecia la carga generada por cada actualización que sale del AGI y llega hasta el Gateway y en azul la carga presente en los mensajes MySQL que envía el Middleware hacia el HSS. Al comparar las dos imágenes se observa que el trafico azul es mayor cuando se implementa IMSeg (Escenario 2), pero aun así, no se presentan retrasos entre la salida de la actualización desde el AGI hasta la llegada de la misma al HSS.

Los valores de los parámetros consignados en la Tabla 10 confirman el aumento de la

carga de tráfico en el escenario 2, lo cual era de esperarse ya que los túneles IPsec necesitan introducir varios bytes adicionales en cada paquete para garantizar confidencialidad, integridad y asegurar la autenticación mediante procesos como cifrado, descifrado, hash, establecimiento de AS, etc.

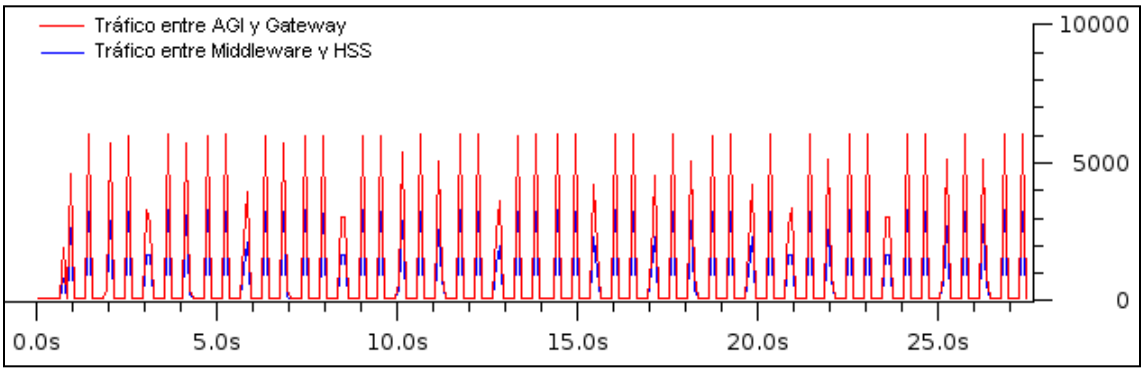

Figura 31. Captura 1 - Escenario 1

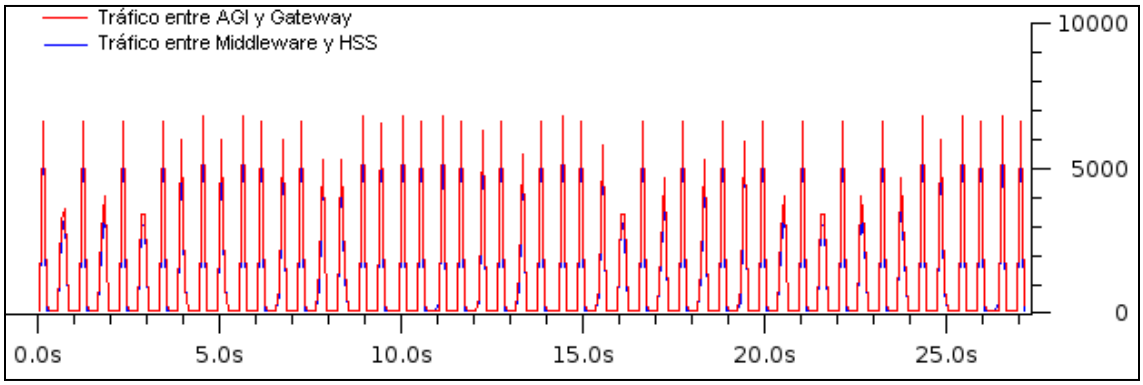

Figura 32. Captura 1 - Escenario 2

Tabla 10. Resultados Captura 1

Tráfico	Escenario 1	Escenario 2
Paquetes	3109	3705
Tiempo entre el primer y el último paquete (segs)	26,704	27,011
Promedio Paquetes/seg	116,423	137,168
Promedio tamaño de paquetes (Bytes)	147,607	171,353
Bytes	458910	634862
Promedio Bytes/seg	17184,876	23504,164
Promedio Mbit/seg	0,137	0,188

- Captura 2

  - Número de actualizaciones: 50
  - Número de usuarios por actualización: 10
  - Retardo entre actualizaciones: 250 ms

En esta captura, al igual que en las siguientes, se puede ver tanto en las gráficas como en las tablas el mismo comportamiento de la captura 1, donde el tráfico azul se incrementa al implementarse IMSeg, pero no se presenta ningún retraso entre salida y llegada de datos. En la sección 5.3.5 se analizarán más a fondo estos resultados.

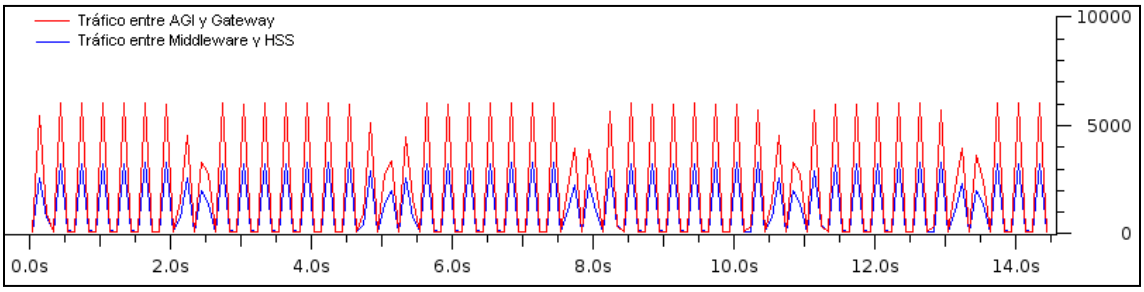

Figura 33. Captura 2 - Escenario 1

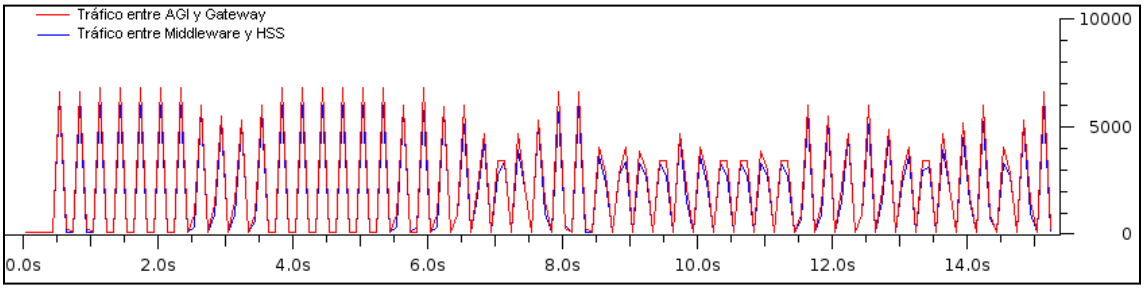

Figura 34. Captura 2 - Escenario 2

Tabla 11. Resultados Captura 2

Tráfico	Escenario 1	Escenario 2
Paquetes	3105	3705
Tiempo entre el primer y el último paquete (segs)	14,265	14,708
Promedio Paquetes/seg	217,658	251,910
Promedio tamaño de paquetes (Bytes)	147,706	171,424
Bytes	458626	635126
Promedio Bytes/seg	32149,350	43183,422
Promedio Mbit/seg	0,257	0,345

- Captura 3
  - Número de actualizaciones: 50
  - Número de usuarios por actualización: 10
  - Retardo entre actualizaciones: 125 ms

