ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

"Software de Tunneling con autenticación de origen y destino".

TESINA DE SEMINARIO

Previa a la obtención del Título de:

INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

Presentada por

CRISTHIAN GERMAN GUERRERO PROAÑO

CARLOS ALBERTO DUCHI CARRILLO

Guayaquil - Ecuador

2011

A G R A D E C I M I E N T O

A Dios, por la bendición de cada día abrir mis ojos y poder disfrutar de todo cuanto me da, al Ing. Ignacio Marin-Garcia por la entrega desinteresada y el tiempo dedicado durante el desarrollo de nuestro proyecto y a mis tíos Ángel y Vicky por facilitarnos los recursos para la implementación del prototipo.

Cristhian Guerrero

A mis Padres por su amor, su esfuerzo y su apoyo incondicional; al Pastor Pablo Alcoser Caicho por sus consejos sabios; a mis amigos por todo el ánimo y paciencia, y sobre todo por su valiosa amistad, y al Msc. Ignacio Marín García por la entrega desinteresada y el tiempo dedicado durante el desarrollo de nuestro proyecto.

Carlos Duchi

D E D I C A T O R I A

A mis padres como regalo, en retribución a todo el esfuerzo empleado hasta ahora por hacer de mi un hombre de bien; a mi hermano y primos como muestra de que los sueños son alcanzables cuando te esfuerzas y luchas por ellos, finalmente a mis abuelos por sus sabios consejos.

Cristhian Guerrero

A Dios a quien amo profundamente; a mis padres: Manuel Duchi Cujilema y María Carrillo Duchi a quienes amo y aprecio grandemente; a mis hermanos: Jonathan y Lady quienes son el motivo de mi esfuerzo y dedicación; y a todos las personas que contribuyeron en mi formación profesional.

Carlos Duchi

TRIBUNAL DE SUSTENTACIÓN

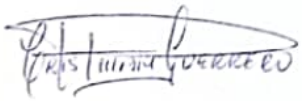
Msc. Ignacio Marin-Garcia
PROFESOR DE LA MATERIA DE GRADUACIÓN

MBA. Marcelo Loor Romero
PROFESOR DELEGADO DEL DECANO

DECLARACIÓN EXPRESA

"La responsabilidad del contenido de esta Tesina, nos corresponde exclusivamente; y el patrimonio intelectual de la misma, a la Escuela Superior Politécnica del Litoral"

(Reglamento de exámenes y títulos profesionales de la ESPOL)



CRISTHIAN GERMAN GUERRERO PROAÑO

CARLOS ALBERTO DUCHI CARRILLO

R E S U M E N

El crecimiento acelerado de los sistemas computacionales y su interconexión por medio de redes, ha logrado que millones de personas puedan llevar a cabo distintas tareas desde cualquier lugar. Es así que por medio de un ordenador se puede acceder de forma rápida y sencilla a múltiples recursos y servicios tales como: correo electrónico, mensajería instantánea, transacciones y transferencias bancarias, entre otros beneficios que se nos brindan por medio de Internet. Esto ha traído soluciones a grandes problemas del pasado como la distancia y el tiempo.

Mucha información personal se mueve en Internet; sin la protección adecuada podría estar disponible para entes no autorizados, pudiendo esto causar: pérdidas económicas, daños a la moral y disminución de la credibilidad en caso de una empresa. Es así que surge la necesidad de que la información que ingrese o que salga de nuestro ordenador se encuentre protegida.

Este proyecto busca explorar las vulnerabilidades existentes al momento de transmitir datos entre sistemas informáticos y hacia Internet para luego desarrollar un prototipo funcional por medio de la integración de técnicas criptográficas y protocolos de seguridad mejorados con la finalidad de brindar mayor protección a estos datos.

ÍNDICE GENERAL

G L O S A R I O

BACKTRACK: Es una distribución Linux pensada y diseñada para la auditoria de seguridad y relacionada con la seguridad informática en general.

CONFIDENCIALIDAD: Garantizar que la información es accesible sólo para aquellos autorizados a tener acceso.

DISPONIBILIDAD: Es la cualidad de mantener accesibles los recursos cuando los usuarios autorizados requieran acceso.

ENRUTAMIENTO: Denominado también encaminamiento es la función de buscar un camino entre todos los posibles de una red de paquetes cuyas topologías poseen una gran conectividad.

ENTIDAD: Para este proyecto se definirá como entidad a una empresa o persona.

ESCALABILIDAD: Es la propiedad deseable de un sistema que indica la habilidad para crecer sin perder la calidad en los servicios ofrecidos.

ETTERCAP: Es un interceptor de paquetes para redes de área local. Soporta direcciones activas y pasivas de varios protocolos (incluso aquellos cifrados, como SSH y HTTPS). También hace posible la inyección de datos en una conexión establecida y filtrado al vuelo aun manteniendo la conexión sincronizada gracias a su poder para establecer un Ataque de Intercepción o Intermediario.

FAIL2BAN: Es una aplicación utilizada para la prevención de intrusos en un sistema, que se basa en la penalización de conexión (bloquear conexión) a los orígenes que intentan accesos por fuerza bruta.

GOOGLE: Es una corporación pública multinacional americana dedicada a ofrecer productos de búsqueda en Internet y tecnologías de anuncios; su principal producto es el motor de búsqueda del mismo nombre, aunque ofrece un gran número de servicios y productos basados en Internet.

GOOGLE-TALK: Es un cliente de mensajería instantánea y VOIP desarrollado por Google

HACKING: Es la explotación de las vulnerabilidades de los sistemas informáticos y sus mecanismos de seguridad a fin de obtener acceso no autorizado a los recursos para intencionalmente o no causar perjuicio.

INTEGRIDAD: Es la propiedad que busca mantener los recursos, libres de

cambios o modificaciones, garantizando la exactitud y completitud de la información.

JMITM2: Es una aplicación basada en la implementación Java SSH desarrollada para llevar ataques de Intermediario a SSH2 en una arquitectura Cliente/Servidor.

MEDUSA: Es una herramienta disponible en la distribución de Linux Bactrack, la cual nos permite realizar ataques de fuerza bruta contra un variado conjunto de protocolos , se destaca por la rapidez en que realiza dichos ataques.

MYSQL: Es un sistema de gestión de bases de datos relacional licenciado bajo la GPL de GNU. Su diseño multihilo le permite soportar una gran carga de forma eficiente. Es el gestor más usado en el mundo del software libre, debido a su rapidez y facilidad de uso.

OPEN SOURCE: Código Abierto, se denomina así a todo software que esta licenciado de tal manera que los usuarios pueden estudiar, modificar y mejorar su diseño mediante la disponibilidad del código fuente

OPENSSH: Es una versión libre de herramientas de conectividad SSH, que cifra todo tráfico eliminando efectivamente ciertos ataques, adicionalmente provee tunneling seguro y varios métodos de autenticación; soporta todas las versiones de SSH.

SERVIDOR BASE DE DATOS: Aplicaciones que permiten organizar datos en una o más tablas relacionadas de manera eficiente.

SERVIDOR PROXY: Es un equipo intermediario situado entre el ordenador del usuario e internet. Puede ser utilizado para registrar el uso de Internet y también para bloquear el a un sitio web

SERVIDOR WEB: Programa que implementa el protocolo HTTP para transferir páginas web, también se da este nombre al ordenador que ejecuta el programa.

SQUID: Popular programa de software libre que implementa un servidor proxy y un dominio para caché de páginas web.

VERIZON RISK: Es una división de la VERIZON dedicada a operaciones de inteligencia sobre riesgos en seguridad informática.

VERIZON: Es líder mundial en la prestación de comunicaciones, información y entretenimientos innovadores Fundada en el año 2000 por la corporación AT&T y posteriormente adquirida por la Corporación General Telephone & Electronics -GTE.

T A B L A D E A B R E V I A T U R A S

AIMC – Asociación para la Investigación de Medios de Comunicación.- Es una asociación de un grupo de empresas Españolas cuya actividad se fundamenta en torno a la comunicación, lleva a cabo estudios e investigaciones de diferentes naturalezas, siendo pionera en los estudios de audiencia en Internet.

CBC – Cipher-Block Chaining.- Es un modo de cifrado inventado en 1976, en el cual se relaciona cada bloque con todos los anteriores por medio de una operación XOR.

DCS – Diagrama de contexto del sistema.- Es un diagrama que establece el límite de información entre el sistema que se está implementando y el entorno en que va a operar.

DDNS – Dynamic DNS.- Es un método, protocolo o servicio de red que provee la capacidad de cambiar la configuración DNS, direcciones u otra información en tiempo real.

DNS – Domain Name System.- Es un sistema de nomenclatura jerárquica para ordenadores, servicios o cualquier recurso conectado a Internet o a una red privada.

FIPS – Federal Information Processing Standards.- Son estándares de acceso público desarrollados por el gobierno de los EEUU para la utilización por parte de todas las agencias del gobierno no militares.

FTP – File Transfer Protocol.- Es un protocolo de red para la transferencia de archivos entre sistemas conectados a una red TCP, basado en la arquitectura cliente servidor.

IP – Internet Protocol.- Es un protocolo no orientado a conexión usado tanto como el origen como el destino para la comunicación de datos a través de una red de paquetes conmutados, no fiable, de mejor entrega posible y sin garantías.

JDBC – Java Database Connectivity.- Permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java, independientemente del sistema operativo donde se ejecute o de la base de datos a la cual se accede, utilizando el dialecto SQL del modelo de base de datos que se utilice.

JRE – Java Runtime Environment.- Es un conjunto de utilidades que permite la ejecución de programas Java.

JSCH – Java Security Channel.- Es una librería que implementa el protocolo de comunicaciones seguras SSH.

PPPoE – Point to Point Protocol over Ethernet.- Es un protocolo de red para la encapsulación de PPP sobre una capa Ethernet, es utilizado mayoritariamente para proveer conexión de banda ancha mediante servicios de cable-modem y xDSL.

RSA – Rivest, Shamir y Adleman.- Es un algoritmo de clave pública cuya seguridad radica en el problema de factorización de números enteros fue desarrollado en 1977 por Ron Rivest, Adi Shamir y Len Adleman de allí su nombre.

SQL – Structured Query Language.- Es un lenguaje declarativo de acceso a base de datos relacionales que permite especificar diversos tipos de operaciones en estas.

SSH – Secure Shell.- Es un protocolo para acceso remoto seguro y otros servicios de red sobre una red insegura.

TCP – Transmission Control Protocol.- Es un protocolo de comunicación orientado a la conexión y fiable de la capa de transporte.

TLS – Transport Layer Security.- Es un protocolo criptográfico que proporciona una comunicación segura por una red.

VOIP – Voice over IP.- Es un grupo de recursos que hacen posible que la señal de voz viaje a través de internet empleando un protocolo IP.

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

I N T R O D U C C I Ó N

ARPANET, una red de comunicaciones, planificada, desarrollada y construida con fondos federales evolucionó notablemente convirtiéndose en una gran interconexión de redes privadas hoy denominada INTERNET; cada día experimenta una expansión abriendo las puertas a nuevos mercados [1]. Estos nuevos mercados representan la prestación de nuevos servicios. Gran parte de la información generada por la prestación de estos servicios se transmite sin la protección adecuada es así que con un simple análisis de tráfico esta podría ser accedida por un ente no autorizado. Debido a esto grandes organizaciones públicas y privadas, e incluso personas naturales, realizan considerables inversiones para proteger la información de cualquier amenaza potencial.

El control de acceso y el cifrado son herramientas esenciales para proteger la información. Ambas herramientas brindan excelentes prestaciones de seguridad en el primer caso con la identificación de usuarios autorizados y en el segundo con la codificación del contenido de la información. Estas junto con la tecnología de JAVA, serán herramientas esenciales para el desarrollo del presente proyecto.

CAPITULO 1

Antecedentes y justificación

La inseguridad informática es uno de los principales problemas de la actualidad, constantemente empresas y personas se ven afectadas; debido a ello surge la necesidad de crear soluciones que contrarresten dicho problema. En este primer capítulo se da una introducción y se justifica el presente proyecto como propuesta a la solución de este problema.

1.1. Descripción

El presente proyecto brinda una alternativa de seguridad a las comunicaciones entre distintos sistemas informáticos, el cual se fundamenta en técnicas como la autenticación y la criptografía para el cifrado de los datos. Adicional a esto, se han utilizado protocolos de seguridad avanzados junto con otras herramientas de seguridad de libre distribución.

La integración de los componentes antes mencionados crea un gran sistema de seguridad para el intercambio de los datos, basado en la arquitectura cliente servidor, permitiendo centralizar el control de los recursos y a la vez facilitando la tarea al usuario. Tomando en cuenta la arquitectura cliente servidor se han creado dos aplicaciones: la aplicación cliente y la aplicación servidor. Los detalles de seguridad para la comunicación de ambas aplicaciones fueron considerados, para lo cual se crearon dos túneles: uno para la gestión y los servicios y otro para el tráfico web.

La aplicación servidor es la más elaborada ya que se encarga de establecer la comunicación con el cliente, autenticar al usuario, realizar el intercambio de datos entre clientes y entre otras cosas más, las mismas que se desarrollaron minuciosamente teniendo en consideración múltiples detalles.

La aplicación cliente es la que interactúa directamente con el usuario debido a esto se consideraron detalles como la portabilidad, la facilidad de uso, la escalabilidad los mismos que son importantes al momento de elegir un software.

La aplicación cliente junto con la aplicación servidor son ofrecidos como software de seguridad al cual hemos denominado **Software de Tunneling**, los mismos que son complementarios**.**

La interacción entre cliente servidor y usuario cliente se da una vez que se encuentre en funcionamiento el servidor, luego será posible instalar la aplicación cliente en un ordenador y acceder a los servicios que el servidor brinde, entre ellos tenemos: creación de túnel SSH, y mensajería instantánea. Cabe destacar que el servidor tendrá ciertas características que permitirán que un usuario con la aplicación cliente pueda hacer uso de ella desde cualquier punto en el mundo que tenga acceso al Internet.

Por medio de la aplicación cliente se podrá realizar el registro de nuevos usuarios en la base de datos del servidor y haciendo uso de la aplicación de mensajería instantánea un usuario que se encuentre registrado podrá comunicarse con sus contactos de forma segura.

1.2. Objetivos

General

- Desarrollar un software para la transmisión segura de datos entre dos o más sistemas informáticos a fin de contrastar el mismo con los métodos actuales.

Específicos

- Demostrar que ciertos métodos de transmisión de datos en la actualidad son inseguros.
- Contrastar dos escenarios, uno en el que se tiene el control total de los recursos y otro en el que se tiene un control limitado.
- Comprobar la estabilidad de la arquitectura cliente/servidor para la centralización del control, los recursos y la integridad de los datos.
- Comprobar que con el uso del algoritmo asimétrico RSA se puede brindar seguridad y protección a los datos.

1.3. Justificación

Por medio de Internet se ofertan productos y servicios tales como servicios bancarios y compras en línea, entre otros. La penetración del Internet tanto en hogares como en las organizaciones ha causado que este mercado tenga un crecimiento exponencial. Es así que cada día se

ofertan nuevos productos y servicios; los mismos que sin la debida responsabilidad y protección de los datos que se generan y que viajan por las redes, se encuentran expuestos a accesos no autorizados y por tanto pueden causar un perjuicio al usuario. Esto es de gran preocupación para los usuarios, más aún cuando se conoce que existen ya muchos casos de crímenes informáticos y por ende que existen personas descubriendo y desarrollando nuevas formas de ataques.

La AIMC (Asociación para la Investigación de Medios de Comunicación) de España público en Febrero del 2011 un reporte [2] de una encuesta realizada a usuarios de Internet entre Octubre y Diciembre del 2010 que muestra que el 34.3% consideran un gran problema la inseguridad en Internet. De acuerdo a diario El Comercio de Ecuador en su artículo “Los delitos informáticos crecieron” [3] la Fiscalía ha registrado un incremento en los delitos informáticos, de 168 casos en 2009 a 638 casos entre Enero y Septiembre del 2010. La VERIZON RISK en colaboración con el Servicio Secreto de los Estados Unidos en un informe [4] publicado en el 2010 indica que el 40% de los ataques a usuarios de internet se realiza mediante Hacking haciendo uso de técnicas como MITM (Man in The Middle) o intermediario, fuerza bruta, entre otros.

Estadísticas como estas resaltan la importancia de proteger los datos que circula en la red, ya que de no tomar las debidas precauciones las pérdidas podrían ser cuantiosas. Debido a ello surge la necesidad de crear una solución como la que presenta este proyecto que aportará de forma significativa a la seguridad de la información, contrarrestando cualquier acto de violación respecto al contenido que circula en la red.

1.4. Metodología

La metodología propuesta abarca distintas fases, permitiendo realizar el trabajo de forma organizada como se muestra en la Figura 1.1 tornándose en un ciclo de desarrollo.

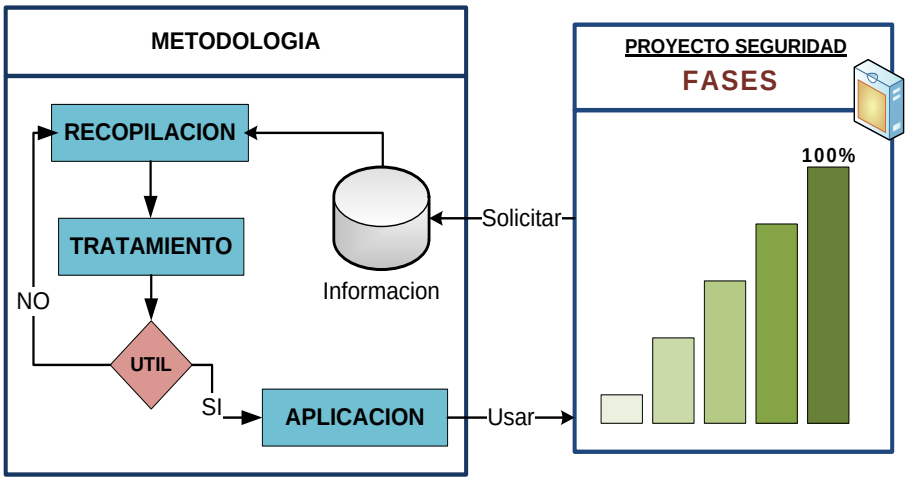


Figura 1.1 – Metodología del proyecto

Se basa en tres principios fundamentales que son: recopilación, tratamiento y aplicación de la información. Se detallará cada uno a continuación.

FASE DE RECOPILACIÓN.- Es la búsqueda de información útil que contribuya en la desarrollo del proyecto. Entre las herramientas y fuentes a consultar tenemos: Internet, libros, revistas, publicaciones, entre otros. Entre la información a recopilar podríamos mencionar: protocolos, algoritmos, tecnologías, códigos de programación, librerías, aplicaciones y demás.

FASE DE TRATAMIENTO.- Es uno de los principios más importantes y en el que se invierte gran cantidad de tiempo, consiste en examinar y escoger una solución de las distintas propuestas que podrían surgir en la fase de recopilación, haciendo un análisis minucioso de cada parte y seleccionando la información conveniente.

FASE DE APLICACIÓN.- Este principio permite iniciar, desarrollar y probar el proyecto, basado en la solución seleccionada en la etapa anterior. Se emplearán tecnologías, conceptos, códigos y componentes.

Aplicando la metodología descrita hemos alcanzado los objetivos propuestos, ya que esta nos ha permitido ingresar en un ciclo de mejora de la solución, pudiendo aun llegar a un nivel más complejo de la solución.

CAPITULO 2

Fundamento teórico

Este capítulo presenta los conceptos fundamentales aplicados durante la implementación del proyecto. Muestra conceptos básicos como: red informática, ordenador, servidor, enrutador, entre otros; así como conceptos más avanzados como: criptografía, protocolos y algoritmos de seguridad. Detalla también conceptos sobre JAVA, un lenguaje de programación utilizado en la implementación tanto de la aplicación cliente como servidor.

2.1 Red informática

Se define como red informática al conjunto de dispositivos de red tales como: ordenadores, enrutadores, conmutadores, servidores, entre otros que se interconectan de tal forma que pueden comunicarse, compartir recursos y prestar servicios [4]. A continuación se detallarán los componentes de una red informática, su clasificación y se hará una corta introducción al modelo de referencia TCP/IP en el que se basan tanto una pequeña red como Internet.

2.1.1 Componentes

Una red se encuentra formada por múltiples componentes sin embargo a continuación se mencionarán los de mayor relevancia para el proyecto.

ENRUTADOR.- es un dispositivo que permite la interconexión de ordenadores y dispositivos de red (Figura 11 del Anexo F), opera en la capa de red del modelo de referencia TCP/IP y permite el correcto direccionamiento de paquetes entre redes determinando la mejor ruta.

ORDENADOR.- es una máquina programable que responde a un sistema específico de instrucciones de una manera bien definida. Puede ejecutar una lista de instrucciones pregrabadas (programas) [5]. Parte del ordenador es el sistema operativo

que es el software principal que se ejecuta en el ordenador; el mismo se encarga de ejercer el control y coordinar el uso del hardware entre diferentes aplicaciones y los diferentes usuarios [6]. Existe una gran variedad de sistemas operativos y varias distribuciones o versiones de los mismos, entre los más relevantes tenemos: Windows, Mac OS y Linux. En el caso de Linux introduciremos UBUNTU el mismo que concentra su objetivo en la libertad de uso y la facilidad de uso e instalación. A diferencia de la mayoría de las distribuciones, que vienen con una enorme cantidad de software que puede o no ser de utilidad, la lista de paquetes de Ubuntu se ha reducido para incluir solo aplicaciones importantes y de alta calidad [7], es por ello que ha sido escogida como sistema operativo de nuestro servidor.

Se puede clasificar los ordenadores por tamaño y potencia en: personal, estación de trabajo, miniordenador, chasis y servidor. El servidor es un ordenador con determinadas características que presta uno o varios servicios, de acuerdo al servicio que presta un servidor puede ser de los siguientes tipos: aplicaciones, FTP (archivos), base de datos, correo electrónico, proxy, web, entre otros.

2.1.2 Modelo de referencia TCP/IP

Es un modelo de descripción de protocolos de red que describe la transmisión de datos entre dos ordenadores. Está formado por cuatro capas que son: Aplicación, Transporte, Internet y Acceso a la Red (Figura 15 del Anexo F). Provee conectividad punto a punto es así que especifica el formato, el direccionamiento y la forma de transmisión de los datos [8].

Toma su nombre por dos de los protocolos que lo integran, TCP (Protocolo de Control de Transferencia) e IP (Protocolo de Internet). Algunas de sus características son: sus protocolos son abiertos y gratuitos, independencia a nivel de software y hardware, esquema común de direccionamiento, protocolos estandarizados de alto nivel ampliamente disponibles y consistentes [9].

2.2 Seguridad informática

Maiwald en su libro “Fundamentos de seguridad de redes” define [11] la seguridad informática como: “*Medidas adoptadas para evitar el uso no autorizado, el mal uso, la modificación o la denegación de los sistemas informáticos”.* En base a la definición anterior podemos citar que la seguridad informática busca cumplir con los siguientes objetivos: brindar protección de los recursos tangibles e intangibles (Figura 7 del

Anexo F), restringir el acceso a entes no autorizados y mantener la confidencialidad, integridad, disponibilidad y responsabilidad de la información. Cumplir con estos objetivos debe ser primordial durante la implementación de cada sistema, metodología, procedimiento o política de seguridad.

La seguridad informática desempeña un papel preponderante en estos días, ya que interviene en operaciones tan simples como una sesión de mensajería instantánea hasta operaciones más complejas que comprenden transacciones y movimientos bancarios incluso en salvaguardar información sobre actividades del gobierno, cuestiones militares o diplomáticas.

La seguridad informática no es la tarea individual de una persona, un producto o un proceso en particular, se enfoca en cuatro aéreas principales: Física, Comunicaciones, Software y Procedimientos (Figura 10 del Anexo F), todas interactúan entre sí, ofreciendo un mecanismo de seguridad robusto, haciendo que cada parte este comprometida con la otra.

2.3 Ataques a la seguridad

La información que se transmite a través de la red podría sufrir cambios o alteraciones respecto a su contenido o podría ser accedida por

alguien no autorizado, perjudicando significativamente a la entidad propietaria de esta. Lo descrito anteriormente podría deberse a un ataque, existen diversas razones por las que alguien es motivado a realizar un ataque entre las más comunes tenemos: diversión o desafío, venganza, terrorismo, obtener un beneficio económico, ventaja competitiva y obtener control o poder. Para la seguridad informática existen cuatro tipos principales de ataque: Acceso, Modificación, Denegación del Servicio y Refutación. Algunos métodos dentro de cada tipo han sido utilizados durante la fase de pruebas por ello se detalla a continuación cada uno.

ATAQUE DE ACCESO.- Consiste en obtener acceso a recursos a los que no se está autorizado acceder, un ejemplo de este tipo de ataque es la escucha furtiva; se lleva acabo generalmente en redes de acceso inalámbrico (Figura 8 del Anexo F), donde el atacante captura el tráfico que pasa por el enrutador y hace uso de la información según su conveniencia. La exploración de vulnerabilidades y la intercepción (Figura 9 del Anexo F) son ataques más complejos, pero muy utilizados.

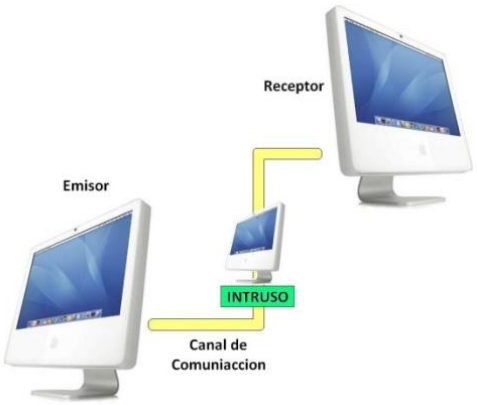


Figura 2.1 – Ataque Hombre en el Medio (Man in The Middle)

ATAQUE DE MODIFICACIÓN.- Se enfoca en el intento de cambiar o modificar la información. Este ataque se vuelve extremadamente serio cuando un atacante adquiere privilegios de administrador provocando inserción y/o eliminación [11].

ATAQUE DE DENEGACIÓN DE SERVICIOS.- Se enfoca en que un usuario legítimo de la red quede sin acceso a los recursos. Cabe mencionar que este tipo de ataque podría ocurrir de forma accidental por personas que manipulan servicios, programas o sistemas sin tener conocimiento del mismo [11].

ATAQUE DE REFUTACIÓN O FALSIFICACIÓN.- Es un ataque dirigido a la responsabilidad de información, consiste en suministrar información falsa o fraudulenta dentro de un evento o transacción real. Esto puede ser llevado a cabo alterando documentos de textos, alterando firmas o

suplantando la identidad de un usuario u ordenador [11]. Otro ejemplo de esto es el ataque Hombre en el medio (Man in the Middle), consiste en que el atacante se coloca en medio de la comunicación haciéndose pasar por ambos sin ser detectado ver Figura 2.1.

2.4 Criptografía

Es una técnica, o más bien un conjunto de técnicas, que originalmente tratan sobre la protección o el ocultamiento de la información frente a observadores no autorizados. La criptografía se divide en dos grupos, los cuales se describen a continuación.

SIMÉTRICOS.- la criptografía simétrica o criptografía de clave privada consiste en cifrar y descifrar la información utilizando una misma clave secreta (Figura 17 del Anexo F). Antes de transmitir o recibir información ambas partes deberán negociar la clave a utilizar, de tal manera que solo si se conoce la clave se podrá cifrar o descifrar la información. La criptografía simétrica es rápida y fácil de implementar en hardware y software.