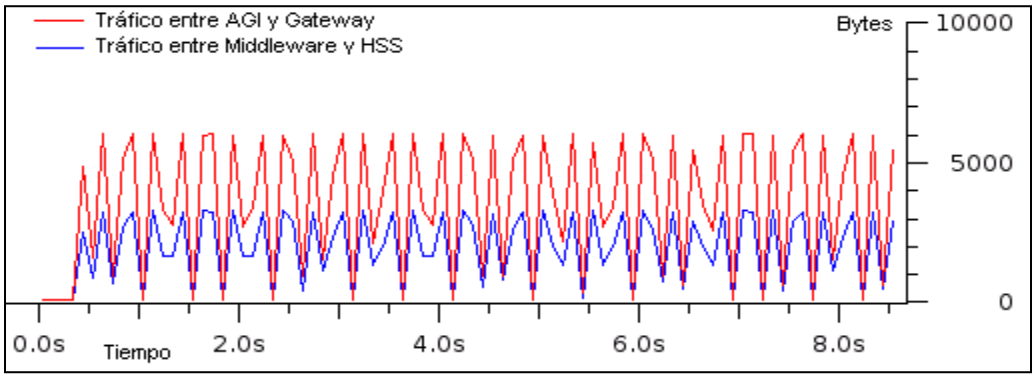

Figura 35. Captura 3 - Escenario 1

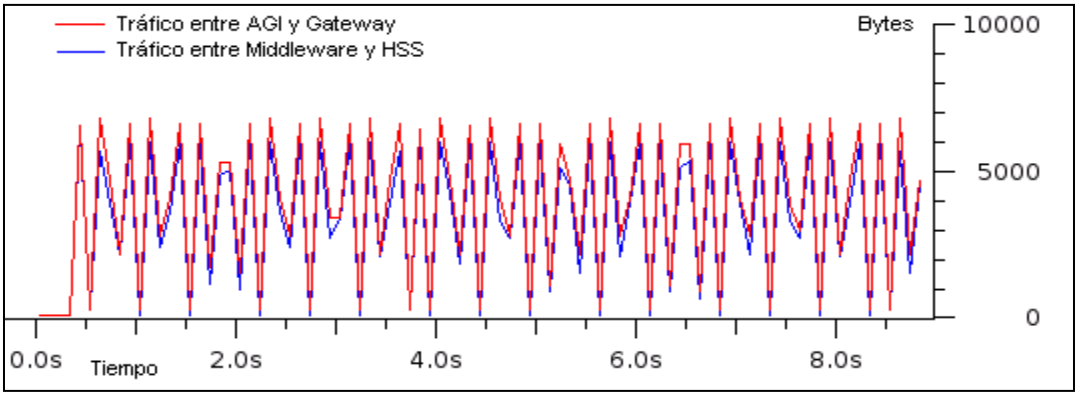

Figura 36. Captura 3 - Escenario 2

Tabla 12. Resultados Captura 3

Tráfico	Escenario 1	Escenario 2
Paquetes	3105	3708
Tiempo entre el primer y el último paquete (segs)	8,148	8,445
Promedio Paquetes/seg	381,094	439,066
Promedio tamaño de paquetes (Bytes)	147,680	171,224
Bytes	458546	634900
Promedio Bytes/seg	56279,887	75178,845
Promedio Mbit/seg	0,450	0,601

- Captura 4
  - Número de actualizaciones: 50
  - Número de usuarios por actualización: 50
  - Retardo entre actualizaciones: 500 ms

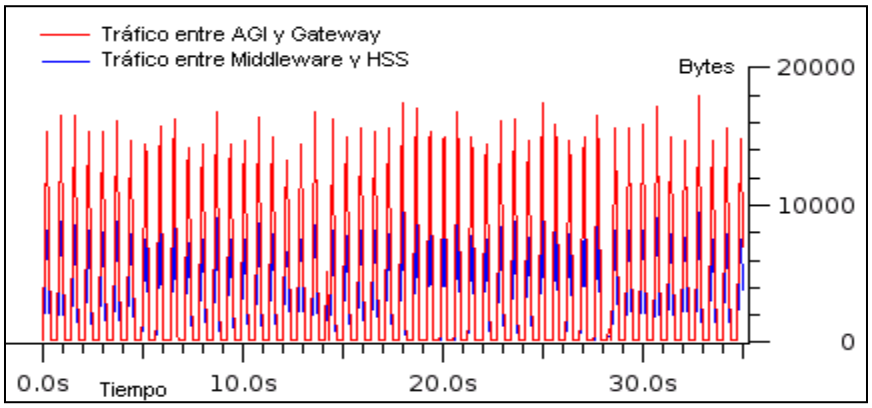

Figura 37. Captura 4 - Escenario 1

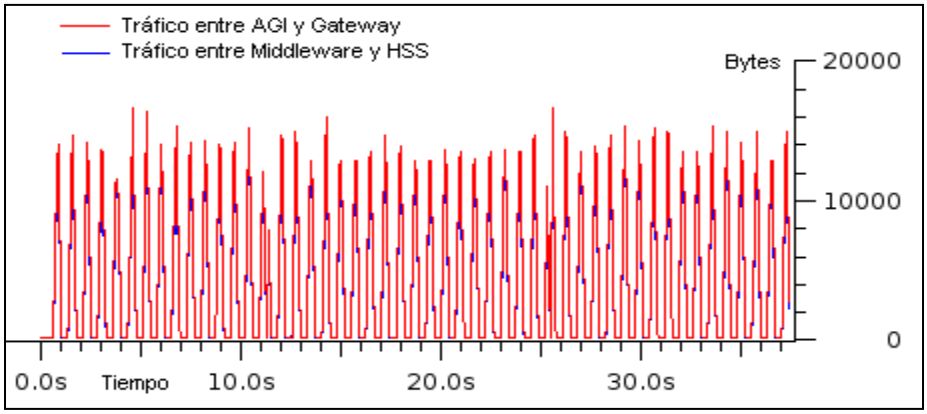

Figura 38. Captura 4 - Escenario 2

Tabla 13. Resultados Captura 4

Tráfico	Escenario 1	Escenario 2
Paquetes	15116	17719
Tiempo entre el primer y el último paquete (segs)	34,934	36,703
Promedio Paquetes/seg	432,705	482,761
Promedio tamaño de paquetes (Bytes)	150,183	174,881
Bytes	2270172	3098710
Promedio Bytes/seg	64985,046	84425,611
Promedio Mbit/seg	0,520	0,675

- Captura 5
  - Número de actualizaciones: 50
  - Número de usuarios por actualización: 50
  - Retardo entre actualizaciones: 250 ms

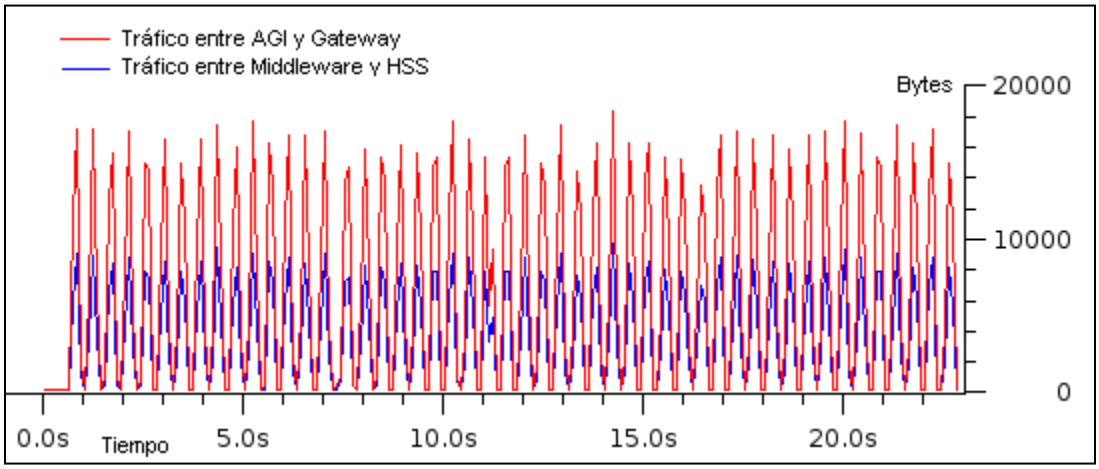

Figura 39. Captura 5 - Escenario 1

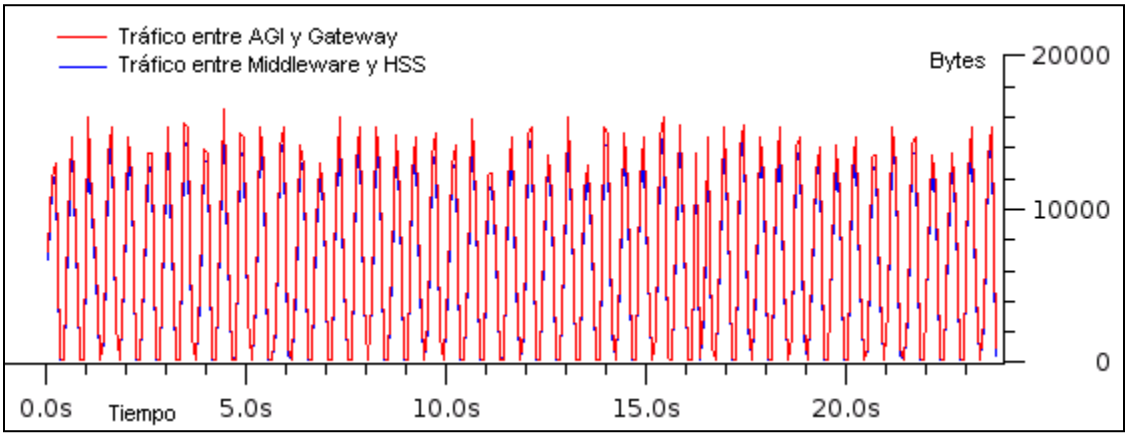

Figura 40. Captura 5 - Escenario 2

Tabla 14. Resultados Captura 5

Tráfico	Escenario 1	Escenario 2
Paquetes	15123	17725
Tiempo entre el primer y el último paquete (segs)	22,080	23,728
Promedio Paquetes/seg	684,924	747,011
Promedio tamaño de paquetes (Bytes)	150,111	174,802
Bytes	2270134	3098374

Tráfico	Escenario 1	Escenario 2
Promedio Bytes/seg	102814,845	130579,443
Promedio Mbit/seg	0,823	1,045

- Captura 6
  - Número de actualizaciones: 50
  - Número de usuarios por actualización: 50
  - Retardo entre actualizaciones: 125ms

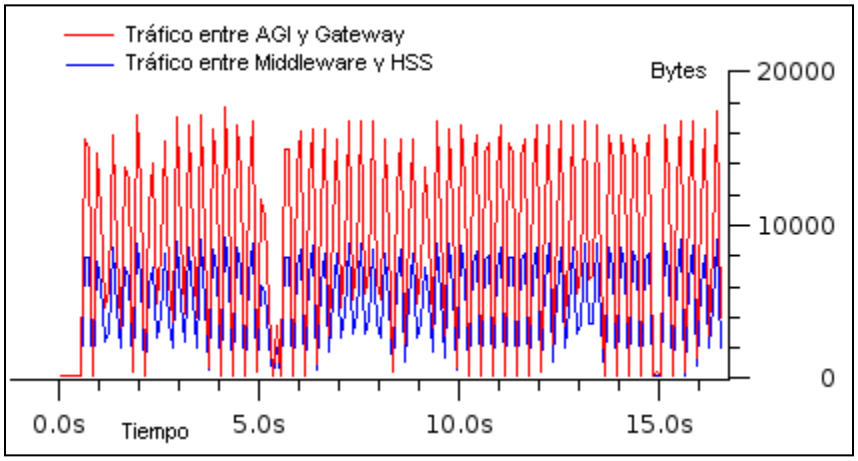

Figura 41. Captura 6 - Escenario 1

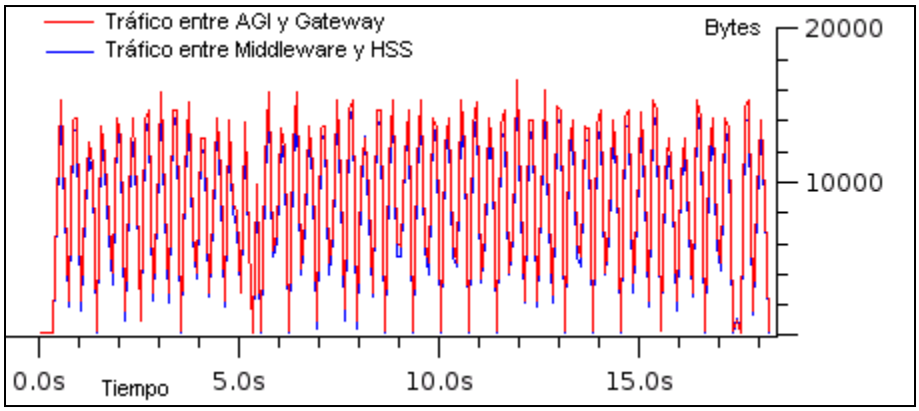

Figura 42. Captura 6 - Escenario 2

Tabla 15. Resultados Captura 6

Tráfico	Escenario 1	Escenario 2
Paquetes	15118	17721
Tiempo entre el primer y el último paquete (segs)	15,951	17,782

Tráfico	Escenario 1	Escenario 2
Promedio Paquetes/seg	947,762	996,592
Promedio tamaño de paquetes (Bytes)	150,172	174,846
Bytes	2270304	3098442
Promedio Bytes/seg	142327,522	174249,861
Promedio Mbit/seg	1,139	1,394

- Captura 7
  - Número de actualizaciones: 50
  - Número de usuarios por actualización: 100
  - Retardo entre actualizaciones: 500ms

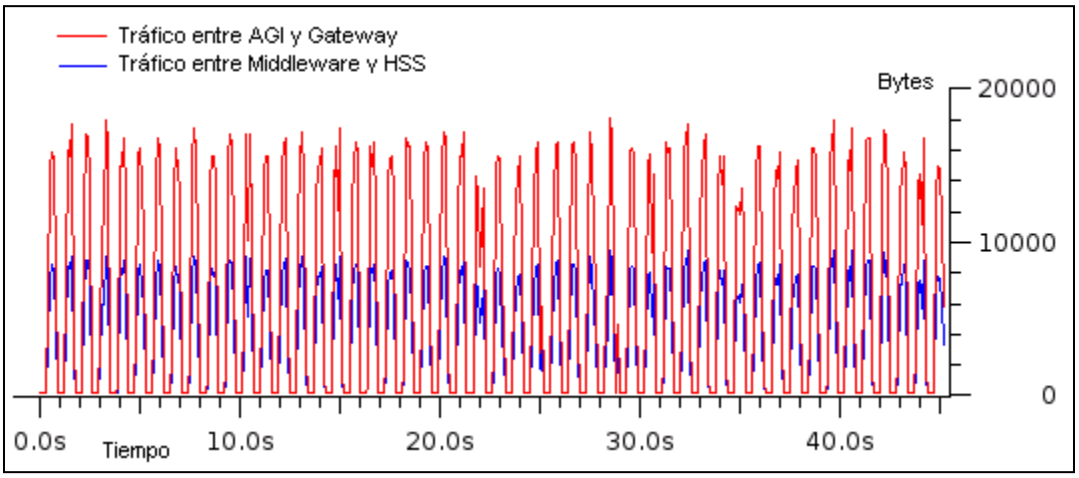

Figura 43. Captura 7 - Escenario 1

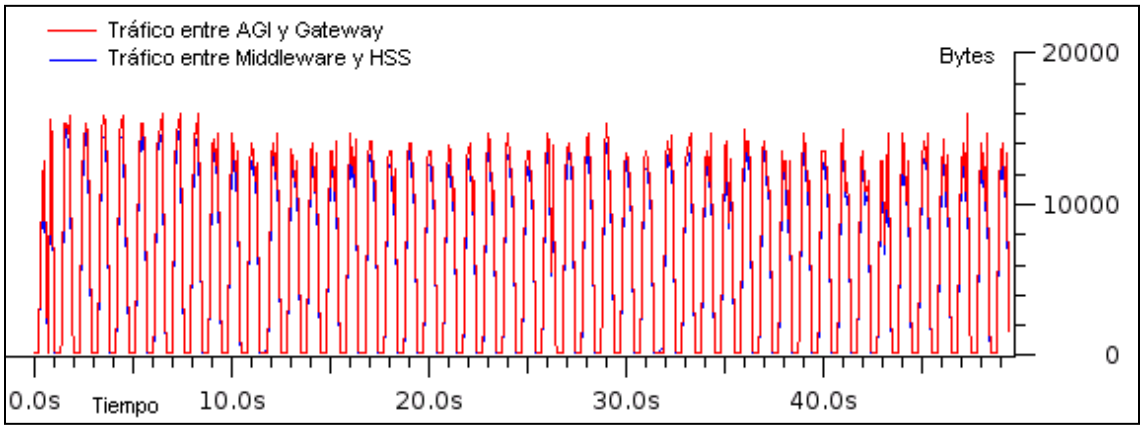

Figura 44. Captura 7 - Escenario 2

Tabla 16. Resultados Captura 7

Tráfico	Escenario 1	Escenario 2
Paquetes	30130	35233
Tiempo entre el primer y el último paquete (segs)	44,810	49,133
Promedio Paquetes/seg	672,392	717,089
Promedio tamaño de paquetes (Bytes)	150,468	175,290
Bytes	4533596	6175994
Promedio Bytes/seg	101173,311	125698,639
Promedio Mbit/seg	0,809	1,006

- Captura 8
  - Número de actualizaciones: 50
  - Número de usuarios por actualización: 100
  - Retardo entre actualizaciones: 250ms