ASIMÉTRICOS.- la criptografía asimétrica o criptografía de llave pública consiste en cifrar y descifrar la información utilizando un par de llaves (Figura 18 del Anexo F). El par de claves denominadas llave pública y llave privada son utilizadas para cifrar y descifrar la información

respectivamente. La clave privada es mantenida en secreto es utilizada para descifrar la información. La clave pública, divulgada junto con la información, es utilizada para cifrar la información. Una de las ventajas de la criptografía asimétrica es su robustez y que no se puede obtener la clave privada a partir de la pública. RSA, publicado por Rivest–Shamir – Adleman en 1978 es uno de los algoritmos criptográficos de llave púbica más utilizado por su gran nivel de robustez y seguridad. En el ANEXO E se especifica el proceso del algoritmo.

2.5 Protocolos seguros

Un protocolo es un conjunto de normas que en el caso de la informática especifican como se transmite información, como se codifica o como se provee seguridad [11]. Los protocolos pueden ser implementados por hardware, software, o una combinación de ambos, mencionaremos a continuación algunos relevantes al proyecto.

SSH.- Secure Shell, Intérprete de órdenes segura; es un protocolo que permite realizar acceso remoto y otros servicios sobre una red insegura. Está formado por tres componentes: Protocolo de capa de Transporte, Protocolo de Autenticación de Usuario y Protocolo de Conexión (Figura F.3) [12]. En el ANEXO E se muestra en mayor detalle cada capa del protocolo.

TLS.- Transport Layer Security, Seguridad en la Capa de Transporte; es un protocolo que provee comunicaciones seguras sobre Internet, está diseñado para prevenir el espionaje, manipulación o falsificación de los mensajes [15]. Su objetivo principal es proporcionar privacidad e integridad a los datos entre dos aplicaciones que se comunican. La versión 1.1 es la última versión admitida, diseñada de forma modular para que pueda soportar versiones anteriores y posteriores a esta.

2.6 Java

Es un lenguaje de programación orientado a objetos, desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria. La mayoría de las tecnologías Java tienen licencia GNU GPL versión 2, de tal forma que prácticamente todo el Java de Sun es software libre (aunque la biblioteca de clases de Sun que se requiere para ejecutar los programas Java aún no lo es).

El lenguaje Java se creó con cinco objetivos principales que son: programación orientada a objetos, permitir la ejecución de un mismo programa en múltiples sistemas operativos (portabilidad), soporte para

trabajo en red, permitir ejecutar código en sistemas remotos de forma segura y facilidad de uso.

Dentro del funcionamiento de JAVA existen tres conceptos a diferenciar y comprender: JRE, JDK y API's que se describirán a continuación. El JRE (Java Runtime Environment) o entorno en tiempo de ejecución de Java es el software necesario instalado en el sistema operativo para ejecutar cualquier aplicación desarrollada con Java. El JDK (Java Development Kit) o equipo de desarrollo de Java es un software que provee herramientas de desarrollo para la creación de programas en java. Finalmente las API's (Application Programming Interface) o interface de programación de aplicaciones es el conjunto de clases utilizado para efectuar toda clase de tareas dentro de una aplicación. Las clases en las API's de Java se organizan en grupos disjuntos llamados paquetes, cada paquete contiene un conjunto de interfaces, clases y excepciones relacionadas.

2.6.1 Seguridad en java

El lenguaje Java hace especial hincapié en la seguridad, esto incluye: seguridad del lenguaje, criptografía, infraestructura de clave pública, autenticación, comunicación segura, y control de acceso. Debido a las restricciones de exportación de software criptográfico Java descompuso sus API's o Librerías

criptográficas en dos: JCA (Java Cryptography Architecture) y JCE (Java Cryptography Extension).

JCA forma parte de Java a partir de la v1.2, bajo el paquete java.security, dispone de elementos de seguridad que no están sujetos a restricciones de exportación. JCE forma parte de Java a partir del JDK v1.4, se incluyen clases de cifrado y descifrado de mensajes y sus clases se encuentran bajo el nombre javax.crypto. Ambas Librerías se dividen en: clases y proveedores de seguridad. Las clases actúan como interfaces en las que se definen una serie de operaciones y los proveedores suministran implementaciones concretas de los algoritmos criptográficos anunciados. Cada instalación JDK cuenta con uno o más proveedores instalados y configurados de forma predeterminada, entre los más comunes se encuentran los que se muestran en la Tabla 9 del Anexo E, proveedores adicionales pueden ser agregados. Cuando un servicio de seguridad se solicita, el proveedor de la más alta prioridad que implementa este servicio es seleccionado, en la Figura 6 del Anexo A se muestra un detalle de la interacción entre proveedores y aplicación.

CAPITULO 3

Diseño

En base a los objetivos del proyecto se desarrolló el diseño, ya que éste nos permite tener una visión amplia y detallada de cada elemento del proyecto incluyendo su funcionamiento, lo cual nos permite tener una guía legible y comprensible para su correcta implementación

3.1 Arquitectura del sistema

El modelo arquitectónico para la realización del proyecto fue el de **cliente-servidor**, el cual nos permite centralizar la comunicación y administración en el servidor así como facilita un entorno escalable de clientes. Además, este modelo es el más utilizado y el que mejor ha evolucionado en los últimos años. En nuestro sistema hemos definido a los elementos en base a dicho modelo y las labores que realizará cada uno son: El CLIENTE es un ordenador donde un usuario se autentica en el sistema e inicia un requerimiento de servicio mientras el SERVIDOR es un ordenador central encargado de atender a múltiples clientes, es quien controla la comunicación de forma concurrente por medio de circuitos virtuales prestando determinados servicios. La figura 3.1 muestra la arquitectura del sistema con dos clientes y el servidor.

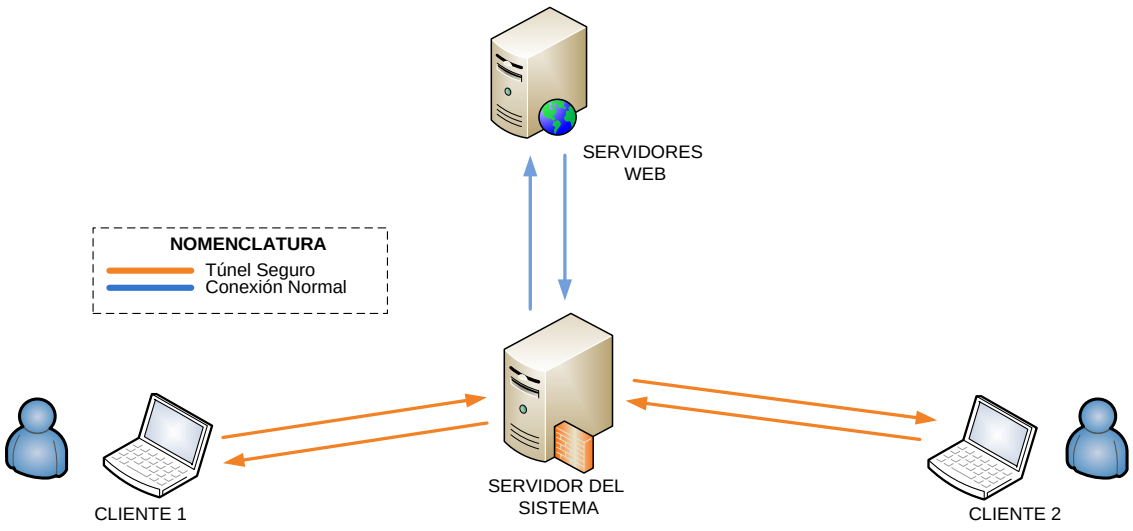


Figura 3.1 – Arquitectura del sistema

3.2 Diseño del funcionamiento

Como se puede observar la figura 3.2 describe de forma secuencial el funcionamiento del sistema, mostrando la interacción entre el usuario, el cliente y el servidor, básicamente lo que ocurre es lo siguiente:

1. El usuario ejecuta la aplicación cliente en el ordenador cliente.
2. La aplicación cliente solicita la conexión a la aplicación servidor y se estable la misma.
3. Se crean los túneles entre aplicación cliente y servidor para la transferencia segura de datos.
4. El usuario puede autenticarse de forma segura con el servidor.
5. Una vez autenticado el usuario puede iniciar la transferencia de datos de forma segura hacia el servidor y viceversa, podrá además configurar su explorador para poder navegar de forma anónima.
6. La transferencia segura continuará hasta que el usuario decida cerrar la aplicación en cuyo caso se solicita a la aplicación servidor el cierre de los túneles y la posterior desconexión.

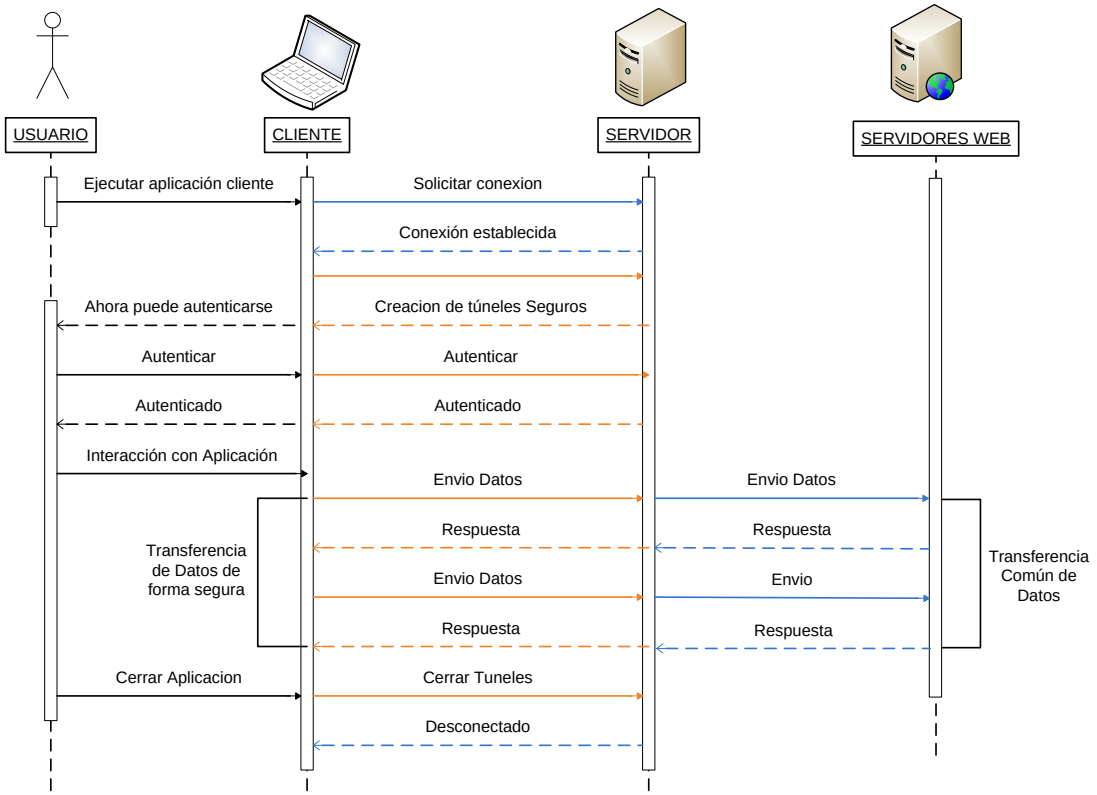


Figura 3.2 – Diseño del Funcionamiento

3.3 Diagrama de contexto del sistema global

El Software de Tunneling que comprende la aplicación cliente y la aplicación servidor será tomado como un **sistema** (Figura 3.3), el cual cifra y descifra los datos manteniendo la confiablidad e integridad de los datos, brindando una comunicación segura.

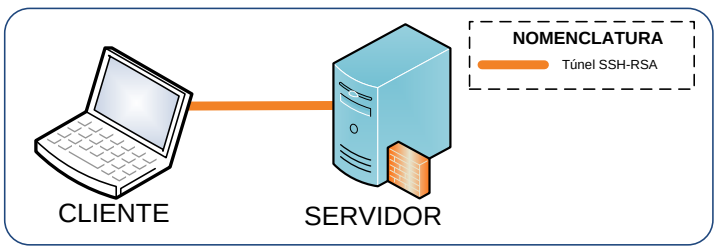


Figura 3.3 – Sistema – Software de Tunneling

A fin de establecer el límite de información entre el sistema que se implementará y el entorno en que va a operar se muestra en la figura 3.4 el diagrama de contexto del sistema (DCS) [14]. El usuario va a interactuar con el ordenador donde se ejecuta la aplicación cliente por medio de los periféricos de entrada (teclado, mouse), enviando datos hacia el servidor a fin de comunicarse con otros usuarios del sistema y podrá tener una respuesta del mismo de forma gráfica por medio de los periféricos de salida (monitor).

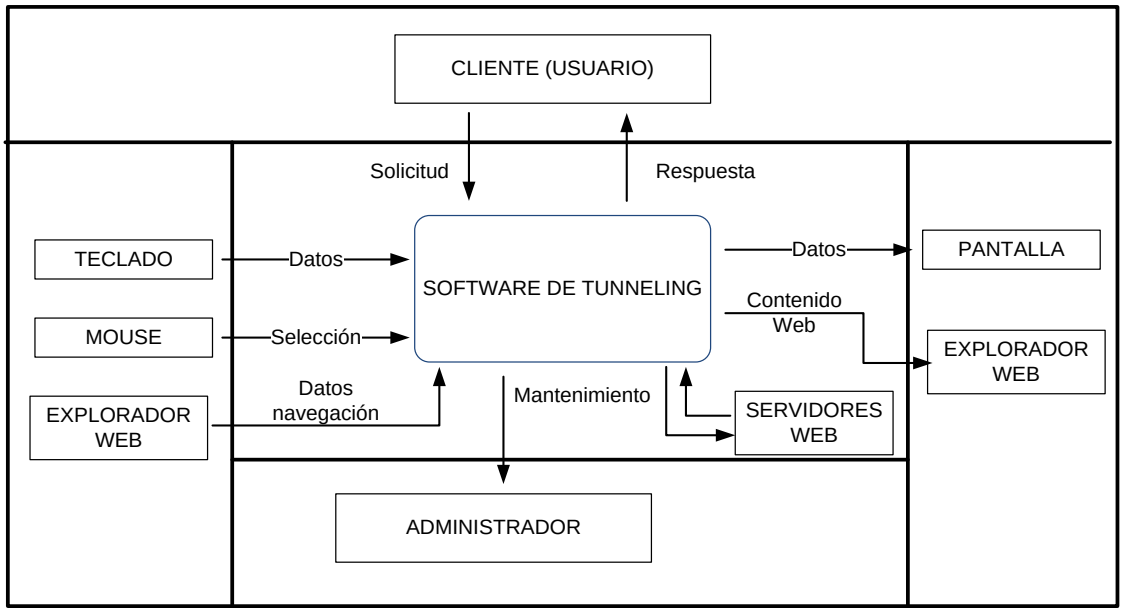


Figura 3.4 – Diagrama de contexto del sistema Global

Se han definido las siguientes entidades externas dentro del sistema:

Cliente (Usuario): Es quien intercambiará datos por medio del sistema con otros usuarios y con la web. El cuadrado superior del diagrama de contexto (Figura 3.4) modela al cliente o usuario.

Teclado/Ratón: Periféricos de entrada que permitirán la interacción del Cliente con el sistema para el ingreso de datos y selección de opciones. El cuadrado de la izquierda del diagrama de contexto (Figura 3.4) modela al periférico de entrada.

Explorador Web: El cliente por medio de los periféricos de entrada interactúa con el explorador web a fin de navegar en el Internet, estos datos de navegación viajarán de forma segura por el sistema, luego del sistema hacia los servidores web, luego de retorno al sistema y finalmente al explorador web. El cuadrado de la izquierda del diagrama de contexto (Figura 3.4) modela al explorador web.

Pantalla: Mostrará gráficamente los datos intercambiados con otros usuarios. El cuadrado de la derecha del diagrama de contexto (Figura 3.4) modela al dispositivo de salida.

3.4 Diagrama de contexto del sistema detallado

La figura 3.5 muestra los principales subsistemas y el flujo de información entre ellos, como se puede observar el sistema está conformado por una aplicación cliente y una aplicación servidor y cada una de ellas está conformada por varios subsistemas

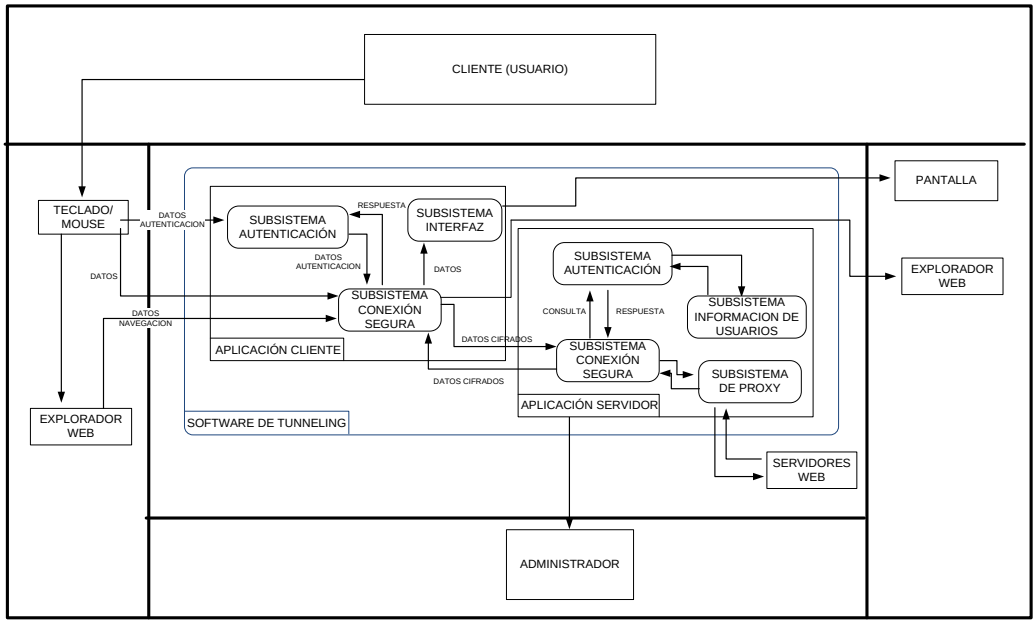


Figura 3.5 – Diagrama de contexto del sistema detallado

La aplicación cliente se encuentra dividido en 3 subsistemas los cuales se describen a continuación:

Subsistema de Autenticación.- Se encarga del proceso de autenticación, recibe los datos del usuario y realiza la consulta al servidor, permitiendo o restringiendo el acceso al sistema.

Subsistema de Conexión Segura.- Recibe datos ingresados por el usuario y asegura que estos datos viajen de forma segura hacia el servidor y viceversa, muestra los datos al usuario por medio del subsistema de interfaz y el explorador web.

Subsistema de Interfaz.- Muestra una interfaz gráfica para la interacción con el usuario, es decir permite el ingreso de datos al sistema y muestra los datos de salida; así como permite la configuración de ciertas opciones.

La aplicación servidor se encuentra dividido en 4 subsistemas los cuales se describen a continuación:

Subsistema de Autenticación.- Recibe los datos de autenticación del usuario enviados desde la aplicación cliente para ser verificados por medio del subsistema de información de los usuarios, finalmente envía un mensaje de respuesta dependiendo si el usuario se encuentra o no registrado en la base de datos.

Subsistema de Conexión Segura.- Recibe los datos enviados por un usuario, descifra las cabeceras y re direcciona los datos al usuario correspondiente cifrando una vez más los mismos o hacia el subsistema de proxy para la posterior comunicación con los servidores web.

Subsistema de Información de Usuarios.- Contiene la información de los todos los usuarios en la base de datos la misma que se encuentra

almacenada en el servidor, permite también realizar consultas a la misma.

Subsistema de Proxy.- Recibe los datos de navegación de los usuarios por medio del subsistema de cifrado/descifrado y re direcciona este tráfico hacia los servidores web, finalmente envía el tráfico que recibe de los servidores web hacia el subsistema de cifrado/descifrado para su posterior entrega al usuario correspondiente.

3.5 Consideraciones de seguridad

Un sistema brinda protección a los datos si la comunicación entre sus subsistemas se realiza de forma segura, la figura 3.6 muestra en mayor detalle la interacción entre las que serían las diferentes partes que conformarán el sistema. En la figura podemos observar dos escenarios, uno en el que tenemos control total de los recursos que sería la comunicación cliente-cliente y otro en el que tenemos un control parcial de los recursos que sería la comunicación para acceder a contenido de tipo web.

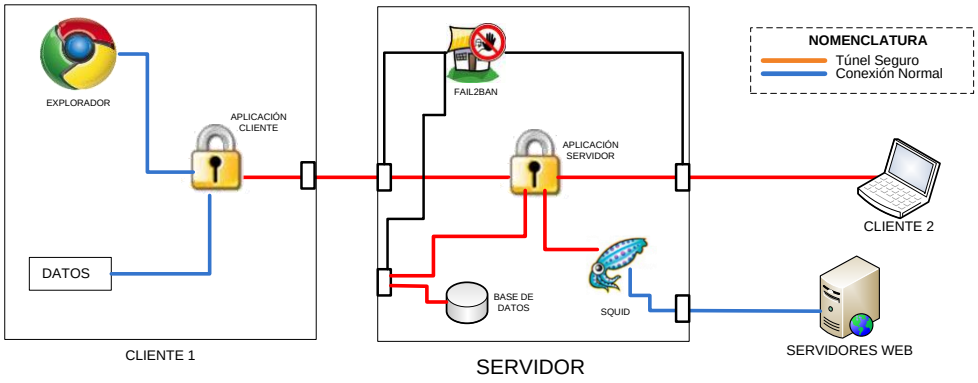


Figura 3.6 – Escenarios de seguridad del Sistema

Para el primer escenario podemos brindar una transmisión segura de los datos debido al control total de los recursos, mientras que para el segundo escenario se puede brindar una transmisión segura de los datos hasta el servidor del sistema obteniendo seguridad parcial y navegación anónima.

Con el esquema mencionado podemos alcanzar las consideraciones de seguridad que son la base sobre la cual construimos nuestro proyecto.

C A P I T U L O 4

I m p l e m e n t a c i ó n

Para la implementación del proyecto se utilizaron los recursos de software y hardware que se mencionarán en este capítulo, además se mencionarán los requerimientos mínimos. Se utilizó software con licencia GNU GLP como UBUNTU-SQUID y se implementó una aplicación de escritorio basada en Java para el servidor así como una para el cliente.

.

4.1 Recursos hardware

El sistema cuenta básicamente con los siguientes recursos de hardware: enrutador, servidor y ordenador; se detallará a continuación su función, los requerimientos mínimos y las características de los componentes utilizados durante la implementación y pruebas.

ENRUTADOR.- permite la conexión de la LAN con Internet. Los requerimientos mínimos son los siguientes: un puerto LAN RJ45 10/100M, un puerto WAN RJ45 10/100M, soporte IEEE 802.3u, soporte PPPoE, asignación dinámica y estática de direcciones de red, soporte de servidores virtuales y DNS dinámico.

ORDENADOR SERVIDOR.- los requerimientos mínimos para el servidor son: 256MB de memoria RAM, disco duro de 5GB y procesador pentium4 o superior. Para el proyecto se ha utilizado un ordenador con las siguientes características: procesador Intel Pentium4 de 2,8 GHz, memoria RAM de 1GB y disco duro de 16GB.

ORDENADOR CLIENTE.- la aplicación de escritorio desarrollada en lenguaje JAVA para el cliente no requiere un alto nivel de procesamiento por lo que se considera que se ejecutará de forma eficiente en un ordenador con procesador Pentium 4 o mayor y al menos 256MB de RAM; no se consideran versiones anteriores pues son ya obsoletas.

4.2 Recursos software

Tanto para el ordenador cliente como para el servidor se deben cumplir con ciertos requerimientos de software los mismos se indican a continuación. En el ordenador Servidor se debe instalar: Ubuntu 11.04 como sistema operativo, MySQL v5.1 como servidor de base de datos, MySQL Administrator como administrador de MySQL, SQUID como servidor proxy, OPENSSH como servidor SSH y JRE 1.6; mientras que en el ordenador Cliente se debe instalar JRE 1.6 en cualquiera que sea el sistema operativo huésped, esto gracias a la portabilidad que brinda Java.

4.3 Desarrollo en java

En la actualidad, existen diversos lenguajes de desarrollo que aprovechan la arquitectura cliente-servidor, uno de ellos es JAVA; para la implementación en Java se utilizo el entorno de desarrollo Netbeans (El Anexo A) y las librerías adicionales JSCH, JDBC, CRYPTO Y SECURITY (Anexo F). La aplicación cliente ha sido desarrollada de forma modular es así que se ha tratado por separado: parte lógica, seguridad e interfaz gráfica (Figura 19 del Anexo F) pudiendo hacerse mejoras en cada módulo de ser necesario.

4.4 Funcionalidades

La funcionalidad de un sistema es lo que el mismo puede hacer, es así que podemos resaltar dos funcionalidades de nuestro sistema: Tunneling y mensajería instantánea; se puede hacer uso de ellas una vez ejecutada la aplicación cliente, iniciado sesión y realizadas algunas configuraciones.

TUNNELING.- se crea un túnel por medio del protocolo SSH entre el cliente y el servidor; es así que al configurar en el explorador del ordenador cliente este túnel como proxy (Anexo B), todo el tráfico que se genere por la navegación en Internet viajará de forma segura hacia el servidor; brindando anonimato y privacidad (Figura 4.1). Cabe indicar que el túnel creado por el software puede ser configurado como proxy en otras aplicaciones que lo permitan, por ejemplo Google Talk.

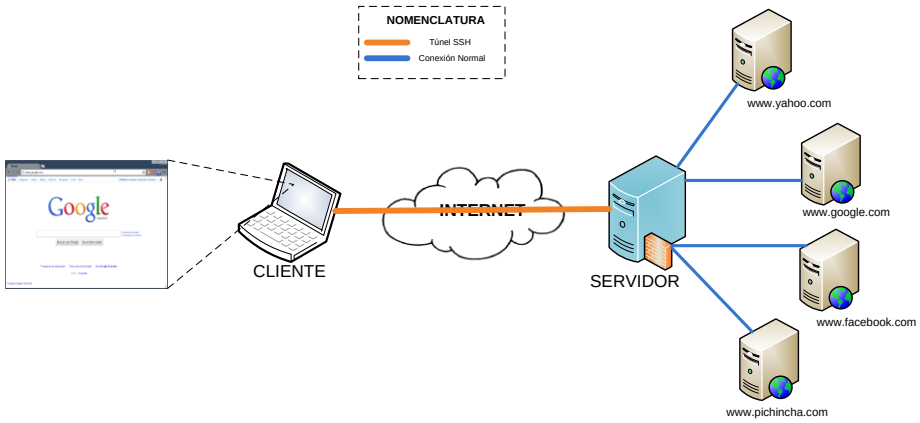


Figura 4.1 – Funcionalidad de Tunneling

MENSAJERÍA INSTANTÁNEA.- por medio de esta funcionalidad se permite a N clientes conectarse de forma segura e intercambiar información (Figura 4.2), la transferencia segura de información es posible mediante la creación de un túnel entre cada cliente y el servidor empleando el algoritmo RSA. Se determinará el número de clientes posibles durante la etapa de pruebas.