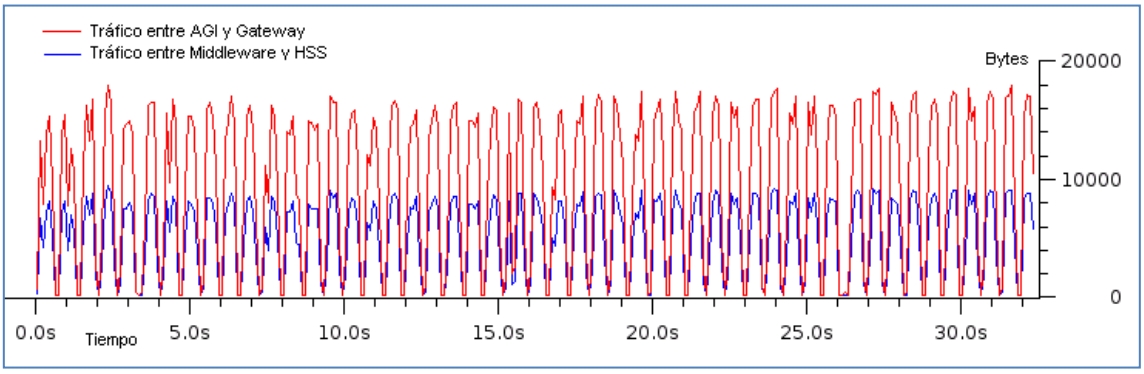

Figura 45. Captura 8 - Escenario 1

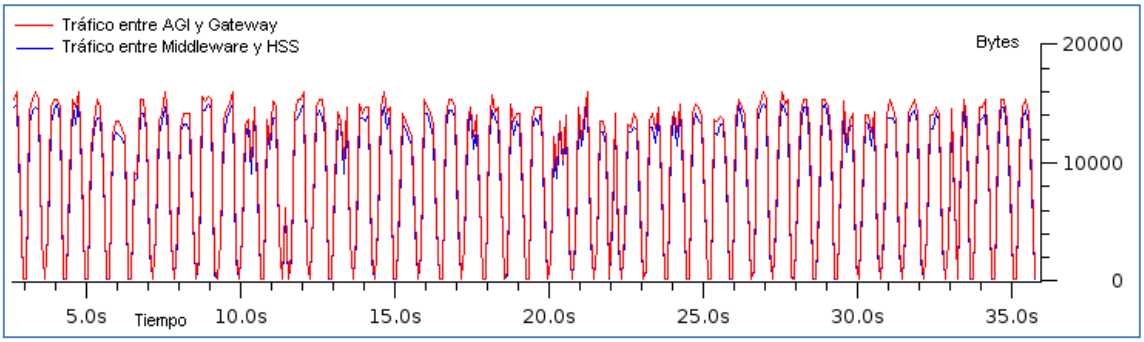

Figura 46. Captura 8 - Escenario 2

Tabla 17. Resultados Captura 8

Tráfico	Escenario 1	Escenario 2
Paquetes	30145	35235
Tiempo entre el primer y el último paquete (segs)	32,301	35,514
Promedio Paquetes/seg	933,243	992,135
Promedio tamaño de paquetes (Bytes)	150,452	175,322
Bytes	4535390	6177462
Promedio Bytes/seg	140408,700	173942,908
Promedio Mbit/seg	1,123	1,392

- Captura 9
  - Número de actualizaciones: 50
  - Número de usuarios por actualización: 100
  - Retardo entre actualizaciones: 125 ms

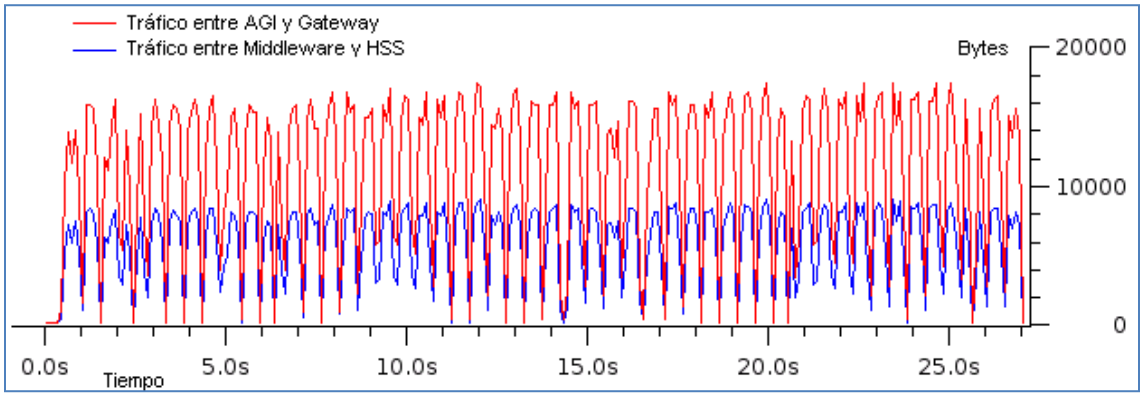

Figura 47. Captura 9 - Escenario 1

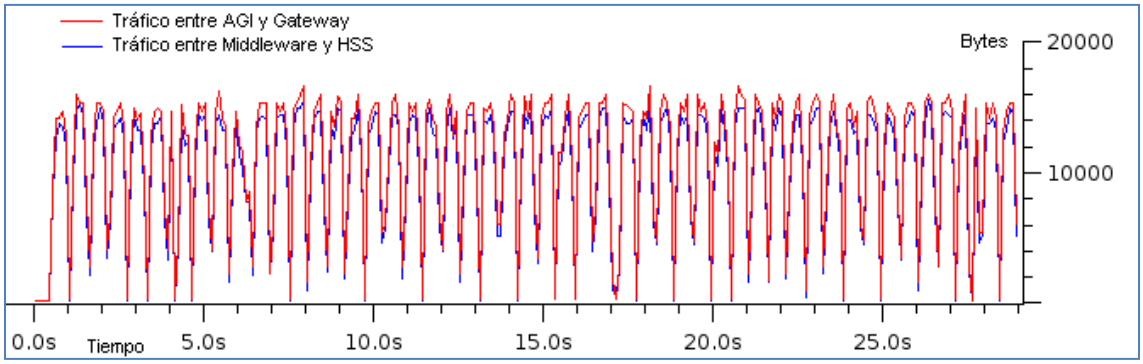

Figura 48. Captura 9 - Escenario 2

Tabla 18. Resultados Captura 9

Tráfico	Escenario 1	Escenario 2
Paquetes	30137	35237
Tiempo entre el primer y el último paquete (segs)	26,527	28,448
Promedio Paquetes/seg	1136,083	1238,636
Promedio tamaño de paquetes (Bytes)	150,302	175,346
Bytes	4529666	6178662
Promedio Bytes/seg	170756,163	217189,659
Promedio Mbit/seg	1,366	1,738

5.3.5. Análisis de los resultados

Las Figura 49, muestra el comportamiento de los dos escenarios de pruebas tomando el parámetro *Bytes* de cada una de las tablas anteriores, graficándose contra el retardo entre actualizaciones para cada captura. Si bien, se observa el incremento del número de KBytes generados al utilizar IMSeg, se aprecia que este número se mantiene prácticamente constante mientras el retardo entre actualizaciones varía, lo que indica que los componentes de red procesan de forma adecuada las peticiones que se están generando hasta para el caso más exigente (100 usuarios cada 125ms) y las retransmisiones por perdida de paquetes son casi imperceptibles, por consiguiente el incremento del tráfico debido a los procesos de seguridad no representa un inconveniente.

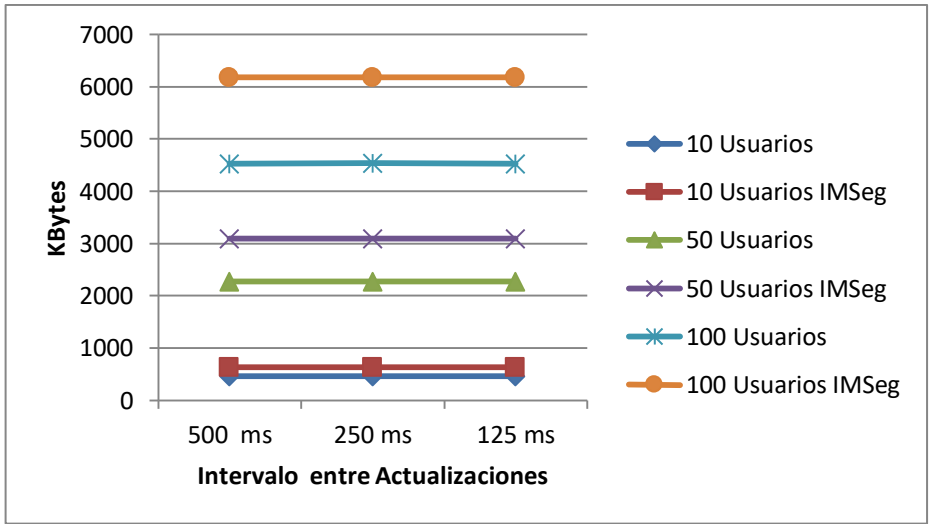

Figura 49. Comparación del número de KBytes variando el Retardo de Actualización en los dos Escenarios

Lo anterior se puede confirmar al observar la Figura 50, en la cual se comparan las velocidades en KBytes por segundo.

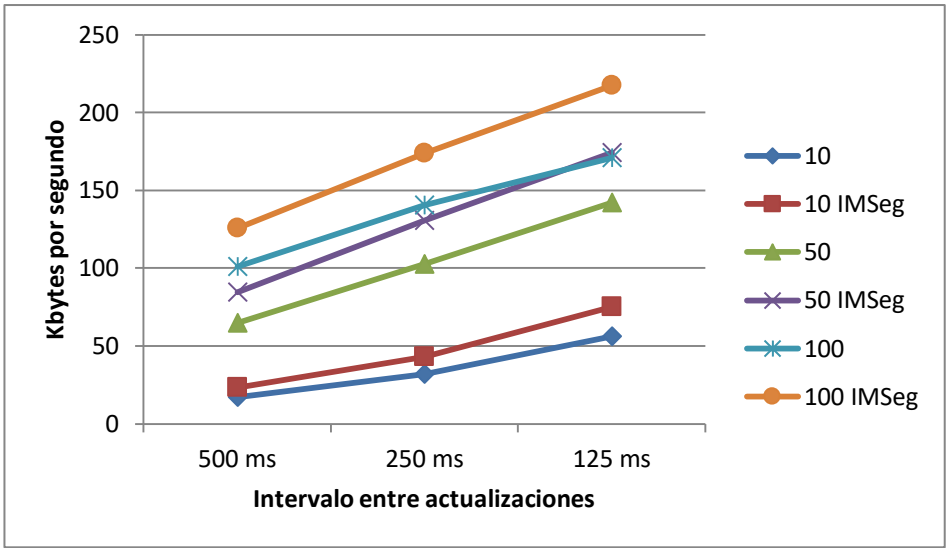

Figura 50. Comparacion de velocidades de transmision de datos para los dos escenarios

En la gráfica anterior se puede apreciar que a medida que se disminuye el intervalo entre actualizaciones el número de KBytes que se transmite por la red aumenta de manera casi lineal, esto para poder procesar adecuadamente cada grupo de peticiones antes de que se envíe el siguiente, de igual manera se observa que la red con IMSeg se comporta de manera similar a la red sin este mecanismo.

A medida que el número de peticiones por segundo aumenta, es decir que el intervalo entre actualizaciones disminuye y el número de usuarios crece, se observa una disminución en el tiempo total de respuesta en ambos escenarios, lo cual muestra que la red con o sin IMSeg atiende eficientemente todas las peticiones hasta en el caso más crítico que se ha probado. Esto se puede corroborar tanto en las tablas 19 y 20 donde se consignan los valores del tiempo total para cada captura en los escenarios 1 y 2 respectivamente como en la Figura 51 donde se comparan los valores de las estas dos tablas.

Tabla 19. Tiempo total de respuesta para el Escenario 1 (resultados en segundos)

Escenario 1	**500 ms**	**250 ms**	**125 ms**
10 Usuarios	26,704	14,265	8,148
50 Usuarios	34,934	22,08	15,951
100 Usuarios	44,81	32,301	26,527

Tabla 20. Tiempo total de respuesta para el Escenario 2 (resultados en segundos)

Escenario 2	**500 ms**	**250 ms**	**125 ms**
10 Usuarios	27,011	14,708	8,445

Escenario 2	500 ms	250 ms	125 ms
50 Usuarios	36,703	23,728	17,782
100 Usuarios	49,133	35,514	28,448

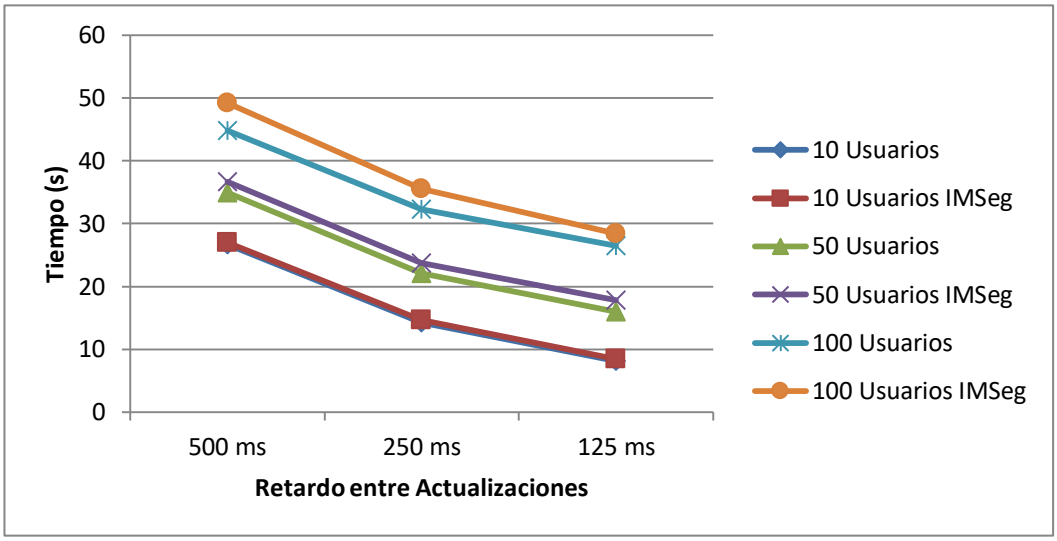

Figura 51. Comparacion del tiempo que tomó llevar a cabo todas las Actualizaciones variando el Retardo de Actualización en los dos Escenarios

Debido al aumento de bytes cuando se implementa IMSeg, es de esperarse que el tiempo total se incremente en alguna medida, lo cual se refleja en la Figura 51, sin embargo no se observa un incremento considerable de este tiempo, pues las líneas que representan al escenario 2 están muy cercanas y tienen prácticamente el mismo comportamiento a las del escenario 1, esto comprueba que el sistema sigue respondiendo de manera adecuada en cada actualización.

Teniendo en cuenta el análisis realizado en SincIMS, es posible determinar un promedio del tiempo que tardan los diferentes componentes de la red en procesar un grupo de peticiones de actualización del siguiente modo:

El número de usuarios también indica el número de peticiones que se envían en un grupo, es decir 10, 50 o 100 peticiones en un grupo y cada grupo se repite 50 veces. Lo que también se varía es el retardo entre cada grupo de peticiones. Por ejemplo para el caso de un retardo de 250 ms y 10 usuarios, se estaría enviando 4 grupos de 10 peticiones por segundo.

De esta manera, si se toma el tiempo total de respuesta y se divide entre 50, se tendría el tiempo promedio de respuesta para un grupo de peticiones incluido el retardo de actualización. Luego, restando este retardo se tendría un promedio del tiempo del procesamiento de un grupo de peticiones de actualización. La ecuación que sintetiza esto es la siguiente:

$$TPP = \left(\frac{TTR}{50} \right) - RA$$

TPP= Tiempo Promedio de Procesamiento (segundos)
TT= Tiempo Total de Respuesta (segundos)
RA= Retardo de Actualización (segundos)

Así, tomando los valores de tiempo total de respuesta de las tablas 19 y 20 y aplicando la formula anterior se obtienen los valores del tiempo promedio de procesamiento, los cuales se condensan en las tablas 21 y 22.

Tabla 21. Tiempo Promedio de Procesamiento de las Peticiones de Actualización en el Escenario 1

Escenario 1	**0,5 s**	**0,25 s**	**0,125 s**
10 Usuarios	0,03408	0,0353	0,03796
50 Usuarios	0,19868	0,1916	0,19402
100 Usuarios	0,3962	0,39602	0,40554

Tabla 22. Tiempo Promedio de Procesamiento de las Peticiones de Actualización en el Escenario 2

Escenario 2	**0,5 s**	**0,25 s**	**0,125 s**
10 Usuarios	0,04022	0,04416	0,0439
50 Usuarios	0,23406	0,22456	0,23064
100 Usuarios	0,48266	0,46028	0,44396

Las tablas anteriores muestran que casi todos los valores de tiempo están por debajo del valor del retardo de actualización, lo que indica que los componentes de la red alcanzan a procesar completamente un grupo de peticiones de actualización antes de que se genere otro, con esto se garantiza la sincronización oportuna y consistente de la información.

En ambos escenarios se presentan dos excepciones (100 usuarios con 250 ms y 125 ms de retardo de actualización), en las cuales el tiempo promedio de procesamiento es mayor al retardo de actualización. En estos casos la información presente en las bases de datos corresponderá a una versión inmediatamente anterior a la que debería presentarse.

Sin embargo, se observa que la adición de IMSeg a la red no ha sido un factor determinante ya que las dos excepciones mencionadas también se presentan en el Escenario 1. Además La comparación entre las tablas 21 y 22 indica es que los tiempos son muy similares para los dos escenarios mostrando una vez más que el uso de IMSeg aumenta en una proporción aceptable la carga en la red.

De todas formas para el caso del servicio simulado de localización, con un usuario que se mueve en un automóvil a 80 Km/h se puede asegurar que las coordenadas que se tienen en las bases de datos del AS y del HSS corresponden a la ubicación actual ya que el mayor retardo que se introduce es de 0,44396 segundos correspondiente a 9,68 metros, que es una distancia menor a la resolución del GPS. Por lo tanto aunque el retardo esté presente, éste no va a ser relevante dado que una variación en la posición menor a 10 metros no es detectable por el GPS.

La Figura 52 muestra la comparación de los valores del tiempo promedio de procesamiento de las peticiones de actualización consignados en las tablas 21 y 22. En esta se puede apreciar que para un número de usuarios dado (10, 50 o 100), este tiempo tiende a mantenerse constante en ambos escenarios, lo cual indica que el sistema en general procesa la información del mismo modo con o sin IMSeg.