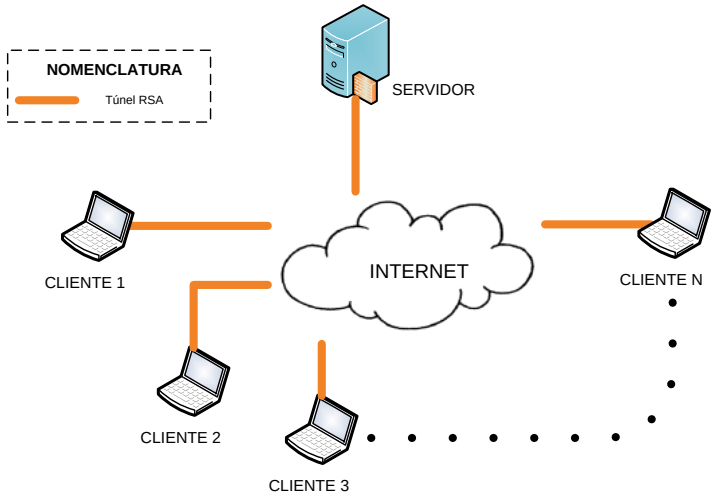


Figura 4.2 – Funcionalidad de mensajería instantánea

CAPITULO 5

Pruebas y Resultados

Una vez cumplido con cada detalle del diseño y la implementación, el presente capitulo procederá a mostrar cada una de las pruebas a la que ha sido expuesta el software, entre las pruebas tenemos: portabilidad, capacidad de usuario, velocidad, ataques de fuerza bruta, escucha furtiva e intermediario. Teniendo como objetivo exponer el buen nivel de seguridad que nuestra aplicación ofrece.

5.1 Equipos utilizados

Se han utilizado varios dispositivos durante el proceso de pruebas, la descripción de los dispositivos se encuentra en la tabla 13 del anexo E. Cabe mencionar que se ha actualizado o instalado de acuerdo al caso en cada ordenador la versión 1.6 del JRE y en algunos casos se ha configurado ciertos permisos para la correcta ejecución de la aplicación que hemos desarrollado. La tabla 14 del anexo E muestra las características del enrutador empleado.

5.2 Prueba 1: Portabilidad

Para la realización de esta prueba, se han utilizado los ordenadores 1, 2, 3, 4 y 8 (tabla E.13) así como el enrutador (tabla E.14), los cuales se interconectaron en un ambiente de LAN como se muestra en la figura 5.1.

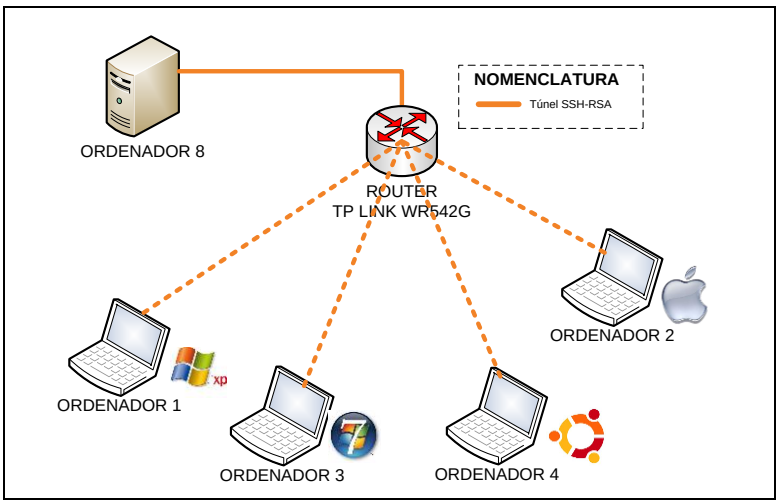


Figura 5.1 – Prueba 1: Prueba de portabilidad

OBJETIVOS

- Verificar la portabilidad de la aplicación por medio de la ejecución de la misma en diferentes sistemas operativos.
- Comprobar que hay comunicación entre los clientes utilizando la funcionalidad de mensajería instantánea.

PROCEDIMIENTO

1. Se ejecuta la aplicación servidor en el ordenador 8 y se activan los servicios que brindará.
2. En los otros ordenadores (1, 2, 3 y 4) se ejecuta la aplicación cliente.
3. Se verifica que no haya errores de ejecución en cada ordenador cliente, por medio de la consola del servidor.
4. Se autentica un usuario diferente en cada ordenador, los cuales se añadieron previamente a la base de datos del servidor.
5. Se procede a comunicarse entre los usuarios por medio de mensajería instantánea.

RESULTADOS

- Se tuvo éxito en la comunicación entre los usuarios, se muestran capturas de pantalla de cada sistema operativo en las figuras 1, 2, 3 y 4 del anexo F; en las mismas se puede observar el

intercambio de mensajes entre cada cliente por medio del servicio de mensajería instantánea.

- Se verifico por tanto la portabilidad de la aplicación al ejecutarla en los sistemas operativos: Mac, Linux-Ubuntu, Windows XP, Windows 7, obteniéndose excelente desempeño de la aplicación en cada sistema operativo, como se muestra en las figuras 1, 2, 3 y 4 del anexo G.

5.3 Prueba 2: Prueba de Capacidad

Para la realización de esta prueba, se han utilizado los ordenadores 1, 5 y 8 (tabla E.13) así como el enrutador (tabla E.14), los cuales se interconectaron en un ambiente de LAN como se muestra en la figura 5.2.

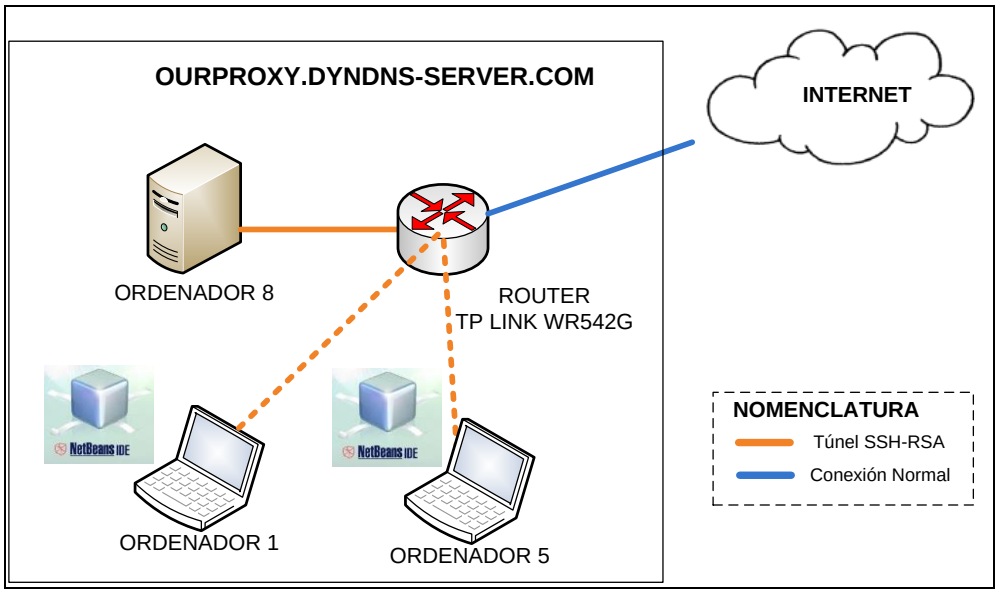


Figura 5.2 – Prueba 2: Prueba de Capacidad

OBJETIVOS

- Comprobar el número máximo de conexiones simultáneas cliente que el servidor soporta, simulando la conexión progresiva de 1000 clientes al menos por dos ocasiones.
- Observar el comportamiento del servidor durante la simulación de conexión de los clientes.

PROCEDIMIENTO

1. Se ejecuta la aplicación servidor en el ordenador 8 y se activan los servicios que brindará.
2. Se modifica el código fuente de la aplicación cliente añadiendo una rutina que simule la conexión y autenticación con el servidor tal como si fueran cien clientes, al hacer esto se crean ambos túneles SSH y RSA para cada cliente.
3. Se ejecuta cinco veces la aplicación cliente con ayuda del IDE Netbeans en los ordenadores 1 y 5, esto permitirá simular la ejecución y conexión de mil de clientes con el servidor.
4. Se verifica que no haya errores de ejecución durante la conexión y autenticación de cada cliente en cada ordenador, mediante la consola del servidor.

5. Configurar el proxy del explorador con el puerto de una de las conexiones al servidor creadas en el paso 3, los puertos van desde el 9090 al 9590 en cada ordenador.
6. Navegar con el explorador y observar el comportamiento del servidor.

RESULTADOS

- El número máximo de conexiones soportadas por el servidor es de 976 clientes simultáneos como se muestra en la figura 9 del anexo G y de 999 para la segunda prueba (Figura G.12), siendo esta una medida relativa, debido ya que sólo por dos de estas conexiones clientes se envió tráfico web como se indicó en el paso 6 del procedimiento.
- Se observó que el rendimiento del servidor disminuyo a medida que se realizaron nuevas conexiones de clientes como lo muestra la figura 10 del anexo G.
- Se observó también que el rendimiento de los ordenadores clientes no se vieron afectados durante el proceso de prueba.

5.4 Prueba 3: Velocidad LAN

Para la realización de esta prueba, se han utilizado los ordenadores 1 y 8 (tabla E.13) así como el enrutador (tabla E.14), los cuales se interconectaron en un ambiente de LAN como se muestra en la figura

5.3. Ambos ordenadores se encuentran en el dominio ourproxy.dyndns-server.com.

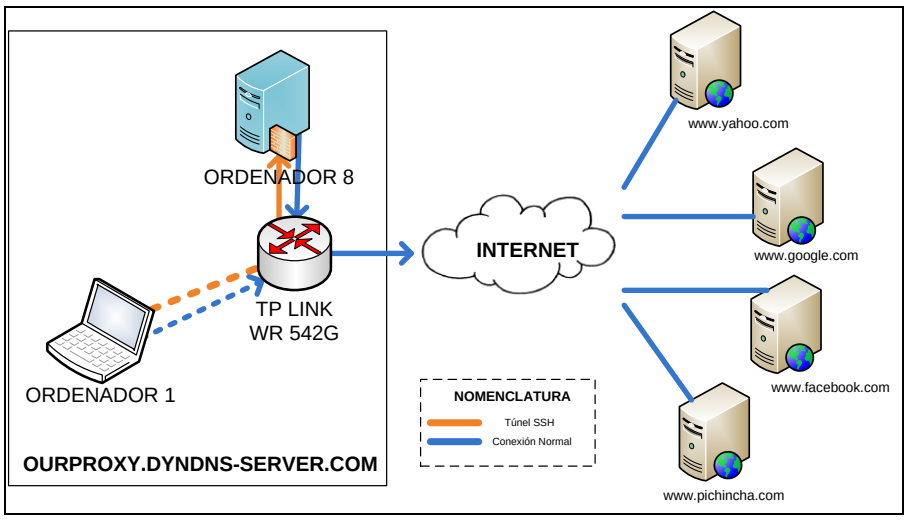


Figura 5.3 – Prueba de velocidad en ambiente LAN

OBJETIVOS

- Medir las velocidades de navegación, carga y descarga con y sin la utilización de la aplicación cliente.
- Comparar las velocidades anteriormente encontradas y contrastar los resultados.

PROCEDIMIENTO

Se utilizará una aplicación alojada en las páginas web http://www.speedtest.net/ (Figura G.5) y http://www.internautas.org/ (Figura F.23) para realizar las mediciones, estas mediciones fueron: tiempo de ping, velocidad de carga y descarga. La figura 5.3 en azul

muestra la conexión directa a Internet que será considerado como *escenario 1*, mientras en naranja la conexión mediante la aplicación cliente considerado *escenario 2*. Se procederá de la siguiente forma:

1. Se realiza las mediciones en el *escenario 1*, se toman 10 muestras con (speedtest) y luego 10 con (internautas) para obtener posteriormente una media en cada caso.
2. Se ejecuta la aplicación servidor en el ordenador 8.
3. Se ejecuta la aplicación cliente en el ordenador 1.
4. Se configura el proxy en el explorador del ordenador 1.
5. Se realizan las mediciones en el *escenario 2*, se toman 10 muestras.

RESULTADOS

- Se han realizado mediciones de velocidad tanto para el *escenario 1* como para el *escenario 2* como lo indica el procedimiento, las tablas 9 y 10 del Anexo E resumen las mediciones tomadas y la media de cada una para el medidor (speedtest), mientras las tablas 15 y 16 del mismo anexo resumen las mediciones tomadas con el medidor (internautas).
- De acuerdo a las mediciones mostradas en las tablas 9 y 10 del anexo E y como se aprecia en las Fig. 5.4 y 5.5 la velocidad de descarga en promedio aumenta con la utilización del proxy

mientras que la de carga se mantiene, el mismo comportamiento se puede observar con el otro medidor.

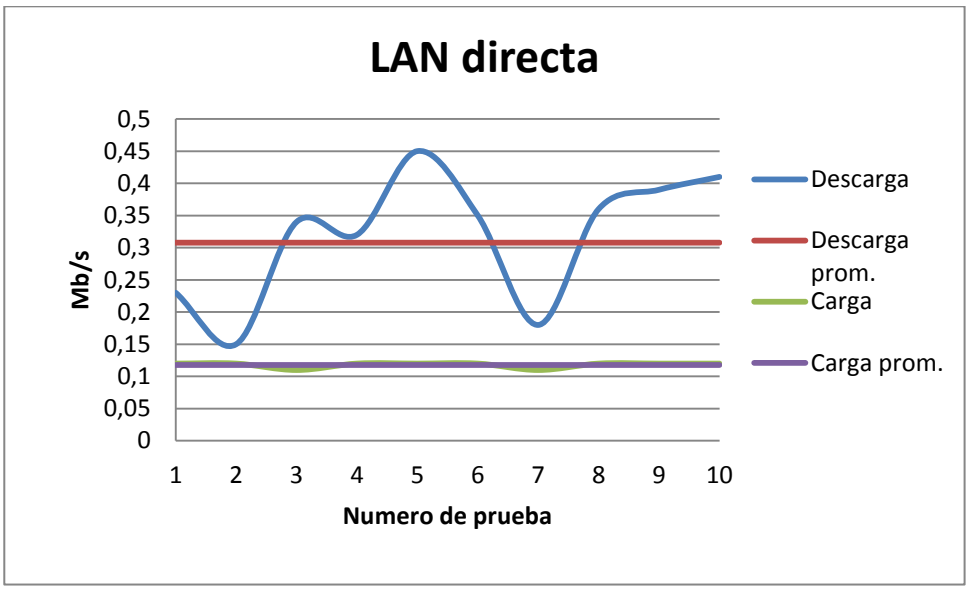


Figura 5.4 – Prueba de velocidad LAN con conexión directa.

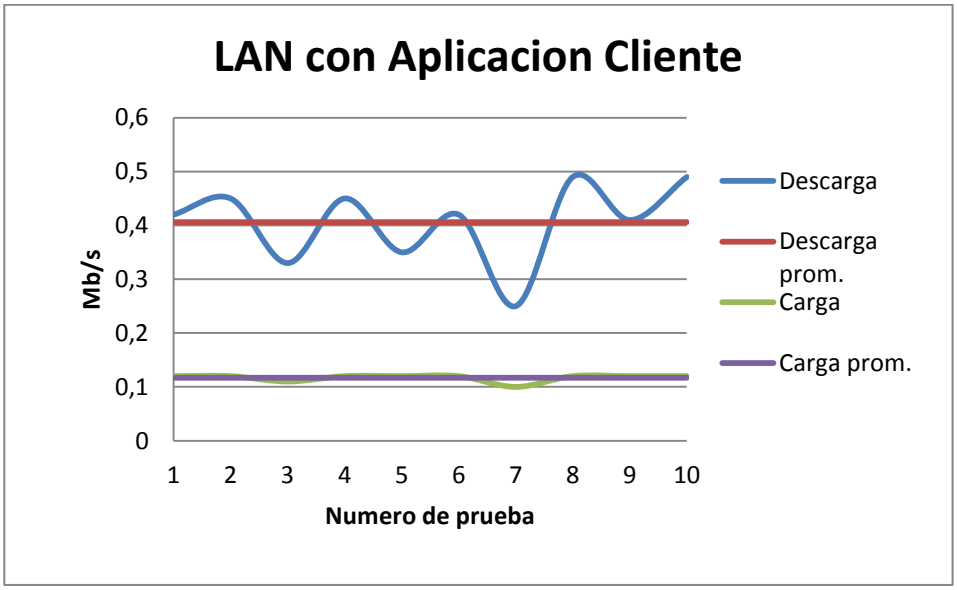


Figura 5.5 – Prueba de velocidad LAN con aplicación cliente.

5.5 Prueba 4: Velocidad WAN

Para la realización de esta prueba, se han utilizado los ordenadores 1 y 8 (tabla E.13) así como el enrutador (tabla E.14), los cuales se interconectaron en un ambiente de WAN como se muestra en la figura 5.6, ambos ordenadores se encuentran en dominios diferentes.

OBJETIVOS

- Medir las velocidades de navegación, carga y descarga con y sin la utilización de la aplicación cliente.
- Comparar las velocidades anteriormente encontradas y contrastar los resultados.

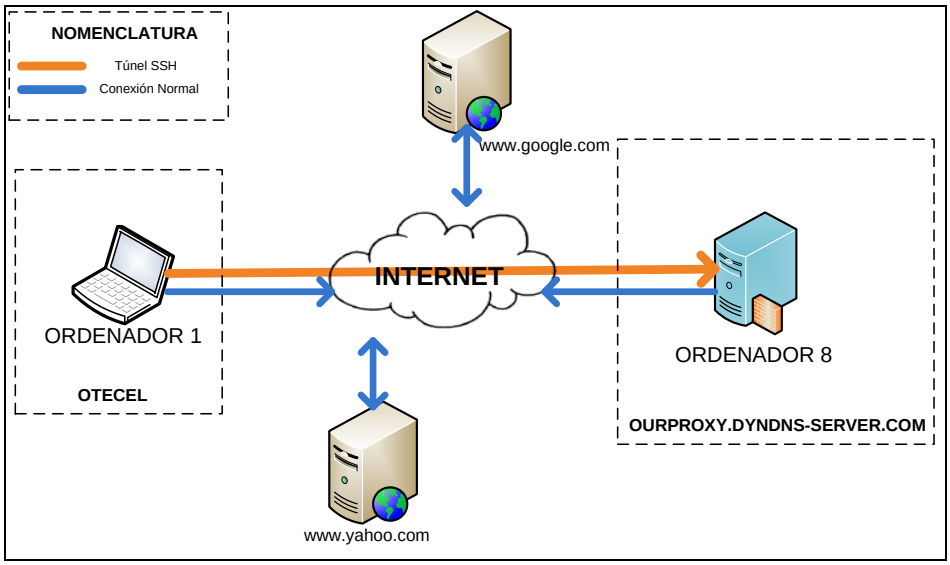


Figura 5.6 – Prueba de Velocidad en ambiente WAN

PROCEDIMIENTO

Se utilizará una aplicación alojada en la página web http://www.speedtest.net/ (Figura G.5) y http://www.internautas.org/ (Figura F.23); para realizar las mediciones, estas mediciones son tiempo de ping, velocidad de carga y descarga. La figura 5.6 en azul muestra la conexión directa a Internet considerado el *escenario 1*, mientras que en naranja se muestra la conexión mediante la aplicación cliente considerado *escenario 2*.

1. Se realiza las mediciones en el *escenario 1*, se toman 10 muestras para obtener una media.
2. Se ejecuta la aplicación servidor en el ordenador 8.
3. Se ejecuta la aplicación cliente en el ordenador 1.
4. Se configura el proxy en el explorador.
5. Se realizan las mediciones en el *escenario 2*, se toman 10 muestras.

RESULTADOS

- Se han realizado mediciones de velocidad tanto para el *escenario 1* como para el *escenario 2*, las tablas 11 y 12 del anexo E resumen las mediciones tomadas y la media de cada una.
- De acuerdo a las mediciones mostradas en las tablas 11 y 12 del anexo E y como lo muestran las Fig. 5.7 y 5.8 la velocidad de

descarga en promedio disminuye con la utilización de la aplicación cliente un 76%; mientras que la de carga disminuye un 78%.

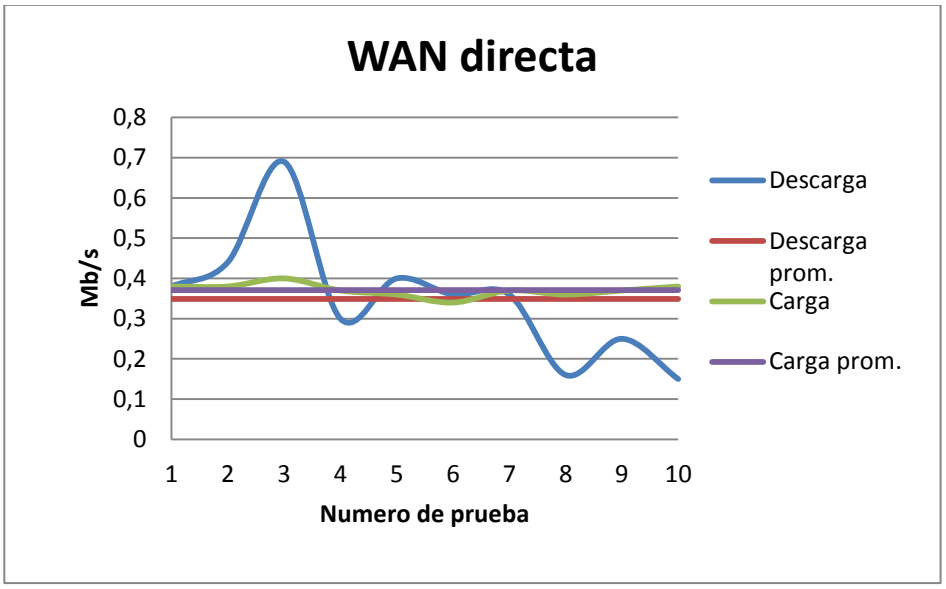


Figura 5.7 – Prueba de velocidad directa WAN.

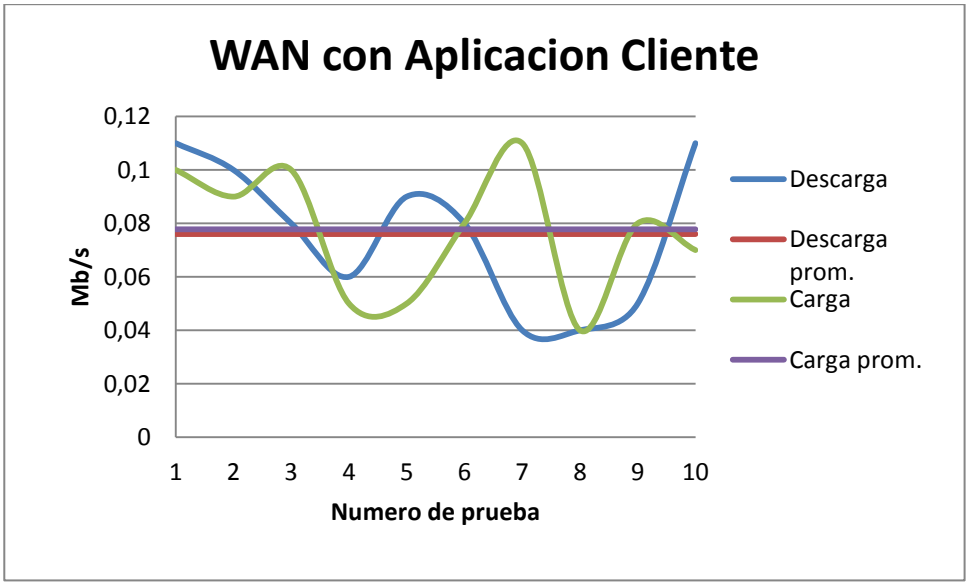


Figura 5.8 – Prueba de velocidad Aplicación Cliente WAN.

5.6 Prueba 5: Escucha furtiva a google talk

Para la realización de esta prueba se hizo uso de la metodología de ataque de acceso escucha furtiva por medio de la utilización de Wireshark un analizador de protocolos. Se utilizaron los ordenadores 5 y 7 (tabla E.13) así como el enrutador (tabla E.14), los cuales se interconectaron en un ambiente de LAN como se muestra en la figura 5.9.

OBJETIVOS

- Comprobar que en una conversación de mensajería instantánea que utiliza Google Talk la información viaja sin protección alguna.
- Comprobar que para la conversación anterior con la utilización de la aplicación cliente la información viaja cifrada.

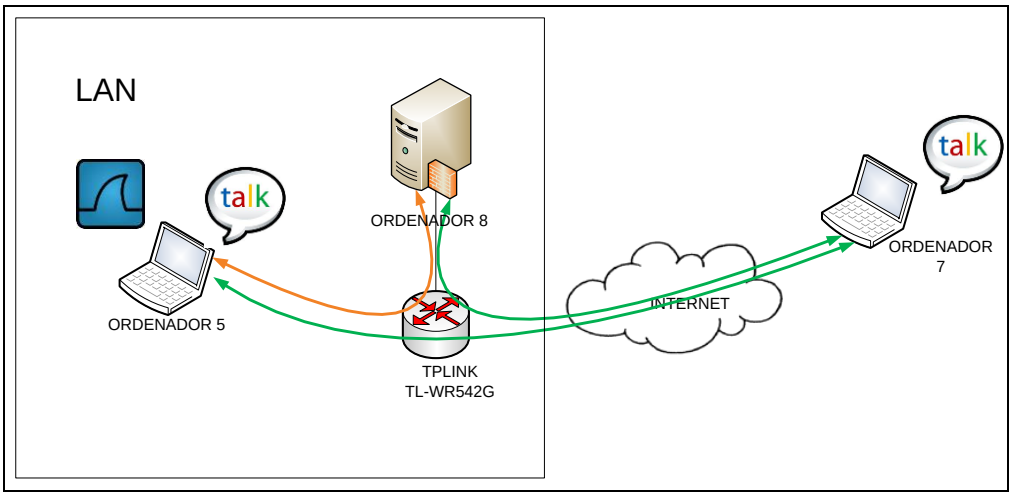


Figura 5.9 – Escucha furtiva a Google Talk

PROCEDIMIENTO

La figura 5.9 muestra en verde la conexión directa a Internet que será utilizada para la conversación con Google Talk como *escenario 1*, en naranja se muestra la conexión a utilizarse por medio de la aplicación cliente como escenario 2.

1. Se ejecuta Wireshark en el ordenador 5 y se selecciona la interfaz con la que se tiene conexión hacia Internet.
2. Para el *escenario 1* se ejecuta Google Talk e inicia sesión.
3. Se establece una conversación a la vez que se podrá ir observando en Wireshark los paquetes capturados.
4. Luego de capturar 2041 paquetes se guarda la captura para analizar los resultados.
5. Para el *escenario 2* se ejecuta la aplicación servidor, aplicación cliente y se configura en Google Talk el proxy como se muestra en la figura 6 del anexo G.
6. Se reinicia Google Talk e inicia sesión.
7. Se establece una conversación a la vez que se podrá observar en Wireshark los paquetes capturados.
8. Se capturan 2041 paquetes y se almacena para analizar resultados.

RESULTADOS

- Al analizar los paquetes capturados con Wireshark en el *escenario 1* se demuestra que la información viaja sin protección alguna (figura 5.10)

Figura 5.10 – Paquetes con conexión directa a Internet

- Para el *escenario 2* se han capturado 2041 paquetes con Wireshark, al analizar cada uno de estos paquetes se demuestra que la información viaja cifrada como se observa en la figura 5.11.

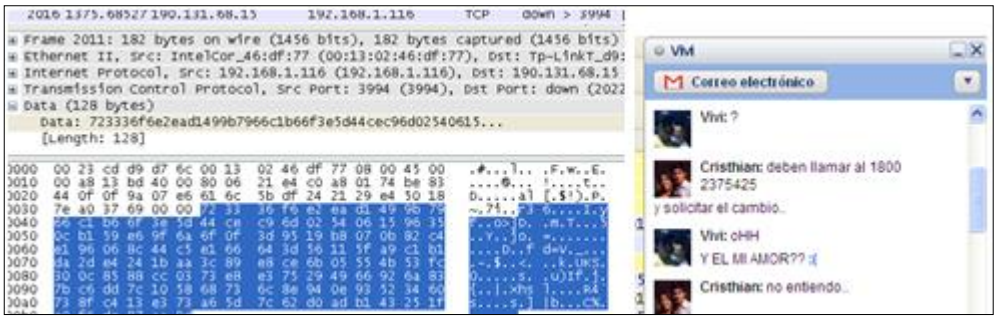


Figura 5.11 – Paquetes con aplicación cliente.

5.7 Prueba 6: Intercepción a páginas web

Para la realización de esta prueba se hizo uso de la metodología de ataque de acceso por intercepción. Se utilizarán los ordenadores 5, 6 y 8 (tabla E.13) así como el enrutador (tabla E.14), los cuales se interconectaron en un ambiente de LAN como se muestra en la figura 5.12

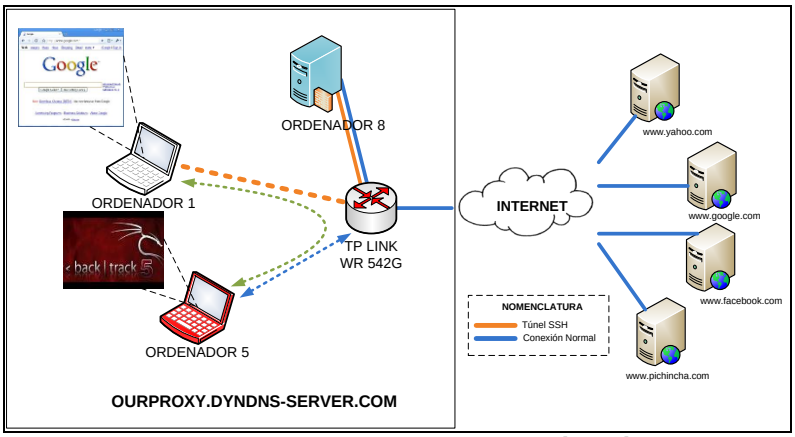


Figura 5.12 – Prueba 6: Ataque Intercepción tráfico web SSL

OBJETIVOS

- Comprobar que es posible interceptar las solicitudes de páginas https y mostrarle al usuario una página http.
- Comprobar que es posible capturar información relevante como usuarios y contraseñas.
- Comprobar que con la utilización de la aplicación cliente se tiene mayor seguridad.

PROCEDIMIENTO

Se ha realizado escucha furtiva con la navegación web del ordenador 1 desde el ordenador 6 por medio de Wireshark, como se muestra en las figuras F.7 y F.8 del anexo F la información viaja cifrada y por tanto no se puede tener acceso a ella, por ello se procede a realizar un ataque de intercepción en dos escenarios: *escenario 1* en verde y azul, redirige el tráfico web del ordenador 1 al ordenador 6 para su análisis y luego hacia los servidores web; como *escenario 2* en naranja donde la información ya no puede ser analizada por el ordenador atacante. El procedimiento es el siguiente.

1. Para el *escenario 1* iniciar el ataque de intercepción como se muestra en el anexo E.
2. En el ordenador 1 iniciar la navegación web en sitios como: gmail, hotmail, yahoo, facebook e iniciar sesión en cada uno de ellos; se podrá observar en el ordenador 6 los datos capturados.
3. Para el *escenario 2* ejecutar la aplicación servidor en el ordenador 8 y la aplicación cliente en el ordenador 1.
4. Configurar en el explorador web el proxy.
5. Iniciar la navegación web y observar el ordenador 6 los datos capturados.

RESULTADOS

- En el *escenario 1* se ha logrado interceptar los usuarios y contraseñas de cuentas de gmail, hotmail, facebook como se ilustra en la figura 5.13 demostrando que al navegar en internet se ofrece una falsa seguridad.
- En el *escenario 2* al utilizar el túnel SSH no se ha logrado capturar la información que se capturó en el *escenario 1* dado que la información se transmite por otros puertos y además se encuentra cifrada, mostrando que existe mayor seguridad en la transmisión de la información.

Figura 5.13 – Intercepción de contraseñas

5.8 Prueba 7: Ataque de fuerza bruta a ssh

Para la realización de esta prueba se utilizaron los ordenadores 6 y 8 (tabla E.13) así como el enrutador (tabla E.14), los cuales se interconectaron en un ambiente de LAN como se muestra en la figura 5.14.

OBJETIVOS

- Comprobar que es posible descubrir una contraseña mediante un ataque por fuerza bruta.
- Comprobar que algunas configuraciones en openssh no son realmente efectivas contra ataques de fuerza bruta.
- Comprobar que el uso de la herramienta fail2ban brinda protección contra ataques de fuerza bruta.

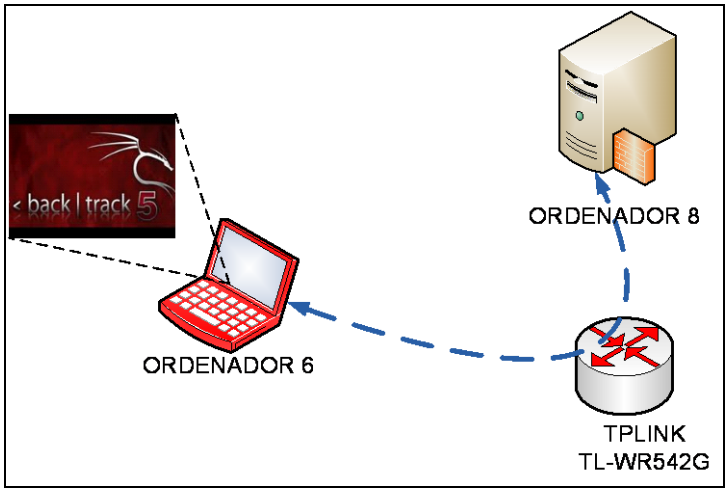


Figura 5.14 – Prueba 7: Ataque por fuerza bruta

PROCEDIMIENTO

Para la realización de esta prueba se hará uso de la herramienta "medusa**"** incluida dentro del sistema operativo Backtrack. El procedimiento a seguir es el siguiente.

1. Para el *escenario 1* en el ordenador 6 comprobar en qué puerto se ejecuta el servicio a atacar, mediante nmap.
2. Crear un archivo de texto que contenga combinaciones de letras y números a ser utilizado para el ataque; coloque 10 contraseñas incorrectas y al final la correcta.
3. Ejecutar en una terminal la siguiente línea de comando: *medusa –h [ip Destino] –u [usuario] –P [nombre del archivo o diccionario] –M ssh –n [num Puerto].*
4. Se mostrará en la terminal los intentos de autenticación, finalmente si se encuentra la correcta se marcará como *success* que sería la decimo primera contraseña del archivo nombrado en el paso 2**.**
5. Para el *escenario 2* realizar las configuraciones para limitar los intentos de autenticación en openssh modificando el archivo "sshd_config" como se muestra en el anexo B.
6. Realizar el paso 3 y verificar si se logra autenticar o si se deniega el acceso.
7. Para el *escenario 3* instalar fail2ban y modificar el archivo para limitar los intentos de autenticación en el servicio ssh.

8. Realizar el paso 3 y verificar si se logra autenticar o si se deniega el acceso.

RESULTADOS

- En el *escenario 1* se logra encontrar la contraseña luego de 11 intentos, esto muestra que si no se limita el número de intentos de autenticación en un momento dado se puede encontrar la contraseña de un usuario y acceder a los recursos y comprometer el sistema de algún modo.
- En el *escenario 2* se logra encontrar la contraseña igual que en el escenario anterior a pesar de haber configurado el máximo de intentos en openssh.
- Finalmente en el *escenario 3* como lo muestra la figura 5.15 al cuarto intento de autenticación el acceso es denegado al ordenador 6, bloqueando eficazmente el intento de ataque.

Figura 5.15 – Contraseña encontrada con ataque de Fuerza Bruta

5.9 Prueba 8: Ataque “Man-in-the-middle” a ssh

Para la realización de esta prueba se utilizarán los ordenadores 1, 6 y 8(tabla E.13) así como el enrutador (tabla E.14), los cuales se interconectaron en un ambiente de LAN como se muestra en la figura 5.16.

OBJETIVOS

- Comprobar que la arquitectura cliente servidor SSH es vulnerable a ataques de intercepción o intermediario bajo ciertas circunstancias.
- Comprobar que la arquitectura cliente servidor SSH es segura cuando se considera la huella digital del servidor como única durante el proceso de conexión y se realiza la respectiva validación.

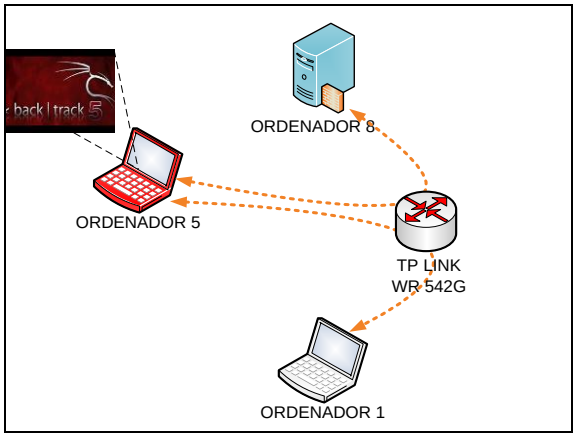


Figura 5.16 – Prueba 8: Ataque intercepción al servicio SSH

PROCEDIMIENTO

Se utilizará “ettercap**”** herramienta incluida en backtrack para monitorear analizar el tráfico además se debe instalar **“**jmitm2**”**una aplicación utilizada para realizar ataques de intercepción que simula un servidor ssh. Se ha desarrollado la prueba en dos escenarios, en el primero no se considera como única la huella digital del servidor ssh y se permite la conexión sin importar la misma; mientras en el segundo si se la considera y se rechazan las conexiones de servidores cuyas huellas digitales no coincidan con la almacenada. El procedimiento que se ha seguido es el siguiente.

1. Para el *escenario 1* ejecutar el servicio SSH en el ordenador 8
2. Lanzar el ataque intercepción de acuerdo al anexo E.
3. Iniciar la aplicación cliente en el ordenador 1.
4. Aceptar la advertencia de conexión huella digital desconocida figura 11 del anexo G y autenticarse.
5. Para el *escenario 2* ejecutar el servicio SSH en el ordenador 8
6. Modificar el código de la aplicación cliente con la validación de la huella digital.
7. Repetir el segundo y tercer paso.
8. La aplicación cliente no le permitirá autenticarse y le advertirá que posiblemente se esté llevando una ataque intercepción.

RESULTADOS

- Para el *escenario 1* como se muestra en la figura 5.17 es posible capturar los datos de autenticación de cualquier usuario que intente conectarse a un servidor ssh si no se toma en cuenta la huella digital del mismo y se permite la conexión si importar esta.
- Para el *escenario 2* se bloquea la conexión al servidor si la huella digital no coincide con la almacenada, esto quiere decir que posiblemente exista un ataque intermediario, no se permite al usuario aceptar la conexión pues sus datos estarían expuestos; al hacer esto se ha brindado mayor seguridad al sistema.

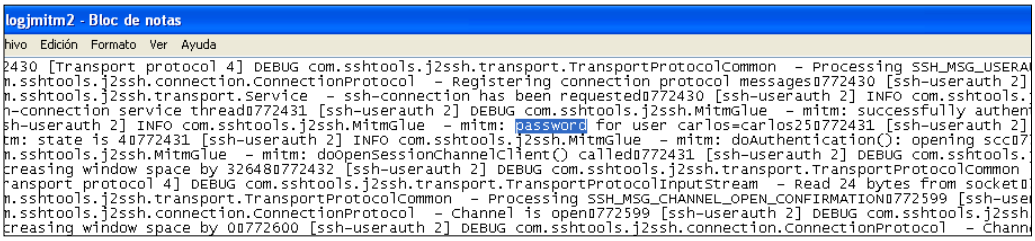


Figura 5.17 – Datos capturados por ataque intermediario.

C O N C L U S I O N E S

1. Como lo muestra el presente proyecto la utilización del modelo cliente/servidor permite la centralización de los recursos y servicios facilitando la escalabilidad de un sistema y la disponibilidad de dichos recursos y servicios.
2. De acuerdo a la prueba 1 de portabilidad se ha logrado demostrar que una aplicación desarrollada en Java es portable; ya que se logró ejecutar la misma en diferentes sistemas operativos sin reportarse problema alguno, debido a esta característica el usuario de la aplicación obtendrá un ahorro de tiempo y recursos ya que no necesitará adquirir una aplicación específica para cada sistema operativo.
3. Como se muestra en la implementación del proyecto en la actualidad existen equipos de red para hogares (fácil uso y bajo coste) que permiten prestar servicios desde una red local hacia el internet como el enrutador utilizado en las pruebas facilitando la realización de prototipos.
4. La reutilización de código de libre distribución permite el desarrollo acelerado de aplicaciones dado que se puede hacer uso del código y modificar el mismo mejorando sus prestaciones o adaptándolo a la aplicación que se desea desarrollar disminuyendo la inversión de tiempo que tomaría desarrollar todo desde cero.

5. Considerar en la aplicación cliente la validación de la huella digital del servidor del sistema como única, tal como se implementó en la aplicación cliente; protege al sistema de ataques (man-in-the-middle), evitando el robo de datos de autenticación y posterior de información personal.
6. De acuerdo a las pruebas de velocidad en un ambiente LAN las velocidades de carga y descarga aumentan con el uso de la aplicación cliente mientras en un ambiente WAN las velocidades disminuyen; concluyéndose que el sistema es más eficiente en términos de velocidad en una LAN a pesar de que en términos de seguridad el rendimiento es el mismo para ambos ambientes.
7. Los navegadores web nos brindan una falsa sensación de seguridad pues como se puede observar en la prueba 6 con un ataque de intercepción es posible capturar información relevante de los usuarios. La implementación de aplicaciones tal como la desarrollada en este proyecto permitirán brindar mayor seguridad a usuarios finales.
8. Algunas aplicaciones de mensajería instantánea transmiten la información entre sus usuarios sin hacer uso de un algoritmo criptográfico, exponiendo ésta a accesos no autorizados; tal y como se demuestra en la prueba 5 (escucha furtiva a google talk); el usuario promedio no percibe estos problemas de seguridad por lo que la aplicación desarrollada intenta minimizar estas vulnerabilidades.

R E C O M E N D A C I O N E S

1. Para la creación del túnel SSH entre cliente y servidor se utiliza autenticación con usuario y contraseña, sin embargo como trabajo futuro pude considerarse realizar la autenticación con la generación de llaves RSA o DSA, al ser estas de mayor longitud y complejidad se tendrá un grado mayor de seguridad.
2. La aplicación servidor no tiene una interfaz gráfica, se podría añadir una con funcionalidades de administración de los servicios que el mismo proporciona.
3. Se podrían añadir algunos otros servicios como el de transferencia de archivos entre clientes y envío de correo electrónico.
4. Se recomienda a las empresas implementar sistemas como el que se ha desarrollado a fin de mejorar la seguridad y proteger sus datos.

ANEXO A

ANEXO A: INSTALACIÓN DE RECURSOS

NETBEANS

Es un entorno de desarrollo integrado principalmente para el lenguaje de programación Java, es un producto libre, gratuito y sin restricciones de uso; permite la creación modular de aplicaciones. Se muestra a continuación el proceso de instalación.

- Descargar la última versión en: www.netbeans.org

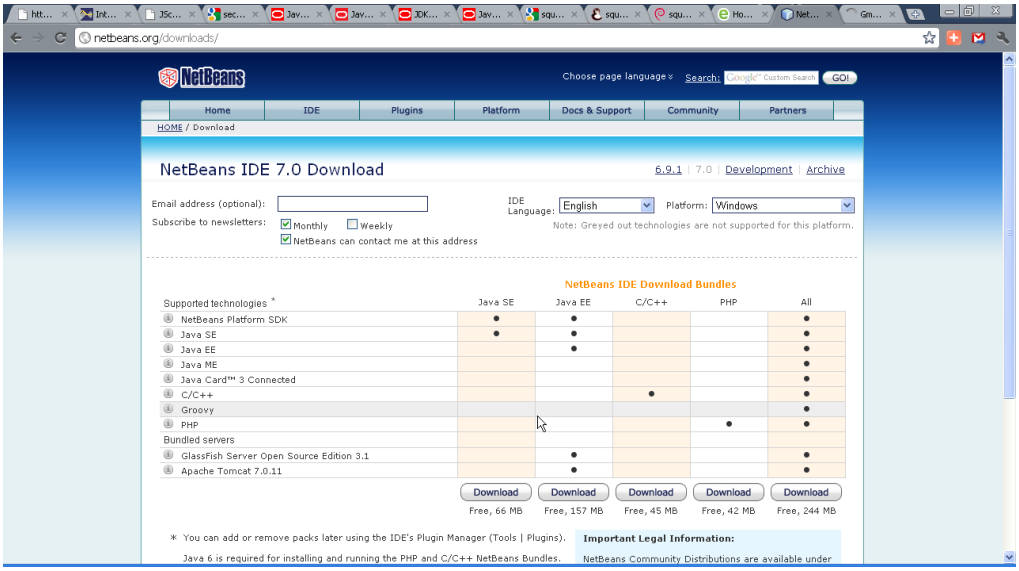


Figura A.1 - Pagina web y opciones de descarga Netbeans

- Ejecutar el archivo ejecutable de extensión .exe.

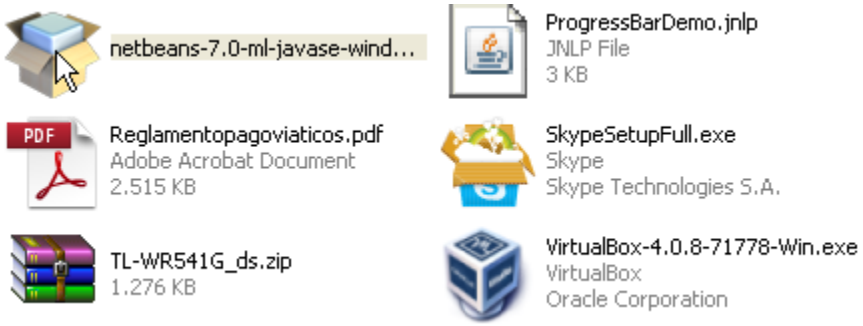


Figura A.2 - Archivo ejecutable .exe

- Escoger instalación predeterminada y con ello tenemos instalado el IDE y listo para implementar una aplicación en JAVA.

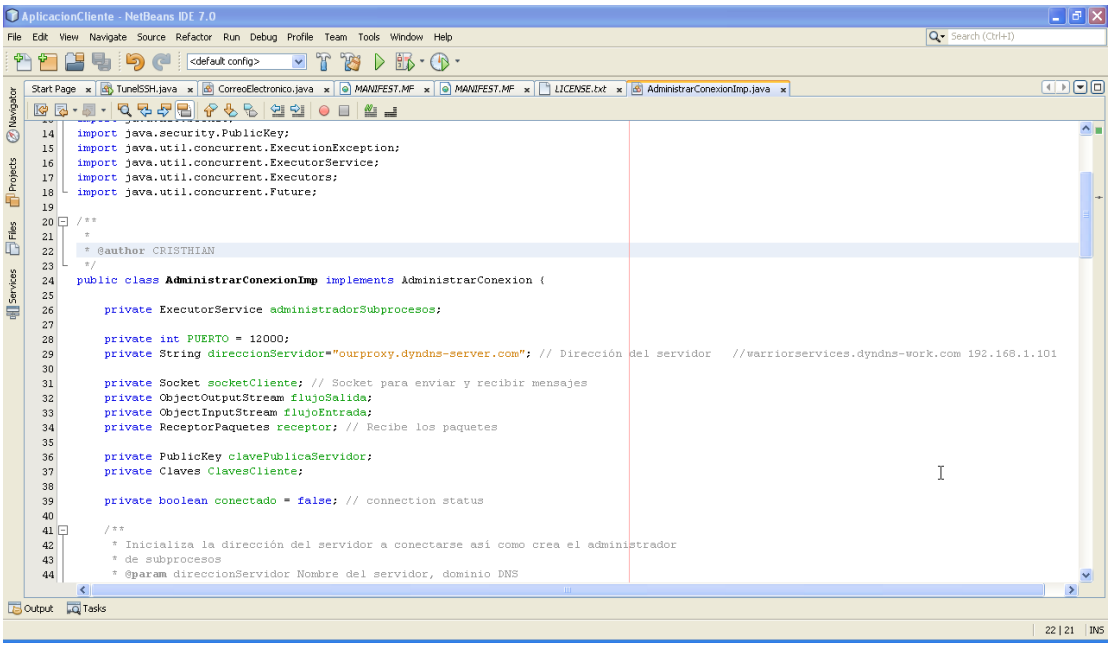


Figura A.3 - Interfaz principal Netbeans 7.0

LIBRERÍAS

El procedimiento para añadir librerías al entorno de desarrollo Netbeans es el siguiente.

- Seleccionar añadir librería en el proyecto (Figura 4).
- Darle un nombre a la librería (Figura 5).
- Escoger el archivo de extención.JAR (Figura 6).
- Finalmente se podrá observar la librería añadida y se podrá hacer uso de sus métodos (Figura 7).

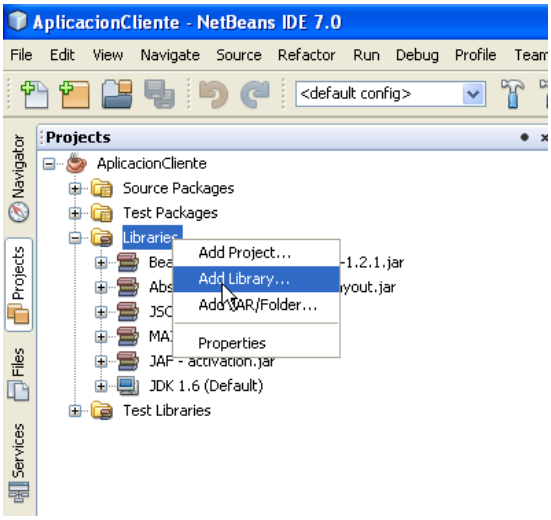


Figura A.4 - Añadir librería a Netbeans

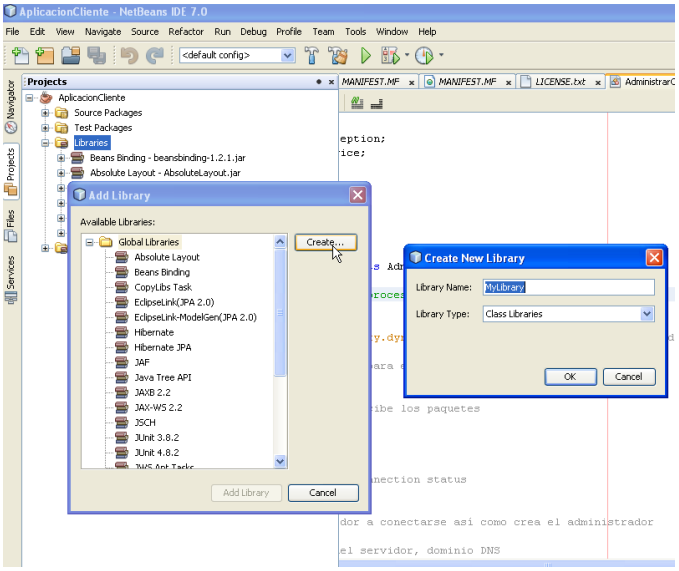


Figura A.5 - Dar nombre a la librería

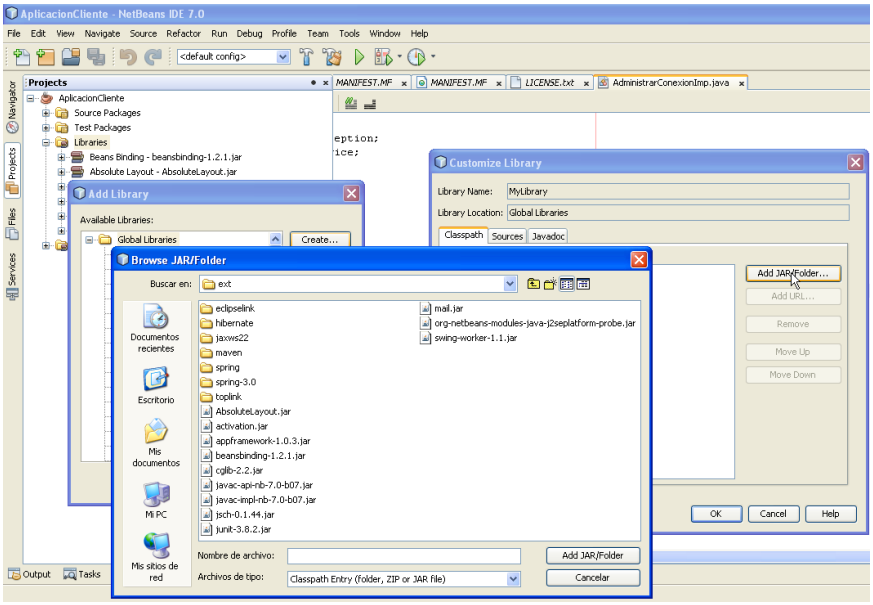


Figura A.6 - Escoger archivo .JAR

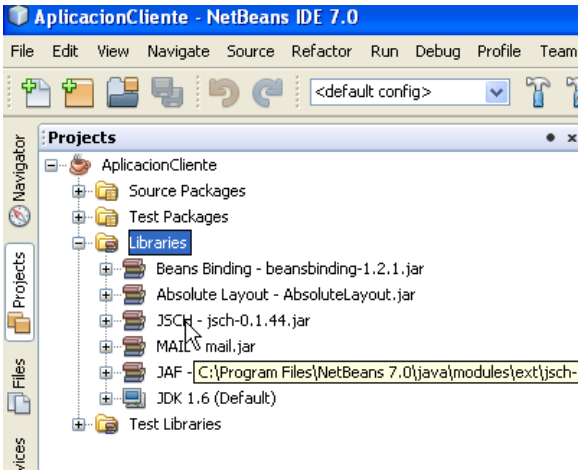


Figura A.7 – Librería añadida

SERVIDOR SSH

Se ha elegido OPENSSH como servidor SSH, por tanto se debe ejecutar en la terminal de Ubuntu el siguiente comando: **sudo apt-get install openssh-server**, una vez descargado el paquete se instala y ejecuta como servicio (Figura 8).

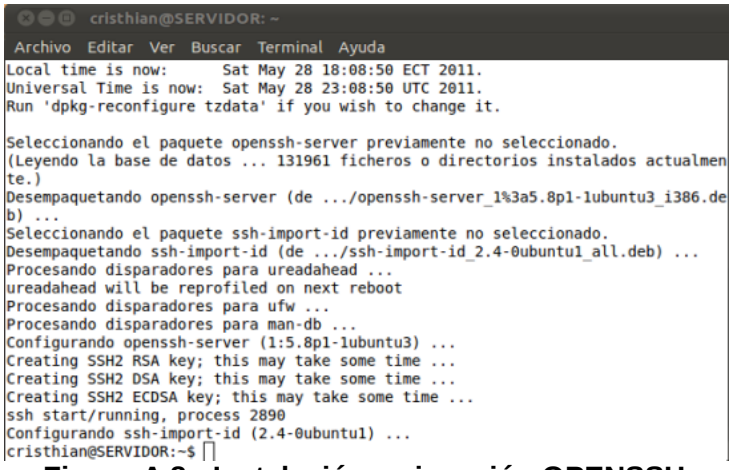


Figura A.8 - Instalación y ejecución OPENSSH

Se comprueba que se encuentra instalado escaneando los puertos abiertos por medio de **nmap** (Figura 9), el puerto configurado de forma predeterminada es el 22.