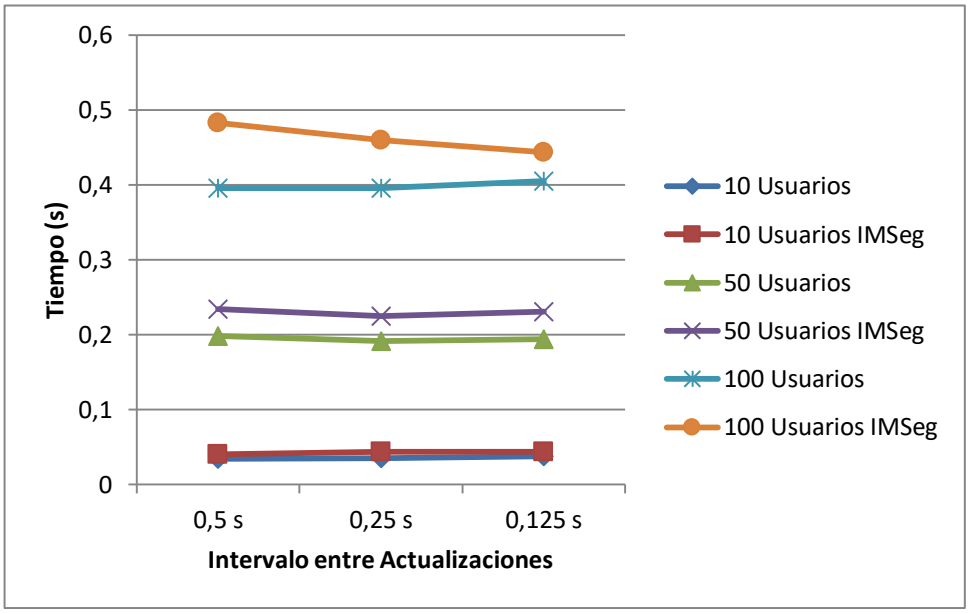

Figura 52. Comparación del Tiempo Promedio de Procesamiento de las peticiones de actualización entre los dos escenarios

Finalmente, como punto de referencia se presentan los resultados obtenidos en SincIMS donde se compara el tiempo total de respuesta entre el método tradicional de sincronización usado en IMS (Interfaz Sh - Open IMS *Core*) y el método que utiliza el *Middleware Sequoia* [6]. Cabe aclarar que, si bien se realizaron pruebas similares, las condiciones de la red y de los equipos utilizados en SincIMS fueron diferentes al presente trabajo, por lo tanto los resultados tienen valores distintos, pero en términos generales el comportamiento que se obtuvo al implementar la red sin seguridad es el mismo al obtenido en el trabajo de referencia.

En la Figura 53 se compara el tiempo total de respuesta entre el mecanismo de sincronización de datos de usuario nativo de IMS (Interfaz Sh) y el mecanismo que utiliza Sequoia para un retardo de actualizaciones de 500 ms.

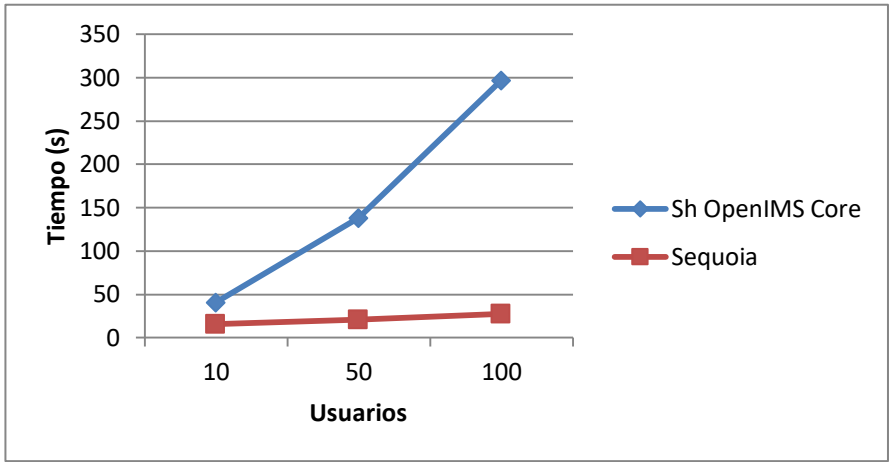

Figura 53. Comparación del tiempo total de respuesta entre el mecanismo nativo de IMS y el mecanismo que utiliza Sequoia con un retardo de actualización de 500 ms

Es notoria la mejoría del rendimiento del método que utiliza el middleware Sequoia, sobre todo cuando se aumenta el número de usuarios, ya que el mecanismo tradicional procesa de manera secuencial cada una de las peticiones de actualización, introduciendo más retardo mientras se aumenta el número de usuarios. En la figura 54 se presenta la misma comparación de la Figura 53 pero entre los Escenarios 1 y 2 del presente trabajo.

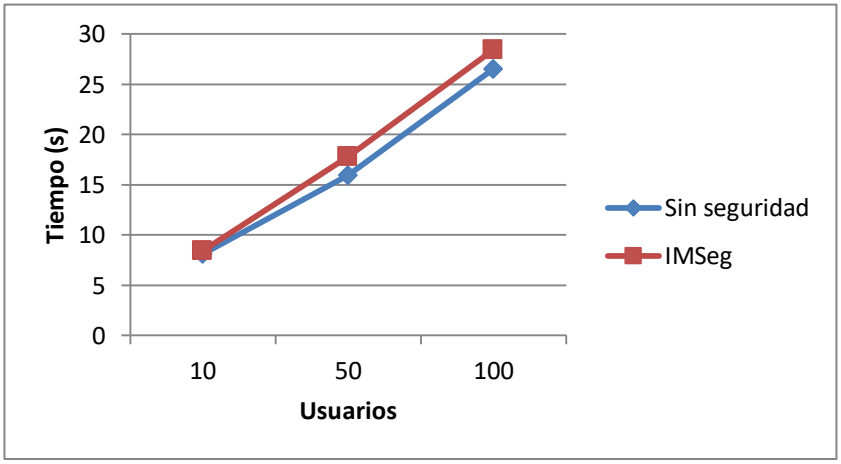

Figura 54. Comparación del tiempo total de respuesta entre los escenarios 1 y 2 del presente trabajo con un retardo de actualización de 500 ms

En la gráfica anterior se puede apreciar claramente que el comportamiento de la red que utiliza IMSeg es muy similar al de la red sin seguridad aumentando el tiempo total de respuesta en una proporcion muy baja.

Las siguientes Figuras presentan las mismas comparaciones anteriores pero para retardos de actualizacion de 250ms y 125 ms.

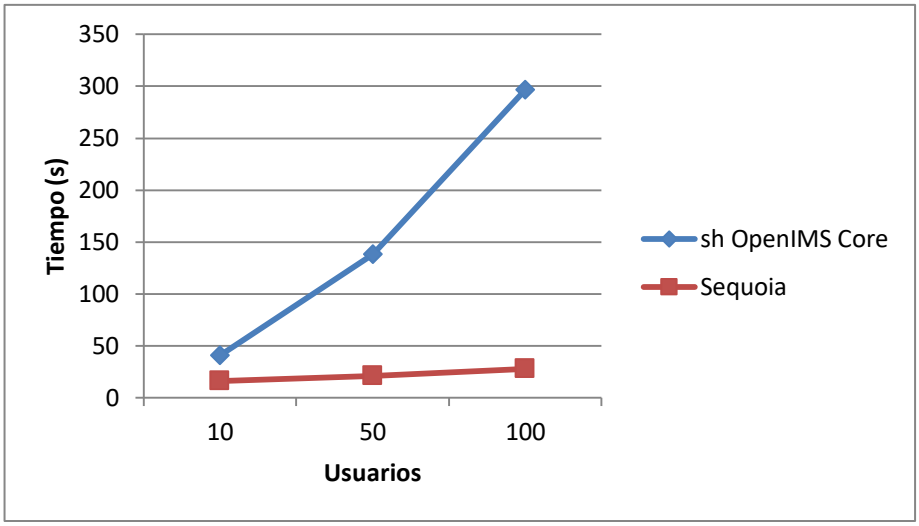

Figura 55. Comparación del tiempo total de respuesta entre el mecanismo nativo de IMS y el mecanismo que utiliza Sequoia con un retardo de actualización de 250 ms

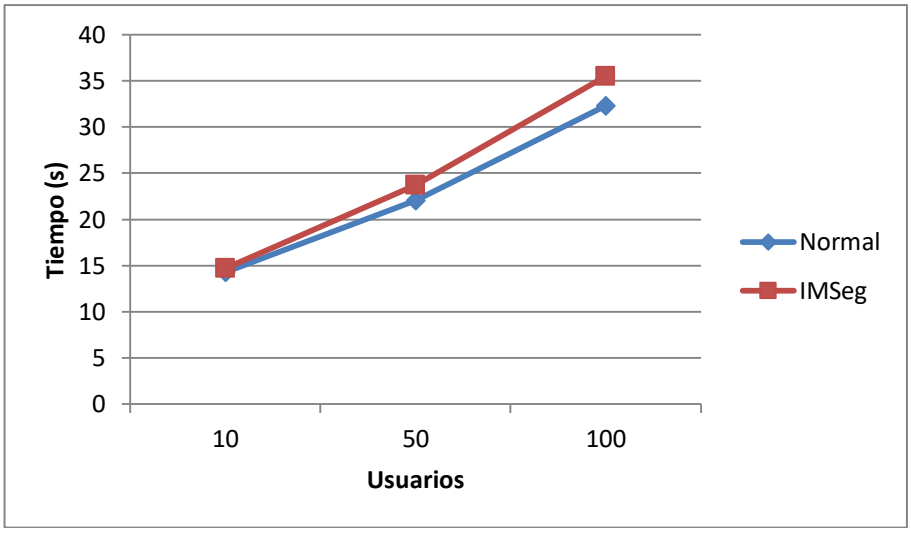

Figura 56. Comparación del tiempo total de respuesta entre los escenarios 1 y 2 del presente trabajo con un retardo de actualización de 250 ms