Figura A.9 – NMAP puerto 22 abierto

SERVIDOR PROXY

Se ha elegido SQUID v 3.1 como servidor PROXY por lo que se debe ejecutar en la terminal de Ubuntu el siguiente comando: **sudo apt-get install squid3**, se descargará el paquete se instalará y ejecutará como servicio (Figura 10).

Figura A.10 – Instalación y ejecución SQUID

Se comprueba que SQUID se encuentra instalado escaneando los puertos abiertos por medio de **nmap** (Figura 11)**,** el puerto configurado por default es el 3128.

Figura A.11 - NMAP puerto 3128 abierto

APLICACIÓN CLIENTE

Para la correcta ejecución es necesario que el sistema operativo tenga instalado el JRE 1.6. En Windows e ubica el archivo .JAR de la aplicación y se lo ejecuta.

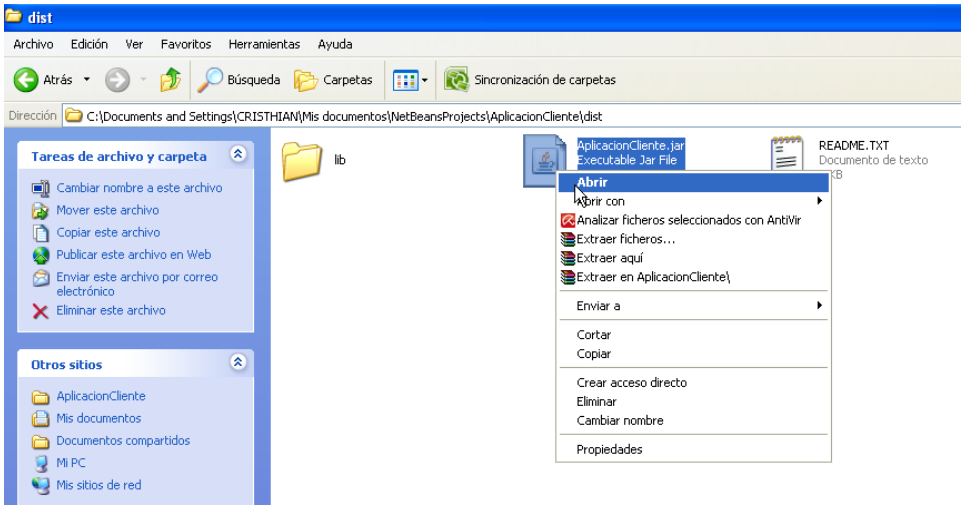


Figura A.12 – Archivo .JAR ejecutable

Una vez lograda la sincronización con el servidor se tendrá lo que muestra la figura 13 en el área de notificación que indica que la aplicación se ejecuta con éxito.

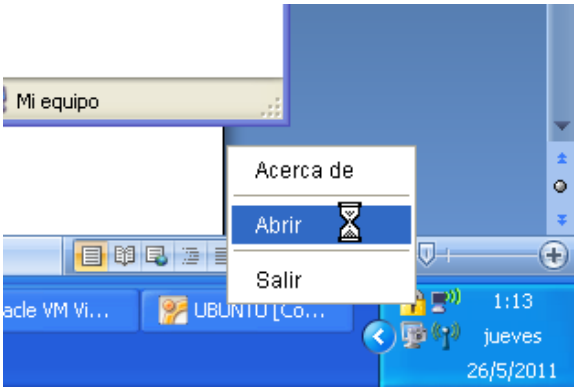


Figura A.13 – Aplicación cliente ejecutada

ANEXO B

ANEXO B: CONFIGURACIONES ADICIONALES

ENRUTADOR

En el enrutador es necesario configurar un dns dinámico para que ordenadores que se encuentre fuera de la LAN puedan solicitar los servicios que brindará el servidor, además es necesario configurar el reenvío de las peticiones desde el exterior por medio de la creación de servidores virtuales; estas configuraciones se detallan a continuación.

DNS DINÁMICO.-Se puede obtener un DNS Dinámico gratuito de un proveedor como DYNDNS, para ello se debe crear una cuenta en su página web (Figura 1). Una vez creada la cuenta se podrá añadir un dominio seleccionando la opción Host Services (Figura 2). Se solicitará el ingreso del nombre del dominio elegido y la IP pública que lo representará (Figura 3). Finalmente se puede configurar el dominio obtenido en el enrutador (Figura 4)

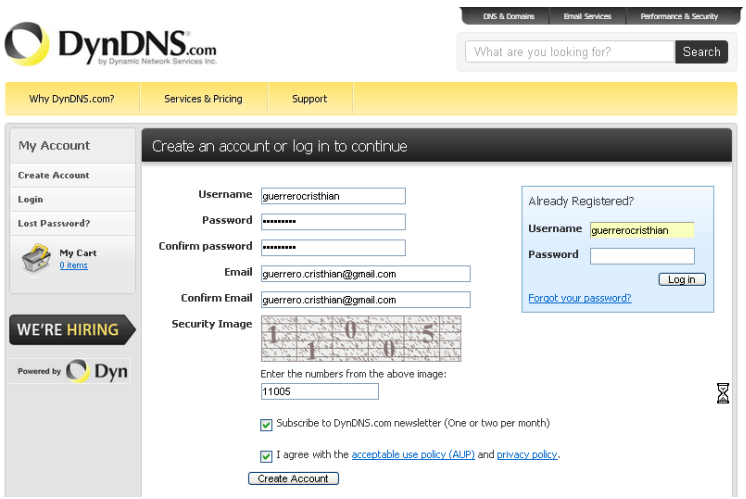


Figura B.1 – Creación de cuenta en DynDns

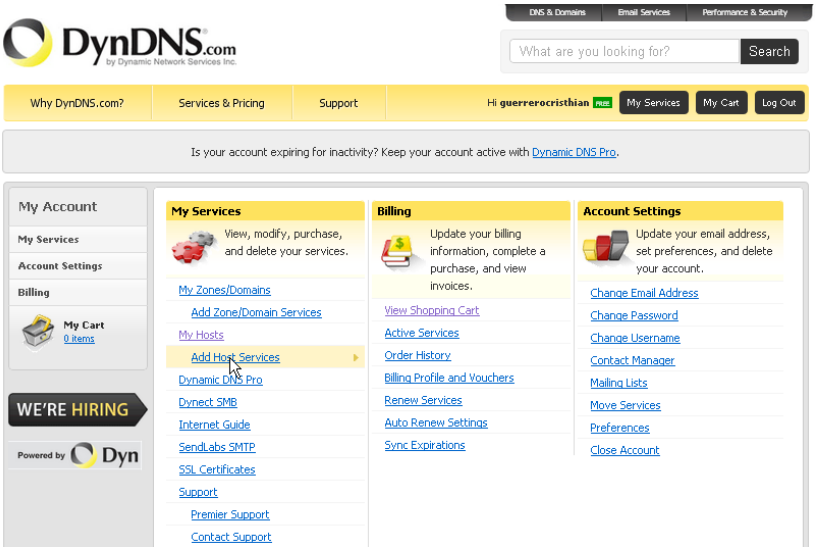


Figura B.2 – Opciones de la cuenta DynDns

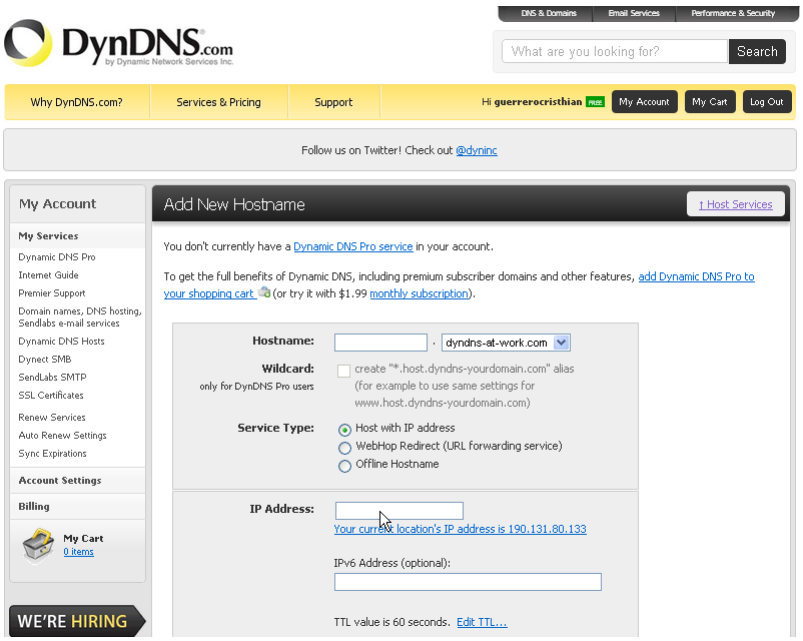


Figura B.3 Creación del dominio dinámico

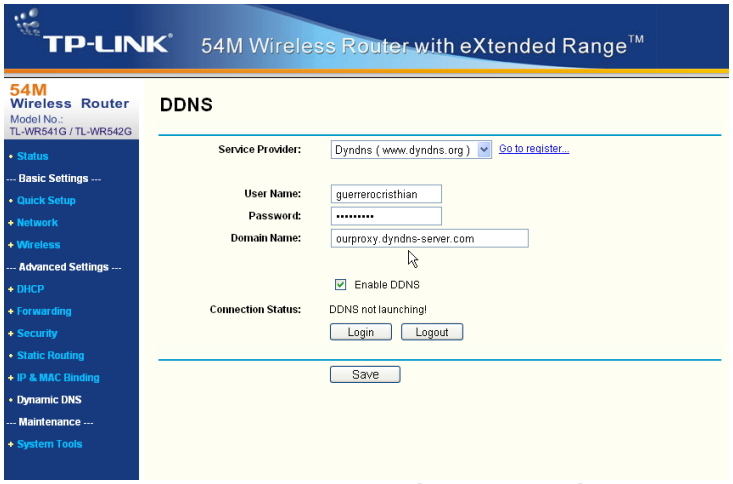


Figura B.4 - Configuración DNS Dinámico

SERVIDORES VIRTUALES.- se debe configurar servidores virtuales que no son más que reenvío de peticiones para que el servidor atienda peticiones desde fuera de la LAN. Se debe especificar la IP de nuestro servidor: 192.168.1.101 y los puertos en los que se encontrará prestando los servicios: túnel SSH-2022 y mensajería instantánea entre 11000 y 12000 (Figura 5).

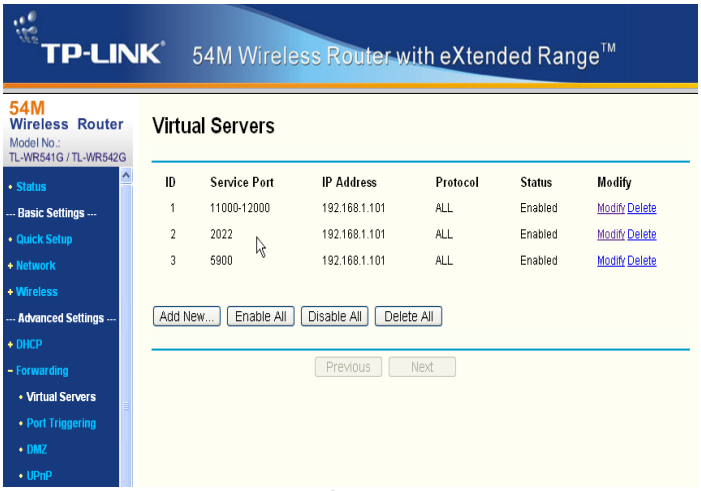


Figura B.5 - Configuración Servidores Virtuales

SERVIDOR PROXY

Para realizar cambios en la configuración de SQUID se debe modificar el archivo **/etc/squid3/squid.conf**. Una buena práctica de seguridad es modificar el puerto que por lo general viene configurado (3128) por otro, en este caso colocaremos 9090 (Figura 6).

Figura B.6 – Cambio de puerto SQUID

Se restringirá el acceso desde el exterior y solo se permitirá el acceso local, es decir; solo aplicaciones que se ejecuten en el servidor podrán solicitar el servicio de proxy. La restricción se logra por medio de ACLs o listas de acceso, además de declararlas (Figura 7) se debe configurar si esta lista se permitirá o se restringirá (Figura 8).

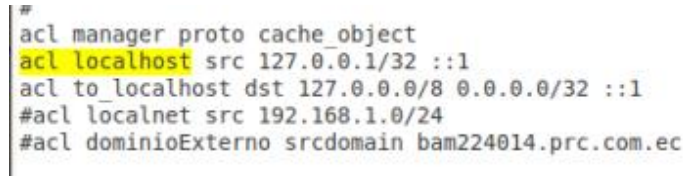


Figura B.7 – Cambio en las listas de acceso SQUID

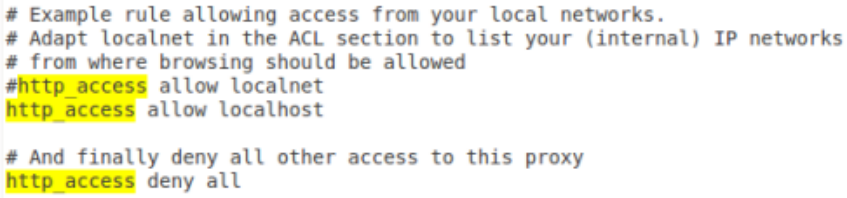


Figura B.8 – Denegar o permitir listas de acceso

SERVIDOR SSH

Para realizar cambios en la configuración de SQUID se debe modificar el archivo **/etc/ssh/ssh_config**. Una buena práctica de seguridad es modificar el puerto que por lo general viene configurado (22) por otro, en este caso colocaremos 2022 (Figura 9).

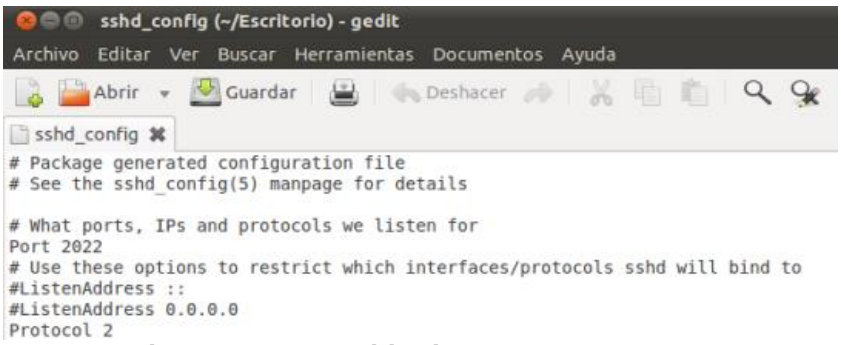


Figura B.9 – Cambio de puerto OPENSSH

Se deshabilitará la autenticación como usuario administrador a fin de que el archivo sshd_config no sea modificado remotamente y se limitarán los intentos de autenticación a 3(Figura 10) a fin de que los intentos de romper claves por fuerza bruta fracase.

Figura B.10 – Cambio en autenticación OPENSSH

ORDENADOR CLIENTE

Una vez instalada la aplicación cliente y a fin de que se haga uso de la funcionalidad de Tunneling que esta brinda se debe realizar una configuración adicional en el sistema operativo. En sistemas windows se debe ir a Inicio/Panel de Control/Opciones de internet que mostrará las propiedades de las conexiones que tiene su ordenador (Figura 11), dependiendo de cuál de ellas se utilice , deberá escoger la opción de configuración; esta le permitirá colocar la dirección y el puerto, que deben ser los que se muestran en la Figura 11.

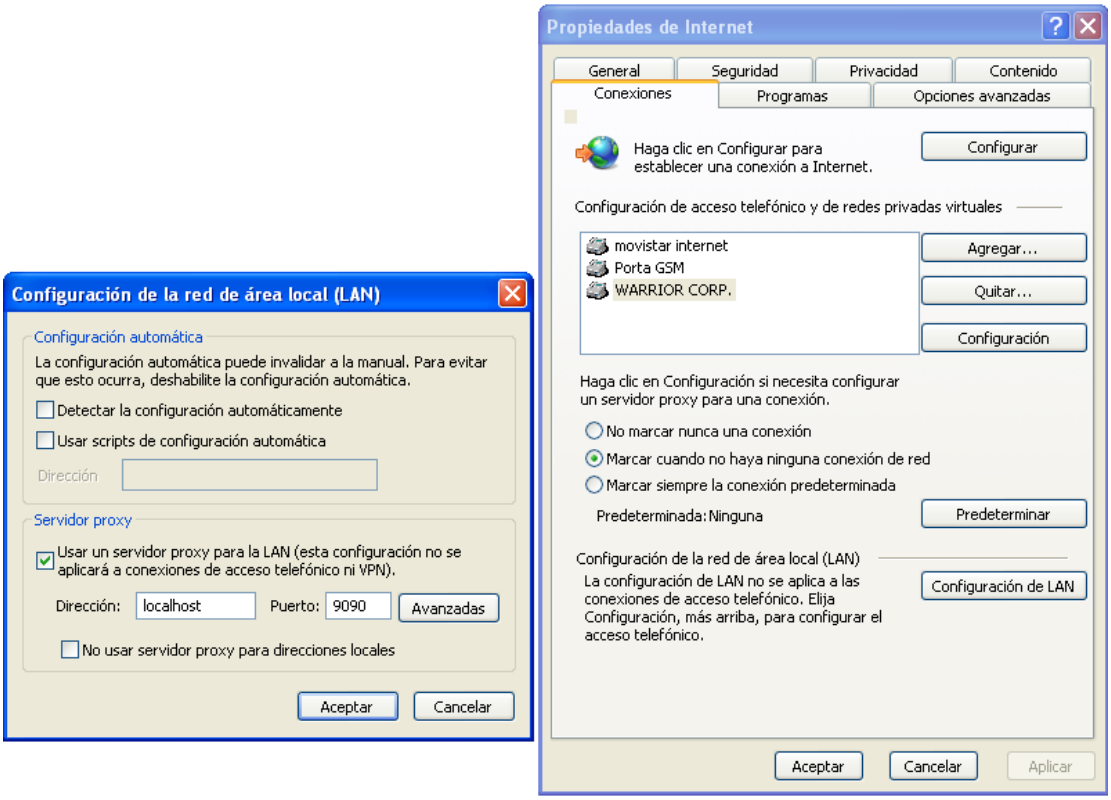


Figura B.11 – Configuración proxy en Windows

ANEXO C

ANEXO C: DETALLES Y PROCEDIMIENTOS

ALGORITMO RSA

Este algoritmo se basa en tres pasos esenciales: generación de llaves, cifrado y descifrado cuyos detalles se muestran a continuación.

Generación de Claves.- para esto ambos usuarios eligen por separado dos números primos **p** y **q** distintos. El tamaño de estos dos números **p** y **q** harán seguro al algoritmo, tomar valores p y q extremadamente grandes (mayores a 1024 bit) harán que RSA sea robusto y seguro.

Luego se hará el cálculo de $n = p \times q$, también se calcula $\boldsymbol{\varphi} = (p-1)(q-1)$, donde $\boldsymbol{\varphi}$ es la funcion de euler. Se selecciona ademas un numero **e** menor que $\boldsymbol{\varphi}$ y coprimo con $\boldsymbol{\varphi}$**,** el cual debera ser gran tamaño como los **p** y **q**. Como ultimo paso se procedera a determinar un numero **d** que cumpla con $\boldsymbol{d} = residuo\ \frac{e^{-1}}{\varphi} e^{-1}$, donde **d** sera el multiplicado modular inverso de **e** **modulo φ.** Los pasos anteriores permitiran obtener la llave publica (**n, e**), representa el modulo y exponente del cifrado respectivamente, y la llave privada (**n, d**), representa el modulo y exponente del descifrado respectivamente.

Cifrado.-Se toma la llave pública (**n, e**) y se calcula lo siguiente:

$$\boldsymbol{Texto\ Cifrado} = residuo\left(\frac{Texto\ original^{e}}{n}\right)$$

Descifrado.- para recuperar la información cifrada **m** se toma la llave privada (**n, d**) y se calcula lo siguiente:

$$\boldsymbol{Texto\ original} = residuo\left(\frac{Texto\ cifrado^{d}}{n}\right)$$

Este algoritmo permite también la autenticación, esto se logra usando la llave privada (**d, n**) para cifrar y la llave pública (**e, n**) para descifrar. El tamaño de los números p y q debe considerarse de acuerdo a la Tabla 1.

Tiempo de Vida del Dato	RSA Tamaño de Clave
Hasta el año 2010	1024 bits
Hasta el año 2030	2048 bits
Hasta 2031 y hacia adelante	3072 bits

Tabla C.1 - Recomendaciones del tamaño de claves RSA
Fuente: www.javamex.com

PROTOCOLO SSH

Como se muestra en la Figura 3 del Anexo F, el protocolo SSH se encuentra dividido en tres protocolos: protocolo de capa de transporte, protocolo de autenticación de usuario y protocolo de conexión, se detalla la función de cada uno a continuación:

PROTOCOLO DE CAPA DE TRANSPORTE.- provee la autenticación del servidor, confidencialidad e integridad. La **autenticación** es basada en host, no en usuario. El servidor normalmente escucha en el puerto 22, del lado del cliente se escoge un puerto arbitrariamente; una vez establecida la conexión ambos sitios envían una cadena de identificación en el que se envía la versión del protocolo y la versión del software. Durante la etapa de intercambio de claves se negociará un algoritmo de cifrado y una clave, cada porción de cada paquete será cifrada con dicho algoritmo [1].

A fin de proteger la **integridad** de la información, se incluye con cada paquete un MAC (Message Authentication Code) que es calculado a partir de la clave secreta, el número de secuencia y el contenido del paquete; El algoritmo MAC (Tabla 2 del Anexo E) también es negociado durante el intercambio de claves. La figura 2 del anexo F muestra la formación del paquete SSH.

[1]. **Ylonen, Tatu y Lonvick, Chris.** "*RFC-4253*". eitf.org. [Revisado el: 20 de Mayo del 2011].

Un método de intercambio de claves (Tabla 3 del Anexo E) especifica como para una sesión, las claves son generadas para cifrado y **autenticación**, y como la autenticación del servidor es realizada [1]. SSH ha sido diseñado para operar con al menos un formato de clave pública, codificación y algoritmo (firma y/o cifrado) (Tabla 4 del Anexo E). El intercambio de claves inicia con el envío tanto de parte del cliente como del servidor de la lista de los algoritmos soportados, el método de intercambio de claves puede usar autenticación de servidor explicita si los mensajes que se intercambian incluyen una firma o alguna otra prueba de la autenticidad del servidor o puede usar autenticación de servidor implícita si el servidor a fin de probar su autenticidad envía al cliente un mensaje con un MAC que el cliente pueda verificar. Luego del intercambio el cliente puede solicitar un servicio [1], si esta solicitud es rechazada por el servidor la conexión se cerrará.

PROTOCOLO DE AUTENTICACIÓN DE USUARIO.- autentica al cliente con el servidor, trabaja sobre el protocolo de capa de transporte y presupone que esta le provee integridad y confidencialidad de los datos; el servidor realiza solicitudes al cliente, según el formato de la Figura 1.

[1]. **Ylonen, Tatu y Lonvick, Chris.** "*RFC-4253*". eitf.org. [Revisado el: 20 de Mayo del 2011].

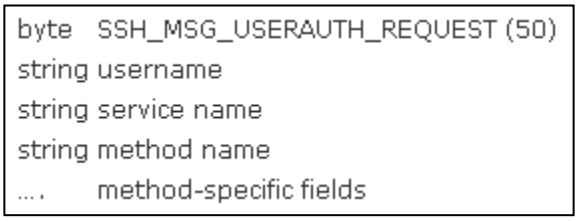


Figura C.1: Formato de solicitud de autenticación de Cliente SSH
Fuente: RFC 4252 [2]

Los métodos de autentificación que el servidor puede solicitar son los siguientes: clave pública, contraseña y basado en ordenador; se detalla cada uno a continuación. Llave publica.- en general el cliente envía un mensaje que contiene su clave pública, firmado con su clave privada; cuando el servidor lo recibe verifica que la clave sea aceptable para autentificación, si lo es verifica que la firma sea correcta. Contraseña.- El cliente envía una contraseña que es protegida por el cifrado del protocolo de capa de transporte. Basado en ordenador: Un ordenador que soporta varios clientes puede proveer autenticación para todos ellos, este método trabaja mediante el envío de parte del cliente de una firma basada en la clave privada del ordenador cliente, por tanto en lugar de que el servidor verifique la identidad del usuario verifica la identidad del ordenador [2].

PROTOCOLO DE CONEXIÓN.- multiplexa el túnel cifrado en múltiples canales lógicos, proporciona sesión de clientes interactiva, ejecución remota de comandos, redirección de conexiones TCP/IP y redirección de conexión X11 [2].

La vida de un canal pasa por tres estados: apertura del canal, transferencia de datos y cierre del canal (Figura 4 del Anexo F). Cuando un sitio desea abrir un canal, este designa un número de canal y envía un mensaje con el formato de la Figura 2:

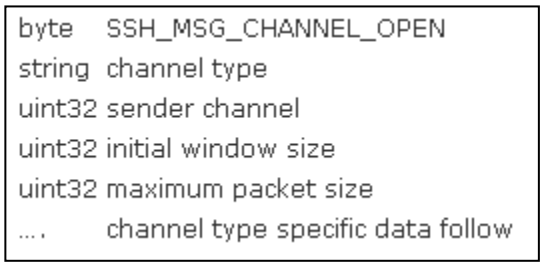


Figura C.2: Formato Apertura de Canal - Protocolo de Conexión
Fuente: RFC 4254 [3].

Hay 4 tipos de canales reconocidos en la especificación del protocolo de conexión: Session, x11, forwarded-tcpip y direct-tcpip. Una de las características más útiles de SSH es port-forwarding, ésta provee la habilidad de convertir una conexión TCP insegura en una conexión SSH segura, es denominada también **Tunneling** (Figura 5 del Anexo F).

LIBRERÍAS JAVA

Para la implementación de las aplicaciones, tanto cliente como servidor se utilizaron las siguientes librerías JAVA: JDBC, JSCH, JAF, JAVAMAIL, JAVA SECURITY y JAVA CRYPTO, se detalla a continuación cuáles son sus funciones y por tanto que papel cumple dentro del proyecto.

JDBC (JAVA DATABASE CONECTIVITY).- Hace posible escribir aplicaciones que permitan: Conectarse a una fuente de datos, enviar consultas y actualizar parámetros de una base de datos y capturar y procesar resultados recibidos de la base de datos en respuesta a las consultas [4], la arquitectura el proceso se muestra en la Figura 12 del Anexo F.

JSCH (JAVA SECURE CHANNEL).- Es una librería que implementa SSH v2 en JAVA, permite la conexión a un servidor SSH y utilizar reenvió de puerto, reenvío de puerto X11, transferencia de archivos entre otras funcionalidades que se pueden integrar al desarrollo de una aplicación en JAVA [5]**.**

JAVAX CRYPTO.- proporciona las clases e interfaces para las operaciones criptográficas. Soporta cifrado simétrico y asimétrico, en

bloques y cifrado de flujo. Se proveen múltiples clases e interfaces, una clase relevante y utilizada en la implementación de este proyecto es Cipher que proporciona la funcionalidad de un sistema criptográfico, es decir cifrado y descifrado de la información (Figura 13 del Anexo F). Admite la implementación de algunos algoritmos como: AES y RSA de longitudes variables y DES y 3DES de longitudes fijas, sus principales métodos y su especificación se detalla en la Tabla 5 del Anexo E:

JAVA SECURITY.- incluye clases que implementan una fácil configuración y control de acceso compartido como arquitectura de seguridad. Soporta la generación y almacenamiento de clave pública incluido la generación de mensajes y firma. Finalmente este paquete soporta la generación de números aleatorios de clases proveedoras. Dos de sus principales clases son: keyPairGenerator y keyPair y dos de sus principales interfaces privateKey y publicKey. La clase KeyPairGenerator es utilizada para generar un par de claves pública y privada, el proceso de generación se muestra en la Figura 14 del Anexo F, contiene varios métodos que se describen en la Tabla 6 del Anexo E. Por otro lado KeyPair es un contenedor para un par de claves (una pública y una privada). Contiene varios métodos (Tabla 7 del Anexo E). La interfaz privateKey agrupa todas las interfaces de clave privada y publicKey agrupa todas las interfaces de clave pública.

ATAQUE INTERCEPCIÓN SSL

Para el ataque de intercepción se utilizaron herramientas como: arpspoof, sslstrip y ettercap incluidos en el sistema operativo backtrack 5 especializado para realizar auditorías informáticas instalado en el ordenador 6, es necesario estar conectado a una LAN y ejecutar las siguientes instrucciones en terminales:.

1. Configurar el ordenador atacante en modo de reenvío por medio de: **echo 1 > /proc/sys/net/ipv4/ip_forward**.
2. Configurar una ruta estática para interceptar el tráfico HTTP: **iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j REDIRECT --to-port 1000**.
3. Buscar los posibles objetivos con: **nmap -sP 192.168.1.0/24**.
4. Una vez identificado el objetivo, utilizar arpspoof para convencer a la red de enviar el tráfico de este hacia el ordenador atacante y luego del ordenador atacante hacia el enrutador. Para nuestro caso el objetivo y el router tiene configuradas las siguientes direcciones de red: 192.168.1.113 y 192.168.1.1 respectivamente ;para lanzar el ataque se debe ejecutar la siguiente expresión en una terminal:

   arpspoof -i eth0 -t 192.168.1.113 192.168.1.1.
5. Tomar el tráfico https y re direccionar como http al ordenador objetivo usando ssltrip: **python sslstrip.py -l 1000.**

6. Finalmente para mostrar los paquetes capturados se utiliza ettercap

   ettercap -Tq -i eth0.

ATAQUE INTERCEPCIÓN SSH

Para llevar a cabo este ataque, se considerará que el servidor ssh tiene la IP 192.168.1.101/24, el ordenador atacante la IP 192.168.1.110/24 y el ordenador víctima la IP 192.168.1.112/24. Los pasos para realizar el ataque son:

1. Configurar el ordenador atacante en modo reenvío:

   echo 1 > /proc/sys/net/ipv4/ip_forward.

2. Crear rutas en el ordenador atacante para redirigir el tráfico que se dirige al puerto donde se ejecuta el servicio ssh :

   iptables -t nat -A PREROUTING -p tcp --dport 2022 -j REDIRECT

   iptables -A FORWARD -j ACCEPT

3. Utilizar ettercap para hacer pensar al ordenador victima que el atacante es el servidor y al servidor que el ordenador atacante es el ordenador víctima:

   ettercap -Tq -M arp:remote /192.168.1.101/ /192.168.1.112/

4. Modificar en el ordenador atacante el archivo /root/jmitm2-0.1.0/bin/conf/server.xml con la ip del atacante y el puerto SSH.

5. Modificar en el ordenador atacante el archivo /root/jmitm2-0.1.0/bin/runm.sh con la IP del servidor SSH y el puerto donde se ejecuta este servicio.
6. Por último ejecutar el script ./runm.sh en el ordenador atacante.

ANEXO D

ANEXO D: MANUAL DE USUARIO

INTRODUCCIÓN

Este manual de introducción a Software de Tunneling ha sido elaborado con la intención de ofrecer la información necesaria para el uso de la aplicación cliente. Para el correcto funcionamiento de la aplicación cliente en el ordenador servidor debe estar correctamente configurado y debe ejecutar la aplicación servidor.

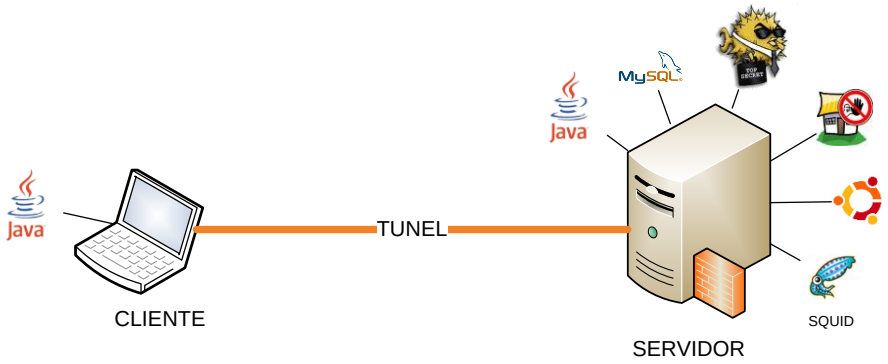


Fig. D.1 Conexión Cliente-Servidor Software de tunneling

Este manual sigue una estructura concreta, detallando los requerimientos del sistema, instalación de la aplicación, creación de cuenta de usuario, configuraciones generales y finalmente el uso de las funcionalidades.

REQUERIMIENTOS DEL SISTEMA

Los requerimientos de hardware para que la aplicación cliente “software de tunneling” se ejecute son los siguientes:

Procesador	RAM (mínimo)	RAM (recomendado)	Disco Duro
Pentium 4 o mayor	256 MB	512MB	80GB

Tabla D.1 Requerimientos de hardware

En cuanto a los requerimientos de software, la aplicación cliente puede ejecutarse prácticamente en cualquier sistema operativo que tenga instalado JAVA JRE 1.6 o mayor.

INSTALACION

Una vez cumplidos los requerimientos del sistema es necesario contar con el archivo AplicacionCliente.jar y la carpeta lib, en sistemas Windows es suficiente con ejecutar el archivo AplicacionCliente.jar

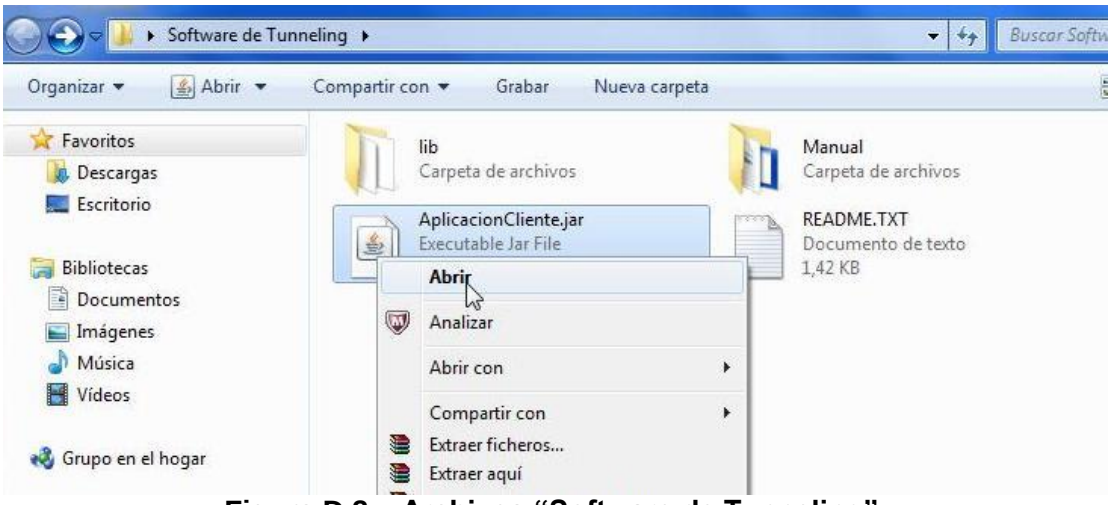


Figura D.2 – Archivos “Software de Tunneling”

En sistemas Linux como Ubuntu es necesario ejecutar una instrucción en una terminal tal como se muestra en la Figura 2.

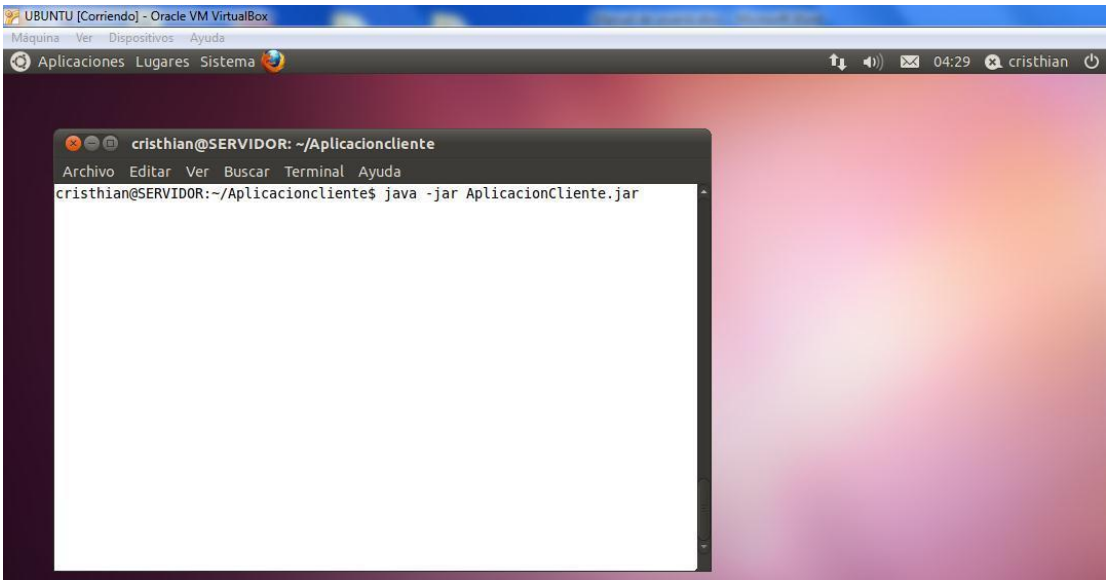


Figura D.3 – Ejecutando "AplicacionCliente" en Ubuntu

Una vez ejecutado el archivo .jar se podrá observar en el área de notificación respectiva en cada sistema operativo un ícono en forma de candado (Fig. 3 – Fig. 4) donde se irán mostrando notificaciones conforme se realiza la conexión con el servidor, el ícono cambiará de color gris a amarillo una vez que la aplicación se encuentre sincronizada con el servidor y que la comunicación sea segura.

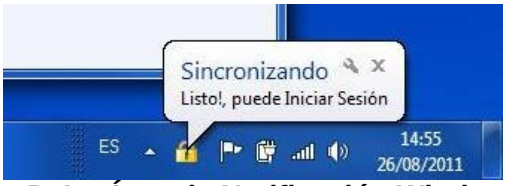


Figura D.4 – Área de Notificación Windows 7 –

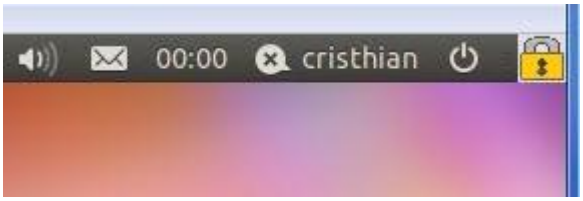


Figura D.5 – Área de Notificación Ubuntu

CREACIÓN DE CUENTA DE USUARIO

La aplicación cliente permite la creación de una cuenta de usuario, solicita para ello ciertos datos que son enviados al servidor de forma segura y posteriormente almacenados, la Fig. 5 muestra la interfaz de bienvenida de la aplicación y el link para crear la cuenta. La Fig. 6 muestra los campos a ingresar, todos son obligatorios

Figura D.6 – Crear cuenta de usuario

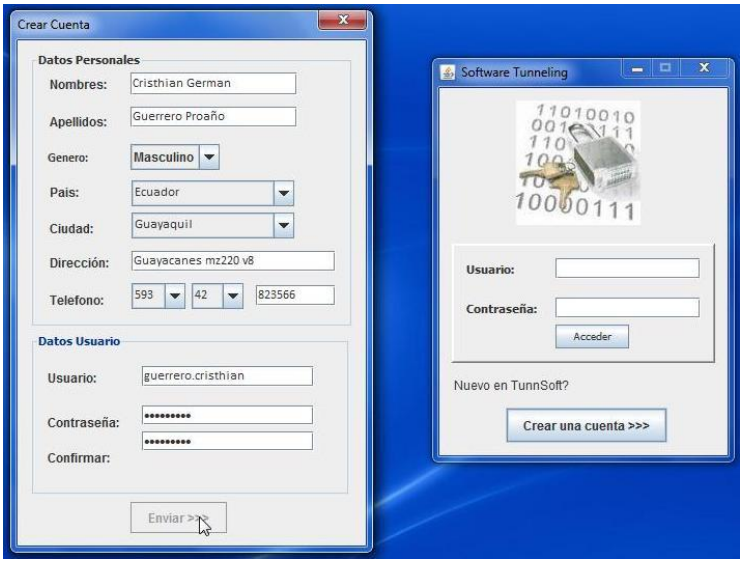


Figura D.7 – Crear cuenta de usuario

La aplicación realiza la validación de cada uno de los campos, una vez que todo se encuentre correcto al presionar el botón enviar en la parte inferior, la aplicación se conecta al servidor (Fig. 7) y si no existe una cuenta registrada con el correo electrónico que se especifica almacenará los datos del usuario en su base de datos y le permitirá posteriormente autenticarse.

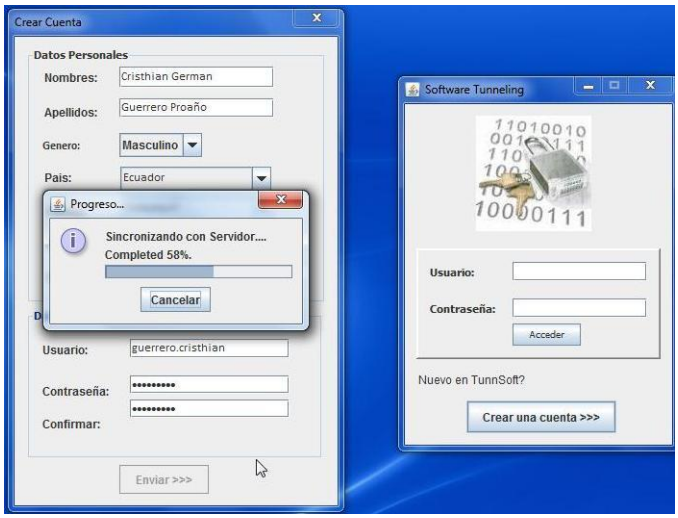


Figura D.8 – Sincronización con el servidor

AUTENTICACIÓN

Una vez creada la cuenta de usuario y almacenados los datos en el servidor, la aplicación cliente le permitirá autenticarse con el usuario y contraseña que especificó durante el registro tal como se muestra en la Fig. 8.

Figura D.9 – Autenticación en la aplicación cliente

Si el usuario y contraseña se encuentran almacenados en el servidor la aplicación cliente le mostrará una notificación como la de la Fig. 9. y le permitirá hacer uso de las funcionalidades que se mostrarán a continuación.

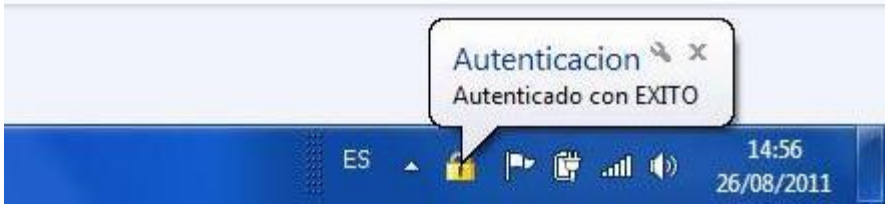


Figura D.10 – Usuario autenticado con éxito

La Fig. 10 muestra la interfaz principal de la aplicación cliente, en ella se muestra la información del usuario suministrada durante el proceso de registro, además se muestra una lista de contactos una vez añadidos los mismos, se mostrarán en gris si están desconectados y a color si se encuentra conectado como en la Fig. 11.

Figura D.11 – Interfaz principal de la aplicación

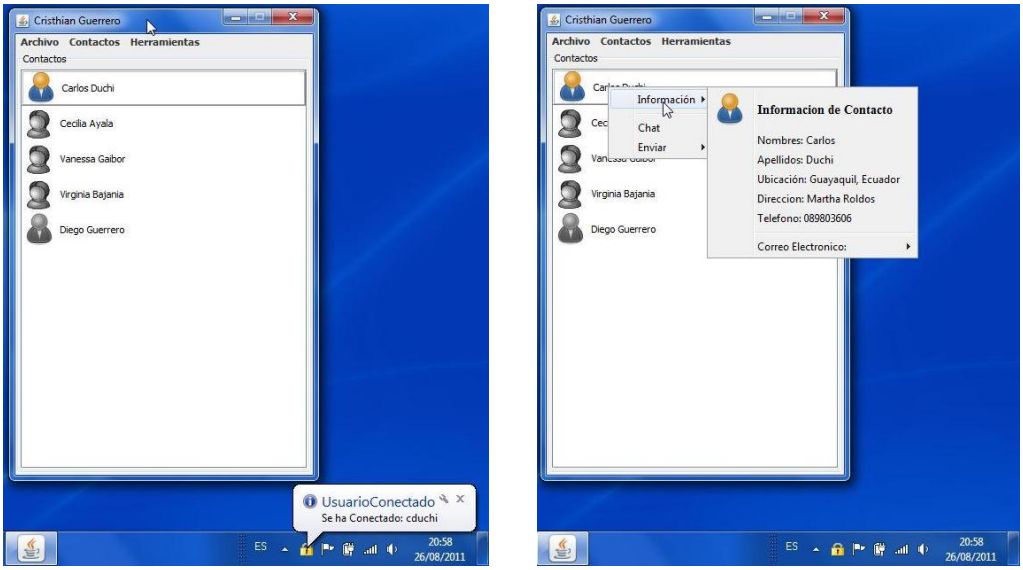


Figura D.12 – Contacto conectado

CONFIGURACIONES ADICIONALES

Una de las funcionalidades de la aplicación es permitir navegar de forma anónima, es decir que todo el tráfico viaja por un túnel protegido hacia el servidor del sistema y desde aquí se hacen todas las peticiones hacia los servidores web, para ello es necesario que una vez ejecutada la aplicación cliente se configure en el explorador de su preferencia un proxy con los parámetros que se muestran en la Fig. 11 es decir *dirección: localhost puerto: 9090***.**

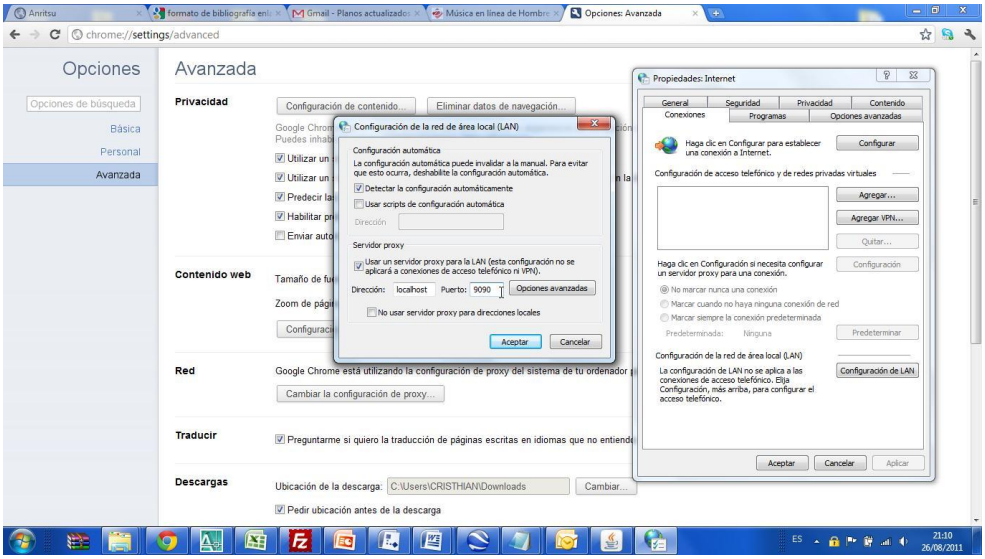


Figura D.13 – Configuración en el explorador

OTRAS FUNCIONALIDADES

Como se mostró en la Fig.11 una vez que se conecta un contacto este aparece a color y se habilita la función de chat con el mismo como se muestra en la Fig. 13, es importante mencionar que la información que se intercambie con alguno de sus contactos viajará de forma cifrada y por tanto protegida.

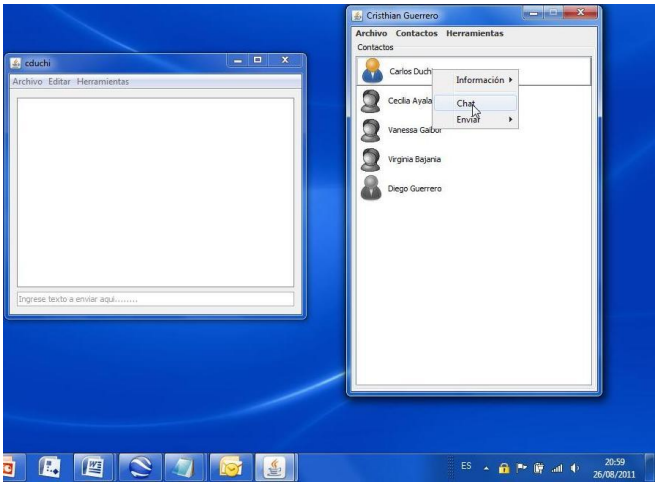


Figura D.14 – Chat habilitado

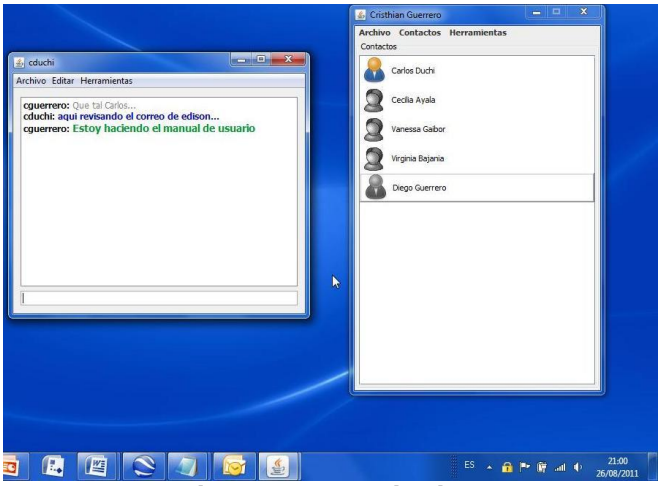


Figura D.15 – Chating

La aplicación permite la configuración de fuentes y colores para una experiencia más agradable como se muestra en la Fig.15.

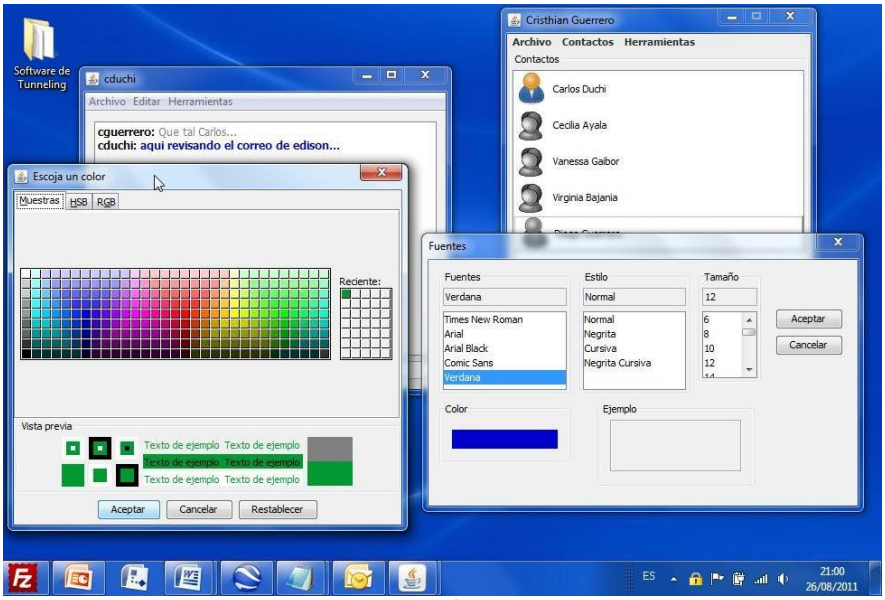


Figura 15 – Configuración de fuente para Chat

ANEXO E

ANEXO E: TABLAS

ALGORITMO	OPERACIÓN	DESCRIPCIÓN
3des-cbc	Requerido	3DES en modo CBC [1].
blowfish-cbc	Opcional	Blowfish en modo CBC
twofish256-cbc	Opcional	Twofish con una clave de 256 bits.
twofish192-cbc	Opcional	Twofish con una clave de 192 bits
twofish128-cbc	Opcional	Twofish con una clave de 128 bits
aes256-cbc	Opcional	AES con una clave de 256 bits [2].
aes192-cbc	Opcional	AES con una clave de 192 bits
aes128-cbc	Recomendado	AES con una clave de 128 bits
serpent256-cbc	Opcional	Serpent con una clave de 256 bits.
serpent192-cbc	Opcional	Serpent con una clave 192 bits
serpent128-cbc	Opcional	Serpent con una clave 128 bits
Arcfour	Opcional	ARCFOUR con una clave de 128 bits
idea-cbc	Opcional	IDEA en modo CBC
cast128-cbc	Opcional	CAST-128 en modo CBC
None	Opcional	Sin cifrado; no recomendado

Tabla E.1 - Algoritmos de cifrado - RFC 4253

ALGORITMO	OPERACIÓN	DESCRIPCIÓN
hmac-sha1	Requerido	digest length= key length=20
hmac-sha1-96	Recomendado	digest length = 12, key length = 20
hmac-md5	Opcional	digest length= key length=16
hmac-md5-96	Opcional	digest length = 12, key length = 16
None	Opcional	no MAC, no recomendado

Tabla E.2 - Algoritmos MAC - RFC 4253

MÉTODO	OPERACIÓN
diffie-hellman-group1-sha1	Requerido
diffie-hellman-group14-sha1	Requerido

Tabla E.3 - Métodos de intercambio de claves - RFC 4253

FORMATO	OPERACIÓN	DESCRIPCIÓN
ssh-dss	Requerido	Raw DSS Key
ssh-rsa	Recomendado	Raw RSA Key
pgp-sign-rsa	Opcional	OpenPGP certificates(RSA Key)
pgp-sign-dss	Opcional	OpenPGP certificates(DSS Key

Tabla E.4 - Formatos de clave pública y/o certificados – RFC 4253

[1] **FIPS46-3.** "*Data Encryption Standard (DES)".* NIST(National Institute of Standards and Technology). [Revisado el: 15 de Marzo del 2011]
[2]. **FIPS-197.** *"Advanced Encryption Standard".* NIST(National Institute of Standards and Technology). [Revisado el: 15 de Marzo del 2011]

Retorna	Método	Parámetro Entrada
Cipher	**getInstance**	(String Algoritmo)
Implementa la transformación especificada. Recorre una lista de proveedores registrados, empezando con el preferido. El objeto Cipher encapsula la implementación de este proveedor que soporta el algoritmo especificado.		
Void	**init**	(Int modo_oper, Key clave)
Inicializa Cipher con una clave en uno de los siguientes modos de operación: cifrar, descifrar, envolver clave o desenvolver clave, dependiente de un valor de modo_oper.		
Byte []	**doFinal**	(Byte [] entrada)
Cifra o descifra datos en una operación simple, o de múltiples partes. El dato es cifrado o descifrado dependiendo de cómo el Cipher fue inicializado.		