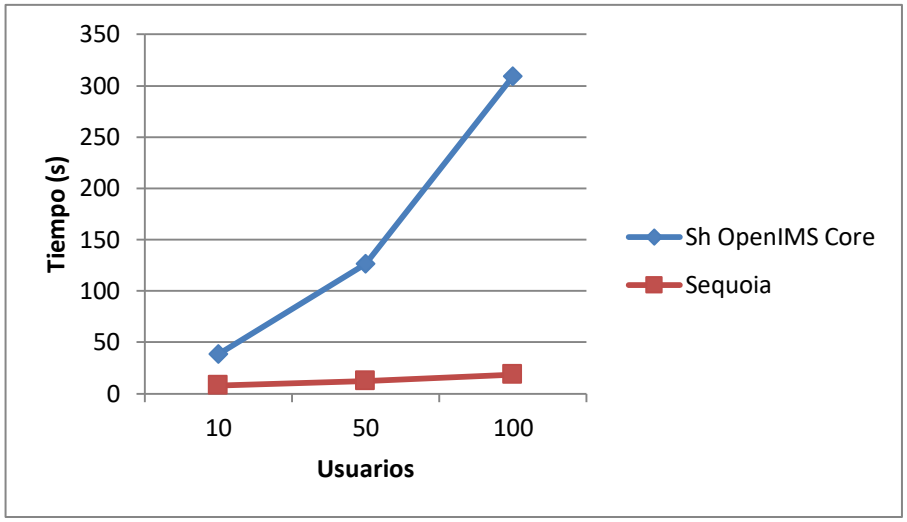

Figura 57. Comparación del tiempo total de respuesta entre el mecanismo nativo de IMS y el mecanismo que utiliza Sequoia con un retardo de actualización de 125 ms

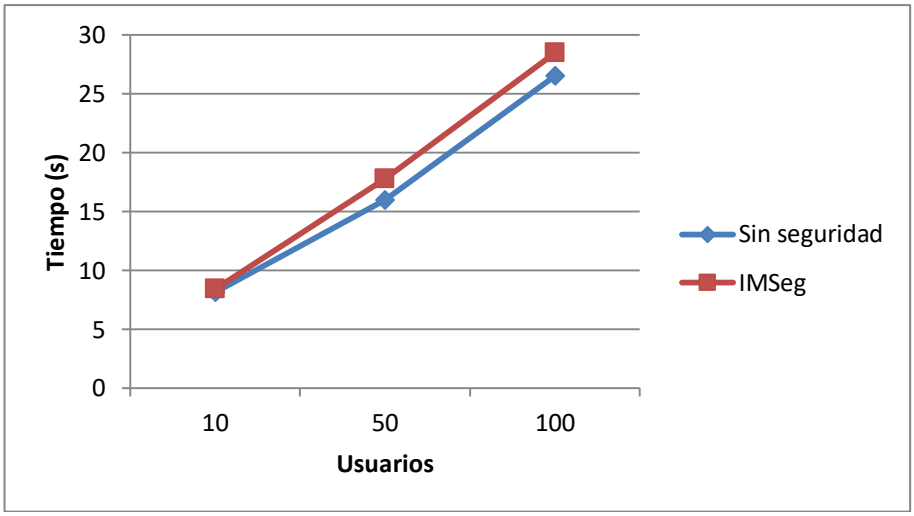

Figura 58. Comparación del tiempo total de respuesta entre los escenarios 1 y 2 del presente trabajo con un retardo de actualización de 125 ms

Las gráficas anteriores muestran prácticamente el mismo comportamiento visto en las Figuras 53 y 54 por lo tanto se deduce que la adición del mecanismo de seguridad IMSeg al método de sincronización de datos de usuario propuesto en SincIMS no deteriora el buen funcionamiento de este, pues si bien se aumenta en una pequeña proporción el número de Bytes que se generan y se transportan por la red, los nodos siguen procesando de manera satisfactoria todas las peticiones y los tiempos de respuesta obtenidos lo demuestran.

De este modo, al observar las gráficas anteriores se puede determinar que el mecanismo SincIMS mejorado para utilizar seguridad mediante IMSeg sigue obteniendo mejor respuesta en tiempo y en procesamiento de las peticiones que el método tradicional de IMS sin utilizar seguridad en las comunicaciones ni en los nodos de la red.

CONCLUSIONES Y TRABAJOS FUTUROS

6.1. Conclusiones

Teniendo en cuenta el componente teórico estudiado en los primeros capítulos y después de la validación del prototipo implementado mediante las pruebas de seguridad y rendimiento se concluye que:

- Entre los protocolos de seguridad en el intercambio de datos vistos aplicables a IMS, IPsec es el que mejor se adapta a las necesidades de seguridad de IMS brindando los servicios de seguridad requeridos.

- La utilización de IPsec, el cual opera en la capa de red, proporciona los servicios de seguridad a los protocolos de capas superiores, haciendo que IMSeg sea una solución de seguridad de red transparente a las aplicaciones, esto permite que tanto los servicios como la señalización de IMS se puedan encapsular y transportar de manera cifrada.

- La combinación de todos los elementos de IMSeg proporciona un alto nivel de prestación de los servicios de seguridad: autenticación, confidencialidad, integridad y no repudio, los cuales son requerimientos esenciales en un entorno IMS.

- La incorporación de IMSeg en un modelo de SecIMS no ocasionó mayores efectos en el nivel de desempeño de este, pues si bien se genera un tráfico de información extra, proveniente de los procesos de seguridad de IPsec, este aumento de tráfico no es significativo ni ocasiona retardos o pérdidas de paquetes.

- El hecho de que IPsec sea transparente a los servicios y señalización de IMS, hace que una adaptación de IMSeg pueda trabajar con los protocolos nativos de IMS sin la necesidad de realizar cambios de configuración del *Core* IMS ni alterar su desempeño.

- Los filtros impuestos a las aplicaciones en IMSeg proporcionan una alta protección contra configuraciones deficientes o software malicioso instalado en los equipos como *malware* y garantizan un alto control en cuanto a la información que entra o sale del servidor y las conexiones establecidas.

- La unión de las políticas de seguridad firewall y los túneles IPsec garantizan el acceso únicamente a los usuarios o equipos autorizados mediante comunicaciones seguras.

- La utilización de certificados digitales X.509 garantiza un manejo más seguro de la autenticación entre equipos y el establecimiento de Asociaciones de Seguridad que el uso de claves pre-compartidas.

- El prototipo de IMSeg se implementó con herramientas de Código Abierto casi en su totalidad, lo cual garantiza su adaptabilidad hacia otros entornos realizando los cambios necesarios, sin la necesidad del pago de licencias de ningún tipo y con el

respaldo que ofrecen las distintas comunidades de desarrolladores, colaboradores y usuarios de este tipo de software alrededor del mundo.

- La utilización de versiones "fuertes" de protocolos seguros como ESP, IKEv2, cifrado AES y SHA2, proporcionan alto grado de confiabilidad en el nivel de seguridad de las comunicaciones en IMSeg.

- El desempeño de la sincronización de los datos de usuario en el entorno de las NGN puede mejorarse mediante propuestas como SecIMS, que en unión con IMSeg proporcionan tanto seguridad como un rendimiento adecuado de los recursos de red.

6.2. Trabajos Futuros

Los siguientes proyectos pueden ser tenidos en cuenta para continuar con la investigación en el intercambio seguro de información en NGN:

- Evaluar IMSeg en un ambiente IMS nativo utilizando la interfaz Sh con el fin de verificar la incidencia del mecanismo de seguridad en esta red.

- Realizar estudios de seguridad entre otras entidades de la red IMS, entre estos se puede considerar proporcionar los servicios de seguridad en la autenticación de usuarios con distintas tecnologías de acceso.

- Adaptar IMSeg para proporcionar los servicios de seguridad requeridos entre dominios IMS.

BIBLIOGRAFÍA

[1]. O. González, "Concepto y arquitectura de las redes NGN", 2006. [En Línea]. Disponible: http://www.itu.int/ITU-D/finance/work-cost-tariffs/events/tariff-seminars/rio_de_janeiro-06/gonzalez-1-sp.pdf

[2]. F. García, "La próxima generación de redes, NGN, un trayecto hacia la convergencia", 2006. [En Línea]. Disponible: http://sociedaddelainformacion.telefonica.es/documentos/articulos/B_A%20FONDO_redesNGN.pdf

[3]. S. Znaty, J. Dauphin, R. Geldwerth, "IP Multimedia Subsystem: Principios y Arquitectura," EFORT, 2007.

[4]. J. Ramió, "SEGURIDAD INFORMÁTICA", Universidad Politécnica de Madrid, España, Sexta edición, 1 de Marzo 2006

[5]. 3GPP TS 33.203, "Access security for IP-based services," 3GPP, 3G Security, Technical Specification Group Services and System Aspects, 2009.

[6]. E. Armero, D Rodríguez, "Sincronización de datos de usuario en redes de próxima generación", Trabajo de Grado, Popayán, Colombia, sep. 2008.

[7]. P. Herzog, "OSSTMM - Open Source Security Testing Methodology Manual," 2009, [En línea]. Disponible: http://www.isecom.org/osstmm

[8]. 3GPP TS 33.210, "Network Domain Security; IP network layer security," 3GPP, 3G Security, Technical Specification Group Services and System Aspects, 2008.

[9]. 3GPP TS 29.109, "Generic Authentication Architecture (GAA); Generic bootstrapping architecture," 3GPP, Technical Specification Group Services and System Aspects, 2008.

[10]. W. Richard Stevens, "TCP/IP Illustrated Volume 1: The protocols", Addison-Wessley, 1998

[11]. ITU-T, "Definición de NGN para la ITU-T" 2009. [En Línea]. Disponible: http://www.itu.int/ITU-T/ngn/definition.html

[12]. Oscar J. Calderón C., "Definición y conceptos asociados a la NGN", 2009. [En Línea]. Disponible: http://calypso.unicauca.edu.co/gntt/oscarc/NGN-I-08/NGN-02-DEFINI-09%20%5BModo%20de%20compatibilidad%5D.pdf

[13]. D. Yan, F Yang, "Vulnerability analysis of service architecture in NGN", 2009, IEEE.

[14]. W. Stallings, "Fundamentos de seguridad en redes, aplicaciones y estándares", 2ª Edición, Pearson Prentice Hall, 2004.

[15]. J. Shimonski, W. Eaton, U. Khan, Y. Gordienko, "Sniffer: Network optimization and troubleshooting handbook", 2002, Syngress.

[16]. Segu-Info, "Amenazas Lógicas - Tipos de Ataques - Ataques de Autenticación", [En Línea]. Disponible: http://www.segu-info.com.ar/ataques/ataques_autenticacion.htm.

[17]. HackMex, revista electrónica. "Man in the Middle", jun 2007. [En Línea]. Disponible: http://www.scribd.com/doc/209821/hackmexnumero1

[18]. Segu-Info, "Amenazas Lógicas - Tipos de Ataques – Denial of Service (DoS)", [En Línea]. Disponible: http://www.segu-info.com.ar/ataques/ataques_dos.htm.

[19]. J. Barret, E. Silverman, G. Byrnes, "SSH, The Secure Shell: The definitive guide", 2ª Edición, O'Reilly, 2005

[20]. J. Gutiérrez, J. Tena (eds.), "Protocolos criptográficos y seguridad en redes", Universidad de Cantabria, 2003.

[21]. N. Doraswamy, D. Harkins, "IPSec: the new security standard for the Internet, Intranet and Virtual Private Networks", 2ª Edición, Prentice Hall, 2002.

[22]. J. Cuesta, M. Puñales, "PKI Infraestructura de Clave Pública Seguridad en Redes Telemáticas," Universidad Politécnica de Madrid, 2002.

[23]. 3G Americas, "IMS Application Enabler and UMTS/HSPA Growth Catalyst," 3G Americas, 2006.

[24]. Ericsson, "IMS – IP Multimedia Subsystem. The value of using the IMS architecture," Ericsson, 2004.

[25]. 3GPP TS 23.228, "IP Multimedia Subsystem (IMS)," 3GPP, Technical Specification Group Services and System Aspects, 2009.

[26]. 3GPP TS 23.002, "Network Arquitecture," 3GPP, Technical Specification Group Services and System Aspects, 2009.

[27]. 3GPP TS 29.329, "Sh Interface based on the Diameter protocol," 3GPP, Technical Specification Group Core Network and Terminals, 2009.

[28]. 3GPP TS 23.008, "Organization of subscriber data," 3GPP, Technical Specification Group Core Network and Terminals, 2008.

[29]. 3GPP TS 33.978, "Security aspects of early IP Multimedia Subsystem (IMS)," 3GPP, 3G Security, Technical Specification Group Services and System Aspects, 2008.

[30]. F. Park, D. Patnaik, C. Amrutkar, "A Security Evaluation of IMS Deployments," Georgia Institute of Technology / Georgia Tech Information Security Center (GTISC), Atlanta, USA, 2008.

[31]. W. Kellerer, M. Wagner, W. Balke, Wolf-Tilo, "Preference-based Session Management for Personalized Services. MoMuC (Mobile Multimedia Communications)," 2003. [En Línea]. Disponible: http://www.l3s.de/~balke/paper/momuc03.pdf

[32]. 3GPP, "Service Aspects; Service Principles," 3GPP, Technical Specification, 2007.

[33]. UIT-T, "Requisitos de seguridad para las redes de la próxima generación", UIT-T Recomendación Y.2701, versión 1, 2007

[34]. Sequoia Project, "Continuent.org Sequoia 3.0 Basic Concepts," 2008. [En Línea]. Disponible: http://www.continuent.com/community/lab-projects/sequoia

[35]. Fraunhofer Institut, "Open IMS Core's Homepage," 2008. [En línea]. Disponible: http://www.openimscore.org/

[36]. Oracle Corporation, "Weblogic Sip Server Homepage," 2009. [En Línea]. Disponible: http://e-docs.bea.com/wlcp/wlss31

[37]. Debian Project, "Debian GNU/Linux Homepage," 2009. [En Línea]. Disponible: http://www.debian.org

[38]. Xelerance Corporation, "Openswan Homepage," 2009. [En línea]. Disponible: http://www.openswan.org

[39]. P. Eronen, J. Korhonen, "Multiple Authentication Exchanges in the Internet Key Exchange (IKEv2) Protocol," Network Working Group Request for Comments: 4739, 2006.

[40]. A. Steffen, "The Open Source VPN Solution for Mixed Platforms," Institute for Internet Technologies and Applications, Oberseestrasse, Suiza, 2009. [En Línea]. Disponible: http://www.strongswan.org/docs/LinuxTag2009_Flyer_VPN_Solution.pdf

[41]. WIDE project, "The racoon2 project Homepage," 2009. [En Línea]. Disponible: http://www.racoon2.wide.ad.jp/w/

[42]. KAME project, "The KAME project Homepage," 2006. [En Línea]. Disponible: http://www.kame.net

[43]. R. Marín, A. Gómez, G. Martínez, "OpenIKEv2 Homepage," ANTS research group, Universidad de Murcia, Murcia, España, 2008. [En Línea]. Disponible: http://openikev2.sourceforge.net/

[44]. Netfilter core team, "Netfilter/iptables Homepage," 2008. [En Línea]. Disponible: http://www.netfilter.org/

[45]. W. Venema "TCP WRAPPER Network monitoring, access control, and booby traps," Mathematics and Computing Science Eindhoven University of Technology, Eindhoven, Paises Bajos. 2002.

[46]. The OpenSSL Project, "OpenSSL Project Homepage," 2009. [En Línea]. Disponible: http://www.openssl.org/

[47]. The OpenSSL Project, "OpenSSL Related Applications," 2009. [En Línea]. Disponible: https://www.openssl.org/related/apps.html

[48]. A. Reelsen, J. Fernández, S. Peña, "Securing Debian Manual," 2007. [En Línea]. Disponible: http://www.debian.org/doc/manuals/securing-debian-howto/#contents

[49]. B. Spengler, "Grsecurity Project Homepage," 2009. [En Línea]. Disponible: http://www.grsecurity.net

[50]. H. Francisconi, "IPsec en Ambientes IPv4 e IPv6," H. Francisconi, Villa Nueva Mendoza, Argentina, 2005.

[51]. C. Guevara, F. Mera, "Criterios Para Establecer Políticas De Seguridad De La Información Y Plan De contingencia, Caso de Estudio El Centro de Datos de La Universidad del Cauca," Trabajo de Grado, Popayán, Colombia, feb. 2008.

[52]. Wireshark Foundation, "Wireshark Homepage," 2009. [En Línea]. Disponible: http://www.wireshark.org/

[53]. Openwall Project, "John the Ripper Homepage," 2008. [En Línea]. Disponible: http://www.openwall.com/john/

[54]. Tcpdump Project, "Tcpdump Homepage," 2008. [En Línea]. Disponible: http://www.tcpdump.org/

[55]. Insecure.Org, "Nmap Homepage," 2008. [En Línea]. Disponible: http://www.nmap.org/

[56]. PaX Team, "PaX Homepage," 2008. [En Línea]. Disponible: http://pax.grsecurity.net/

[57]. Ettercap NG Project, "Ettercap Homepage," 2009. [En Línea]. Disponible: http://ettercap.sourceforge.net/