Tabla E.5 – Principales métodos Clase Cipher

Retorna	Método	Parámetro Entrada
KeyPairGenerator	**getInstance**	(String Algoritmo)
Retorna un Objeto KeyPairGenerator que genera un par de claves del algoritmo especificado		
Void	**Initialize**	(Int Tamaño_Clave)
Inicializa el generador de claves pares para un tamaño de clave especificado		
KeyPair	**genKeyPair**	()
Genera un par de claves. Una publica y una privada.		

Tabla E.6 – Principales métodos Clase keyPairGenerator

Retorna	Método	Parámetro Entrada
PrivateKey	getPrivate	()
Devuelve una referencia al componente de la clave privada de un par de claves.		
PublicKey	getPublic	()
Devuelve una referencia al componente de la clave pública de un par de claves		

Tabla E.7 – Principales métodos Clase keyPair

PROVEEDORES PRESENTES EN JAVA				
SunPKCS11	SunJCE	SunSASL	SunJSSE	SunPCSC
SUN	SunJGSS	XMLDSig	SunMSCAPI	RsaSign

Tabla E.8 – Proveedores de Seguridad en Java

Parámetros	Mediciones										Prom.
Ping (ms)	36	36	37	41	38	40	70	38	33	46	41.5
Descarga (Mb/s)	0,23	0,15	0,34	0,32	0,45	0,35	0,18	0,36	0,39	0,41	0,32
Carga (Mb/s)	0,12	0,12	0,11	0,12	0,12	0,12	0,11	0,12	0,12	0,12	0,118

Tabla E.9 – Mediciones conexión directa LAN

Parámetros	Mediciones										Prom.
Ping (ms)	42	38	64	41	39	38	70	47	38	40	45.7
Descarga (Mb/s)	0,42	0,45	0,33	0,45	0,35	0,42	0,25	0,49	0,41	0,49	0,41
Carga (Mb/s)	0,12	0,12	0,11	0,12	0,12	0,12	0,10	0,12	0,12	0,12	0,117

Tabla E.10 – Mediciones con aplicación cliente LAN

Parámetros	Mediciones										Prom.
Ping (ms)	170	114	125	103	100	167	171	118	199	170	144
Descarga (Mb/s)	0,38	0,44	0,69	0,30	0,40	0,36	0,36	0,16	0,25	0,15	0,35
Carga (Mb/s)	0,38	0,38	0,40	0,37	0,36	0,34	0,37	0,36	0,37	0,38	0,37

Tabla E.11 – Mediciones con conexión directa WAN

Parámetros	Mediciones										Prom.
Ping (ms)	135	189	183	167	244	264	244	184	242	264	212
Descarga (Mb/s)	0,11	0,10	0,08	0,06	0,09	0,08	0,04	0,04	0,05	0,11	0,08
Carga (Mb/s)	0,10	0,09	0,10	0,05	0,05	0,08	0,11	0,04	0,08	0,07	0,08

Tabla E.12 – Mediciones con aplicación cliente WAN

Equipo	Tipo	Sistema Operativo	Procesador	RAM
ORDENADOR 1	Cliente	Windows XP SP3	Intel Celeron 1.86Ghz	1GB
ORDENADOR 2	Cliente	Mac OS X	Intel Core i5 2.4Ghz	4GB
ORDENADOR 3	Cliente	Windows 7	Intel Core 2.4Ghz	4GB
ORDENADOR 4	Cliente	Ubuntu 11.05	Intel T2050 1.6Ghz	2GB
ORDENADOR 5	Cliente	Windows XP SP3	Intel T2050 1.6Ghz	2GB
ORDENADOR 6	Cliente	Back Track 5	Intel T2050 1.6Ghz	2GB
ORDENADOR 7	Cliente	Mac OSX 10.6	Intel Core i5 2.5Ghz	4GB
ORDENADOR 8	Servidor	Ubuntu 11.05	Intel Pentium 4 2.8Ghz	1GB

Tabla E.13 – Ordenadores utilizados para pruebas

Característica	Especificación
Interfaces	4 10/100Mbps RJ45 LAN 1 10/100Mbps RJ45 WAN 1 11/54Mbps WLAN
Memoria SDRAM	64MB
Estándares Inalámbricos	IEEE802.11g, IEEE 802.11b
Estándares WAN	IP Dinámica/IP Estática/ PPPoE PPTP/L2TP/BigPond
Estándares LAN	IEEE 802.3 y IEEE 802.3u
Características especiales	Servidores virtuales, DMZ, DNS dinámico y enrutamiento estático

Tabla E.14 – Características enrutador TPLINK WR542G

Parámetro	Mediciones										Prom.
Ping (ms)	413	410	410	409	408	408	413	407	415	408	410,1
Descarga (Mb/s)	0,522	0,515	0,492	0,494	0,485	0.485	0,487	0,472	0,501	0,612	0,5089
Carga (Mb/s)	0,114	0,113	0,99	0,107	0,114	0,111	0,78	0,83	0,114	0,114	0,3387

Tabla E.15 – Mediciones Conexión directa LAN (Interauta)

Parámetro	Mediciones										Prom.
Ping (ms)	201	201	201	196	202	202	196	197	195	202	199,3
Descarga (Mb/s)	0,52	0,554	0,504	0,486	0,643	0,515	0,487	0,544	0,488	0,44	0,5181
Carga	0,105	0,11	0,107	0,104	0,104	0,105	0,107	0,056	0,101	0,103	0,1002

Tabla E.16 – Mediciones con Aplicación Cliente LAN (Interauta)

Parámetro	Mediciones										Prom.
Ping (ms)	409	407	409	409	409	409	408	409	410	410	408,9
Descarga (Mb/s)	0,662	0,524	0,484	0,654	0,485	0,95	0,485	0,485	0,485	0,566	0,578
Carga (Mb/s)	0,113	0,11	0,113	0,114	0,114	0,114	0,114	0,113	0,114	0,114	0,1133

Tabla E.17 – Mediciones Conexión directa WAN (Interauta)

Parámetro	Mediciones										Prom.
Ping (ms)	201	201	201	201	201	195	195	195	195	196	198,1
Descarga (Mb/s)	0,572	0,444	0,838	0,665	0,794	0,58	0,58	0,433	0,504	0,397	0,5817
Carga (Mb/s)	0,097	0.106	0,104	0,108	0,102	0,1	0,1	0,101	0,105	0,104	0,1023

Tabla E.18 – Mediciones con Aplicación Cliente WAN (Interauta)

ANEXO F

ANEXO F: FIGURAS

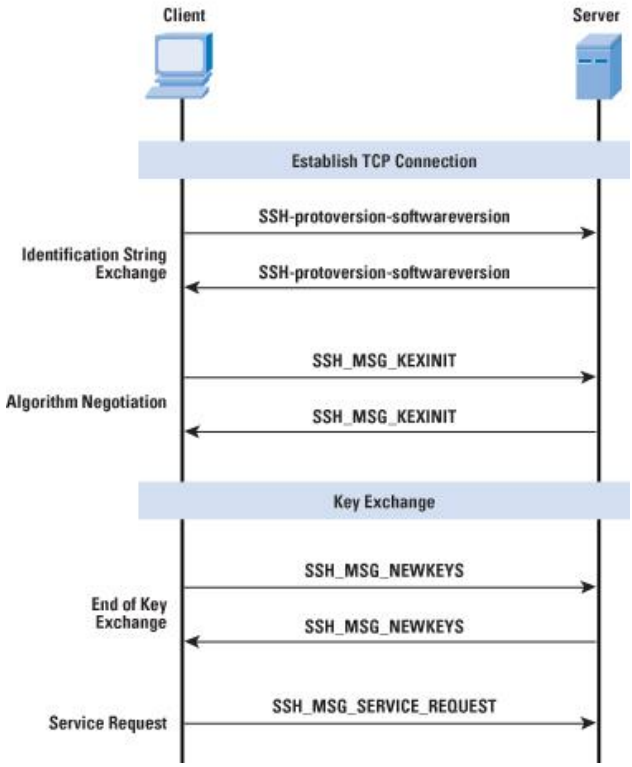


Figura F.1 - Inicialización Protocolo de Capa de Transporte SSH
Fuente: Tomado de Internet Protocol Forum [1]

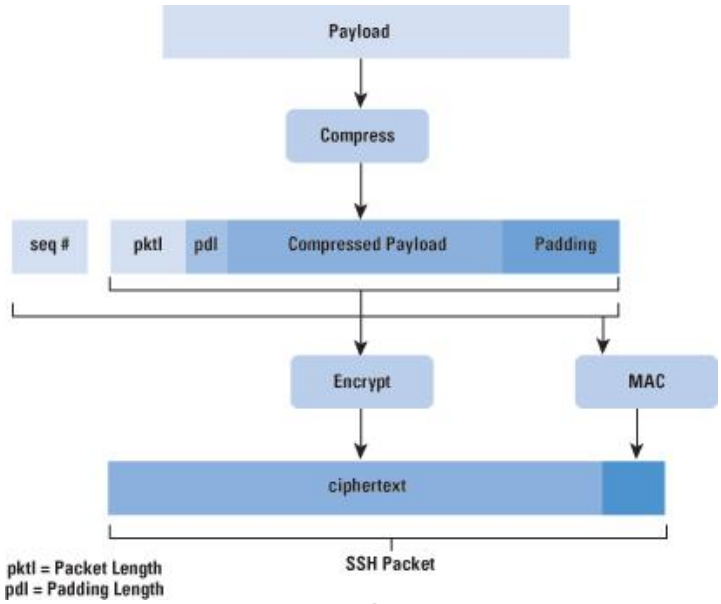


Figura F.2 - Formación del paquete SSH
Fuente: Tomado de Internet Protocol Forum [1]

[1]. **Stallings, William.** "*SSH*".The Internet Protocol Forum. [Revisado el: 10 de Marzo del 2011].

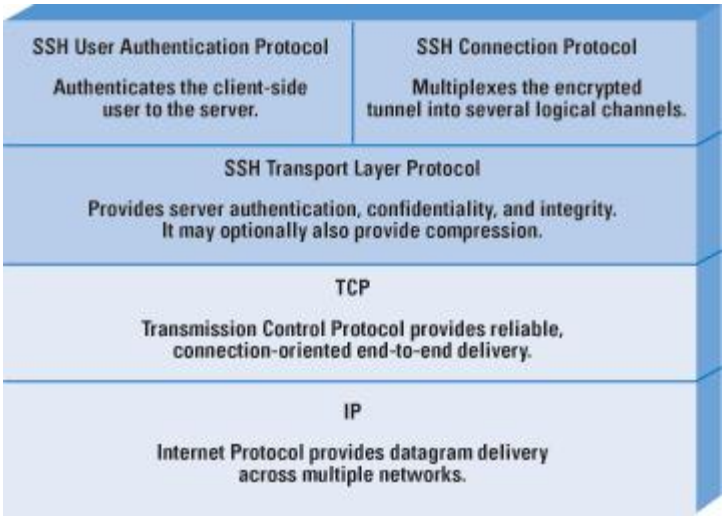


Figura F.3 – Pila del Protocolo SSH
Fuente: Tomado de Internet Protocol Forum [1]

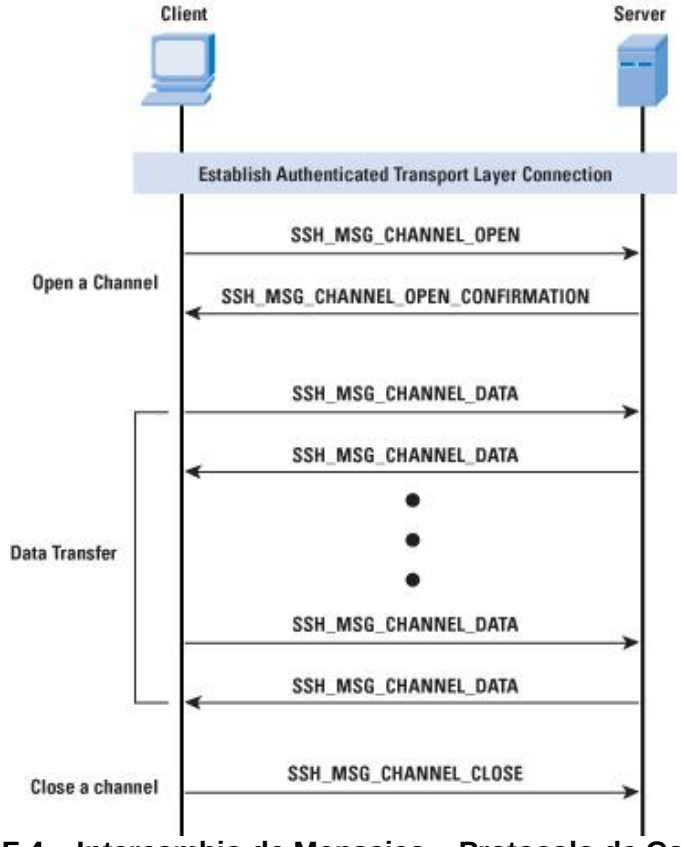


Figura F.4 – Intercambio de Mensajes – Protocolo de Conexión
Fuente: Tomado de Internet Protocol Forum [1]

[1]. **Stallings, William.** "*SSH*".The Internet Protocol Forum. [Revisado el: 10 de Marzo del 2011].

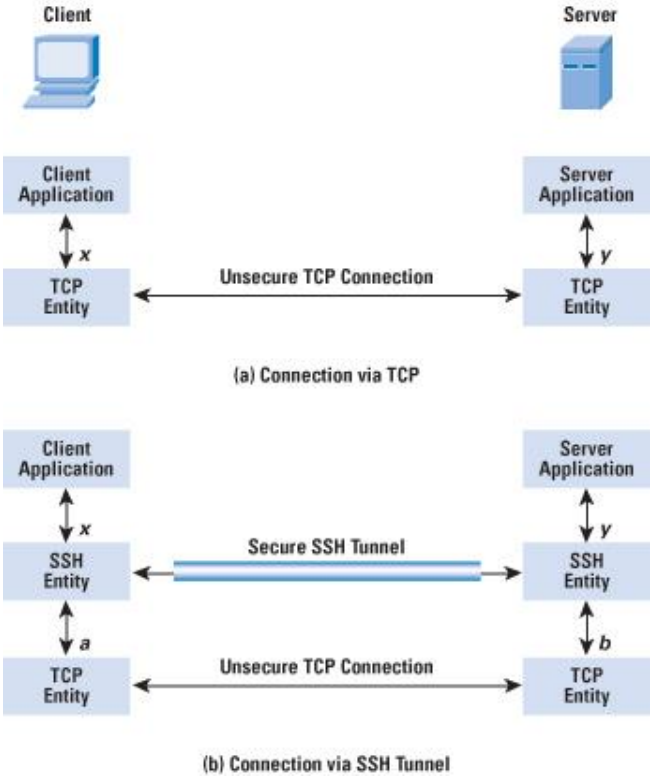


Figura F.5 –Tunneling SSH– Protocolo de Conexión

Fuente: Tomado de Internet Protocol Forum [1]

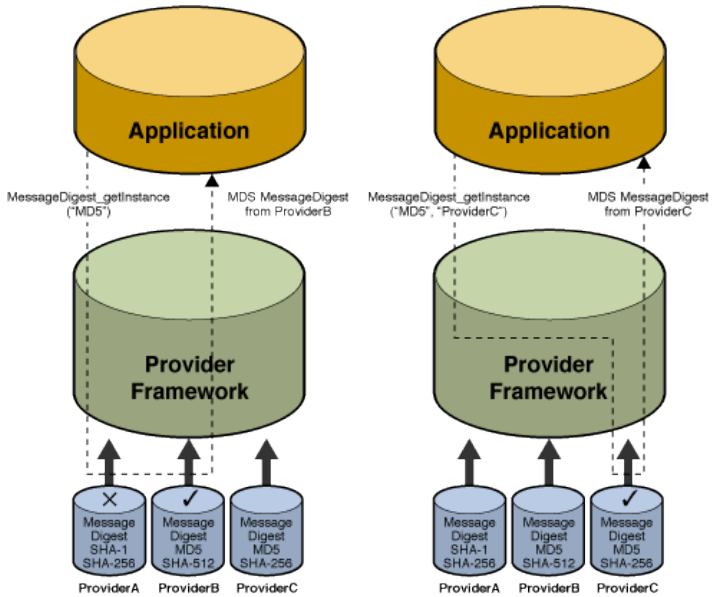


Figura F.6 –Interacción Proveedor-Aplicación en JAVA

Fuente: Tomado de Internet Protocol Forum [2]

[1]. **Stallings, William.** "*SSH*".The Internet Protocol Forum. [Revisado el: 10 de Marzo del 2011].
[2]. **Microsystems, Sun.** "*JavaMail Guide for Service Providers*". Oracle.com. [Revisado el: 15 de Mayo del 2011].

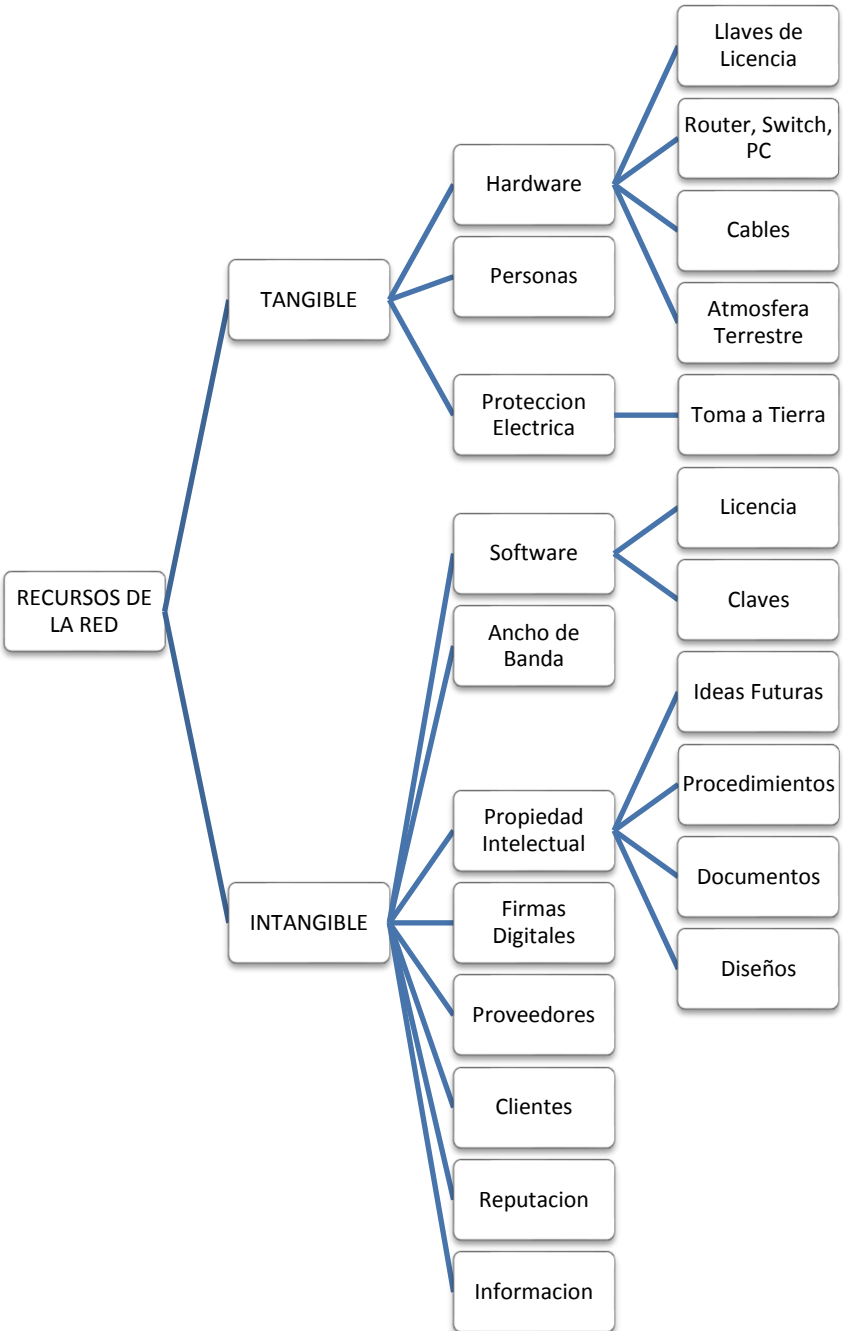


Figura F.7 – Recursos de Red

Fuente: Seminario de Graduación Seguridad en Redes 2010/Ing. Ignacio Marín

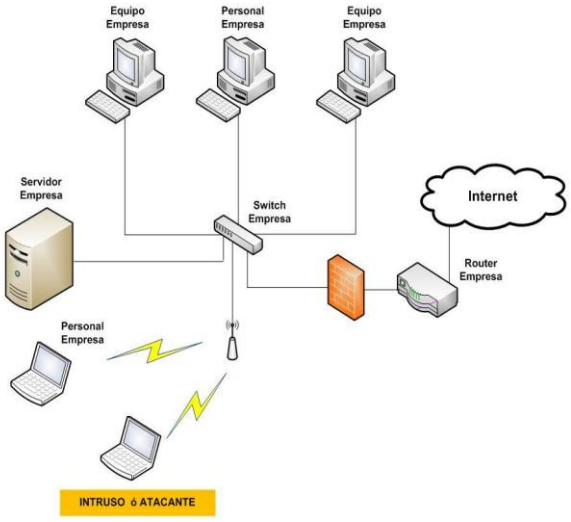


Figura F.8 - Ataque de acceso sobre una WLAN

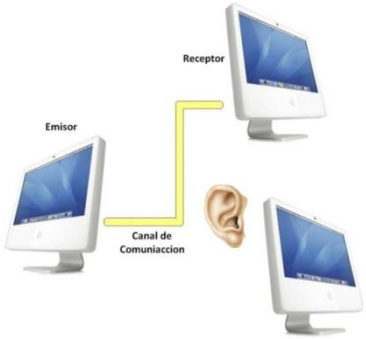


Figura F.9 - Ataque de Intersección

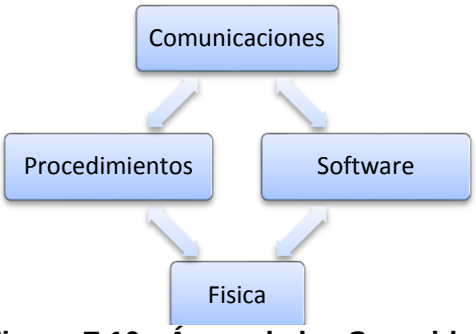


Figura F.10 - Áreas de las Seguridad

Fuente: Seminario de Graduación Seguridad en Redes 2010/Ing. Ignacio Marín

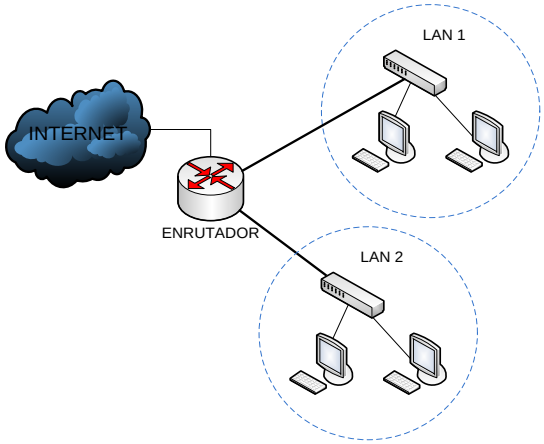


Figura F.11 – Función del Enrutador

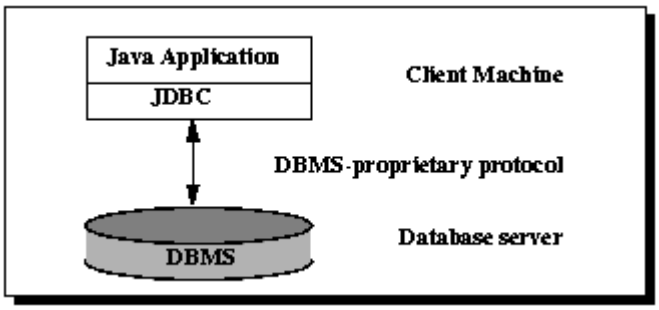


Figura F.12 - Arquitectura de acceso a Base de Datos
Fuente: Tomado de [3]

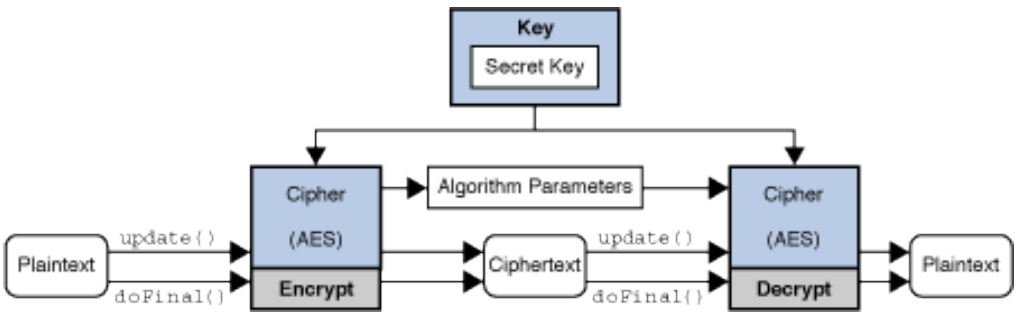


Figura F.13 – Sistema Criptográfico con Cipher
Fuente: Oracle - Java Arquitectura Criptográfica

[3]. **JDBC, Oracle. "***JDBS Introduccion***".** ORACLE. [Revisado el: 20 de Mayo del 2011].

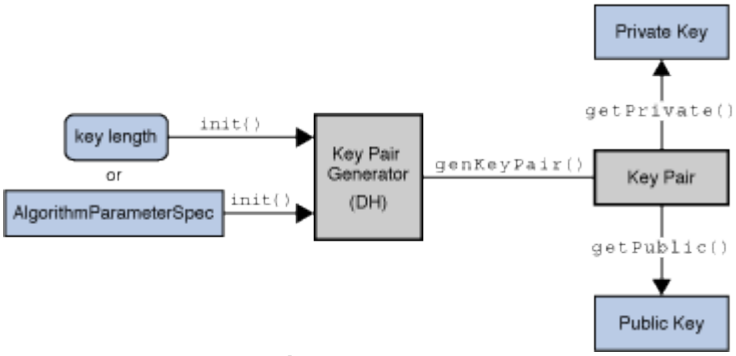


Figura F.14 - Generación de Claves con keyPairGenerator
Fuente: Oracle - Java Arquitectura Criptográfica

Figura F.15 - Modelo TCP/IP

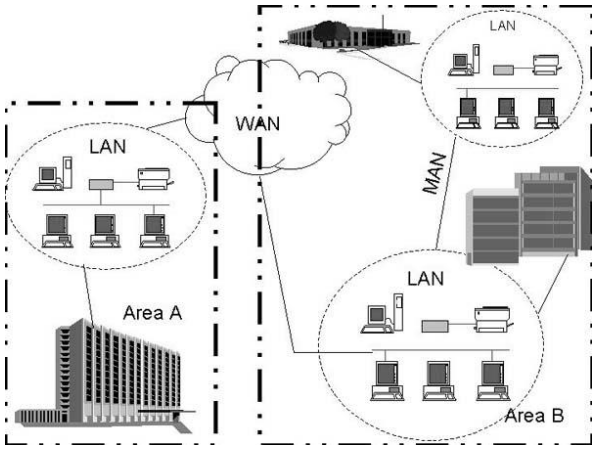


Figura F.16 - Redes por Cobertura (LAN, MAN y WAN)
Fuente: Tomado de redes-sociedad-miblog.blogspot.com [4].

[4]. **Perez, Alejandra.** "*La Red y la Sociedad*". redes-sociedad-miblog.[Revisado el: 25 de Marzo del 2011].

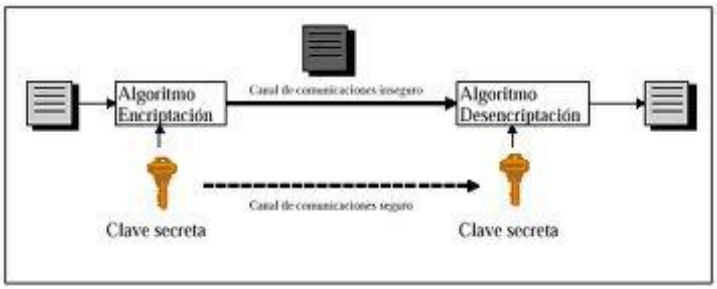


Figura F.17 - Algoritmo criptográfico simétrico

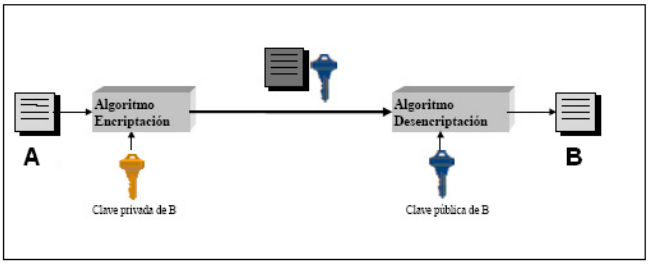


Figura F.18 - Algoritmo criptográfico asimétrico

Fuente: Pagina web wikilearning

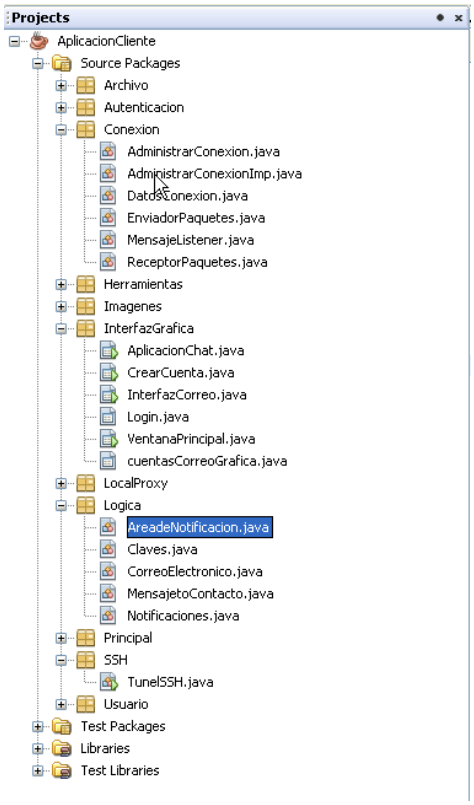


Figura F.19 – Vista modular de la Aplicación Cliente en Netbeans

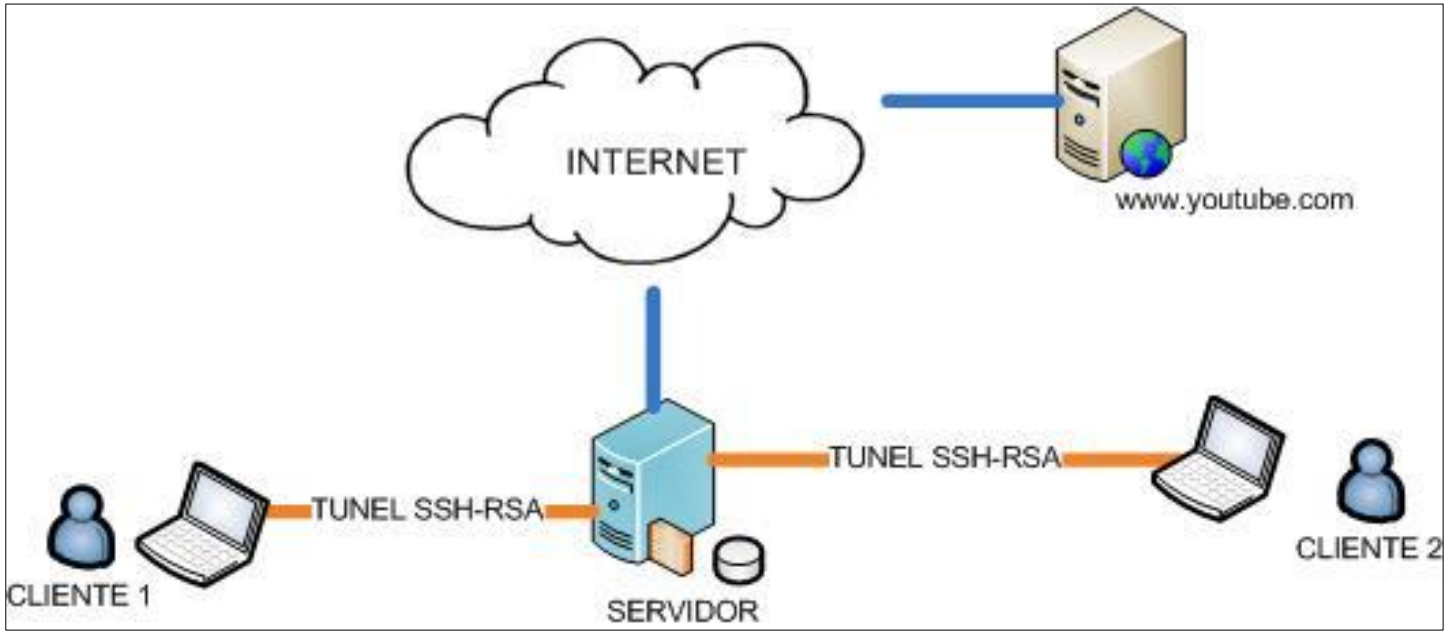


Figura F.20 - Arquitectura del Sistema

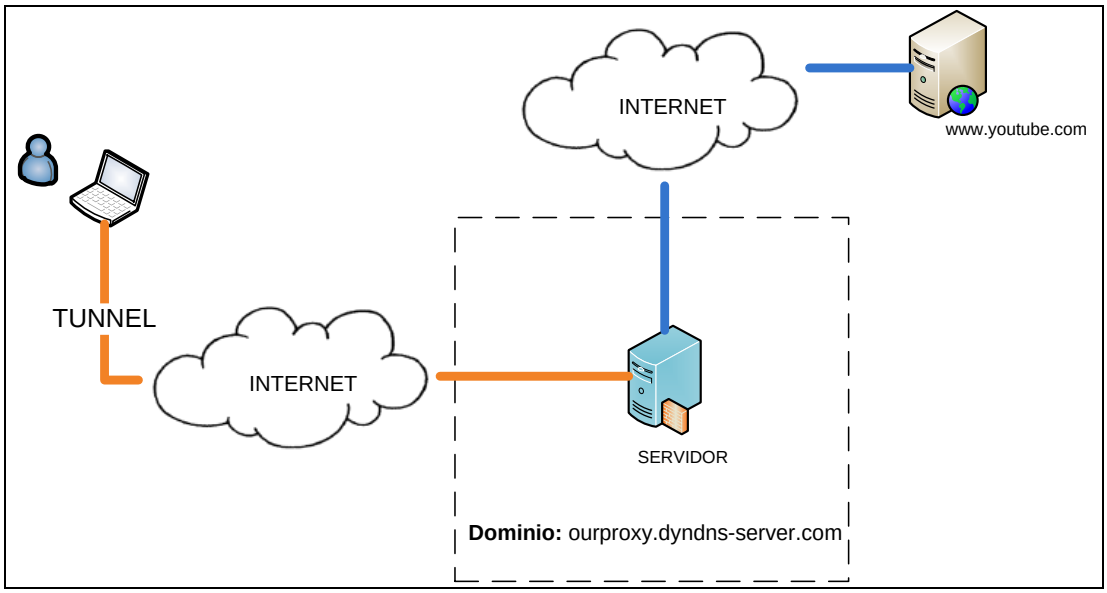


Figura F.21 – Usuario externo del sistema

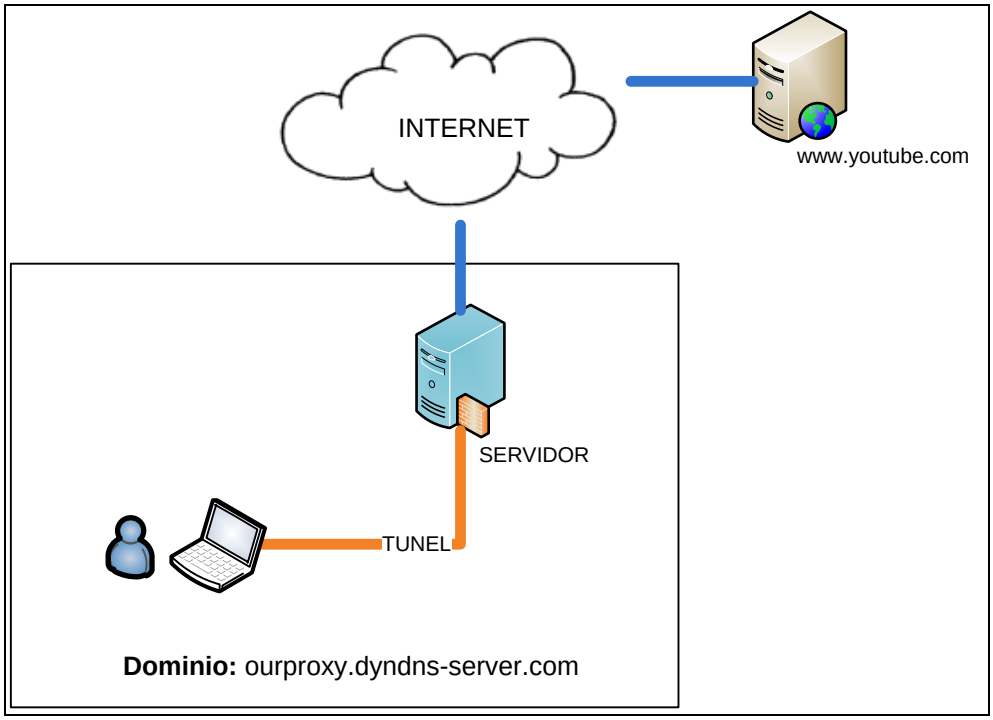


Figura F.22- Usuario interno del sistema

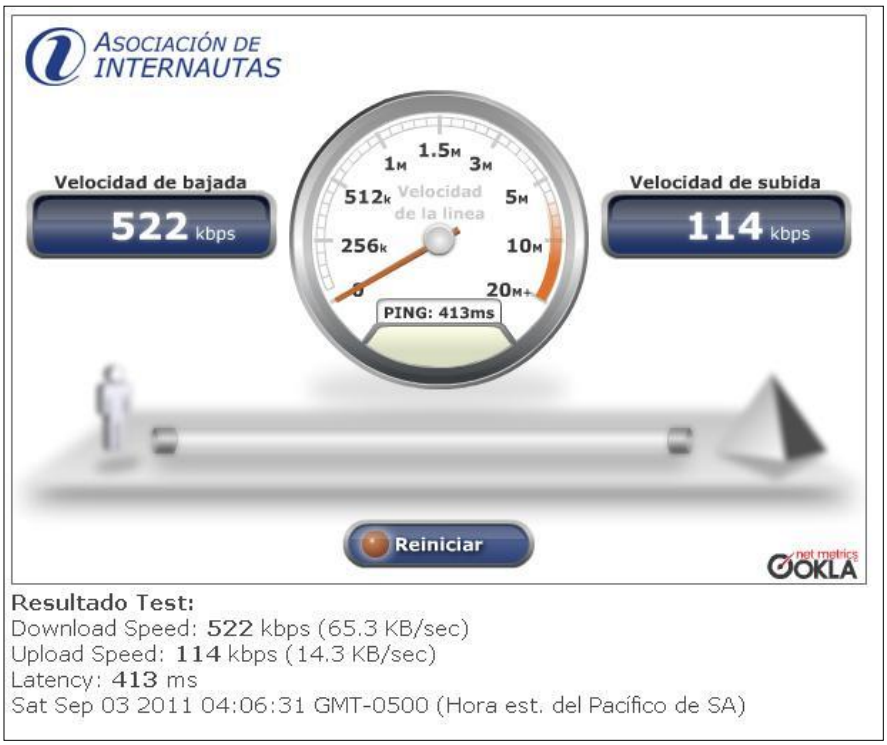


Figura F.23- Pagina web Internautas (Medición de ancho de banda)

ANEXO G

ANEXO G: PRUEBAS

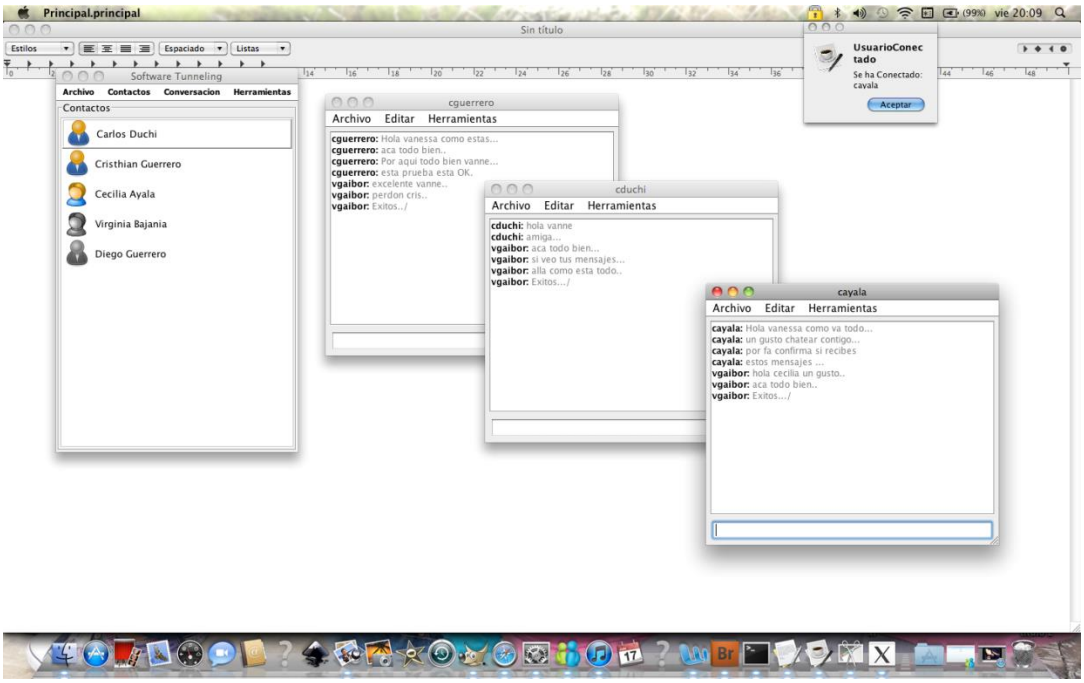


Figura G.1 – Portabilidad en MAC

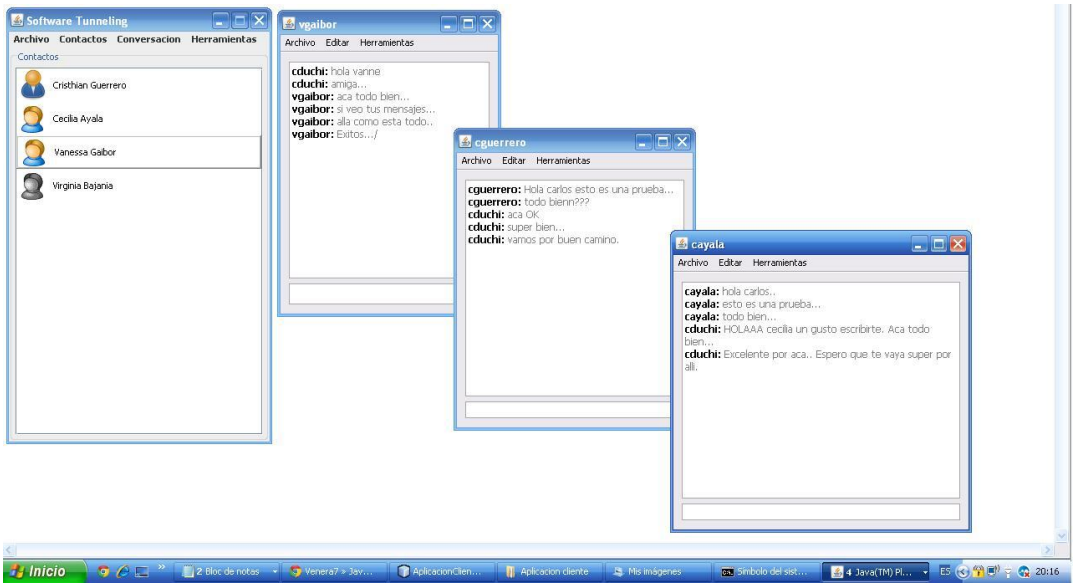


Figura G.2 – Portabilidad en Windows XP

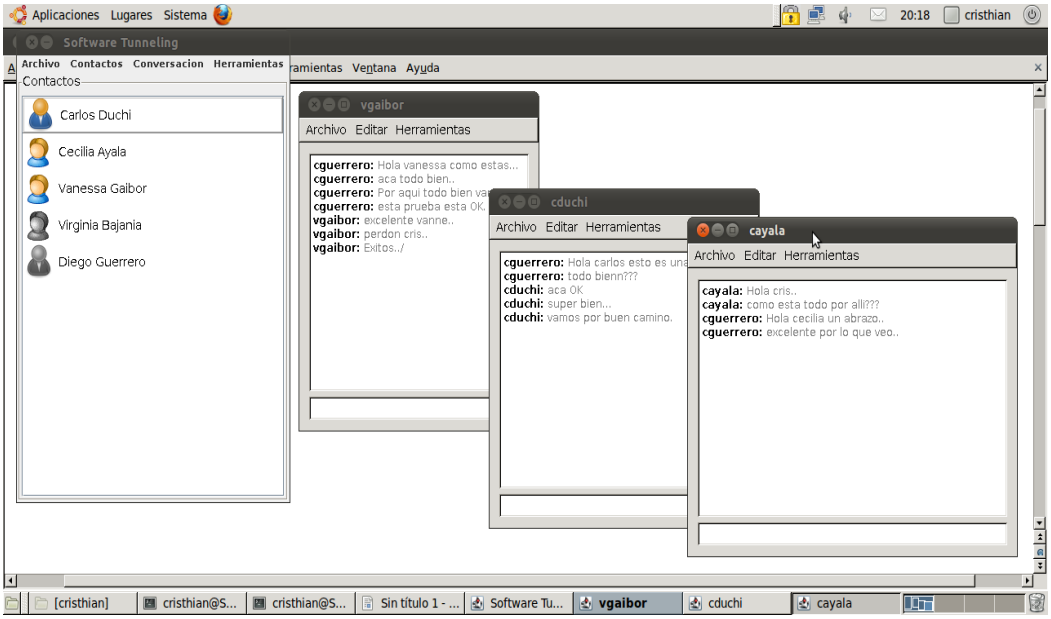


Figura G.3 – Portabilidad en Ubuntu

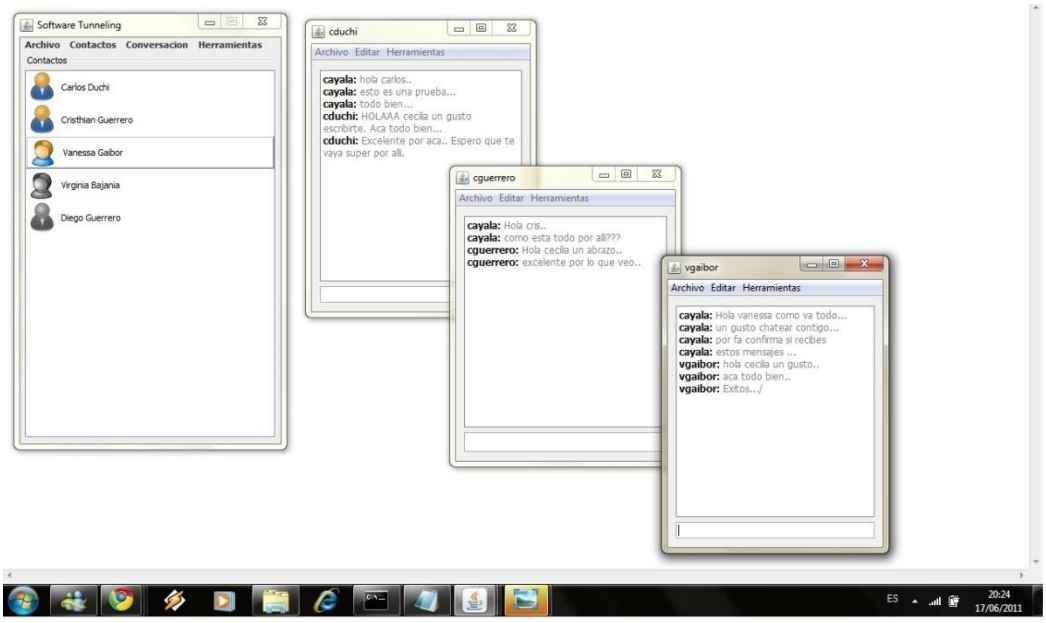


Figura G.4 – Portabilidad en Windows 7

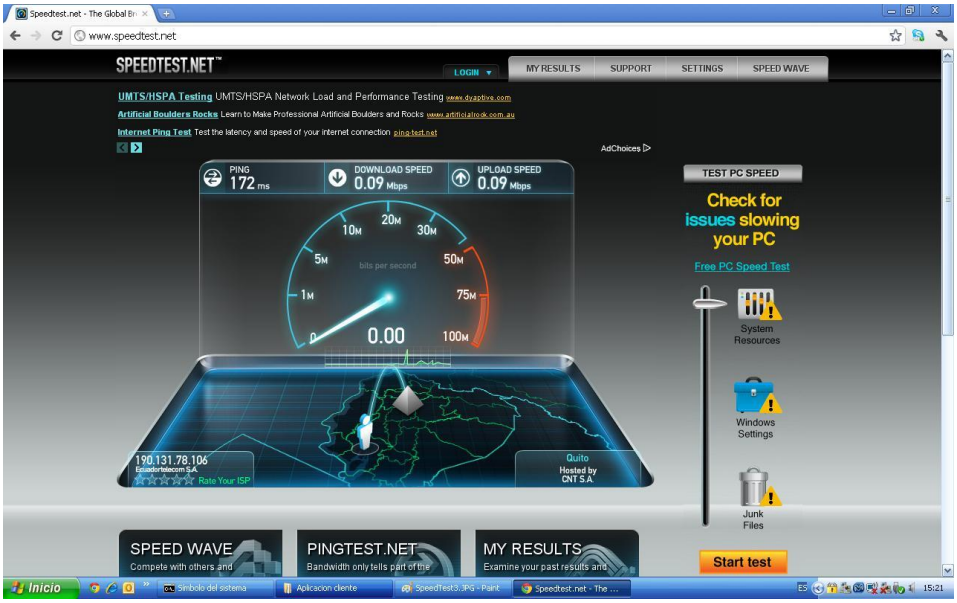


Figura G.5 – Herramienta SpeedTest

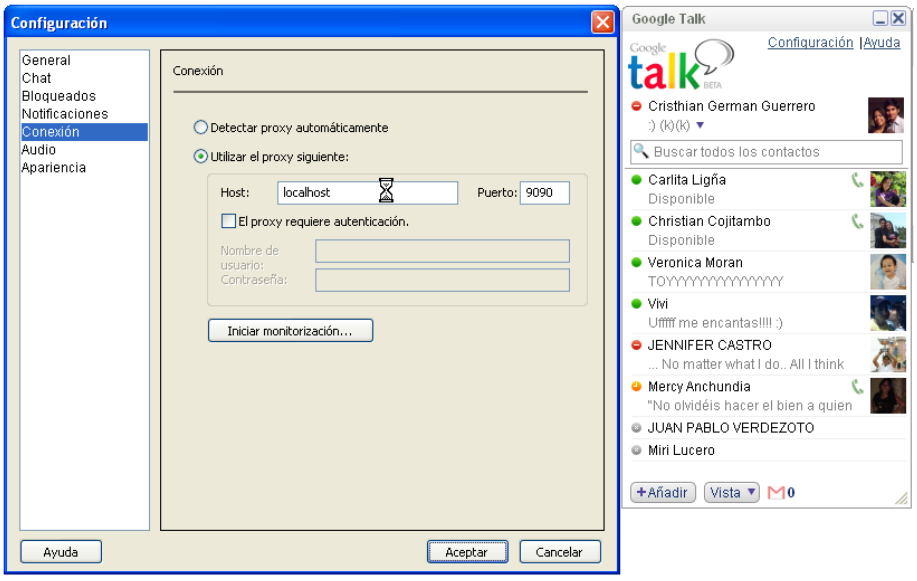


Figura G.6 - Configuración del proxy en Google Talk

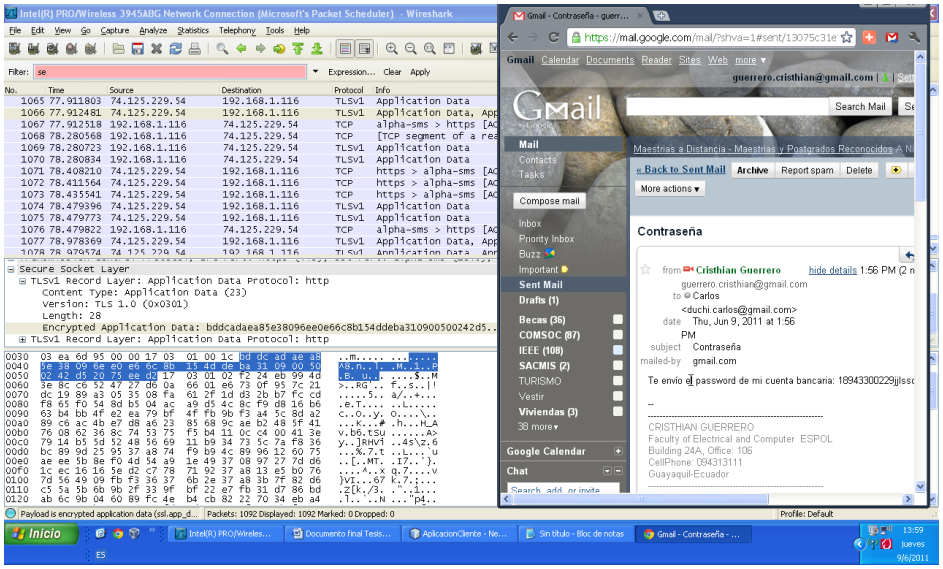


Figura G.7 – Captura Wireshark Gmail

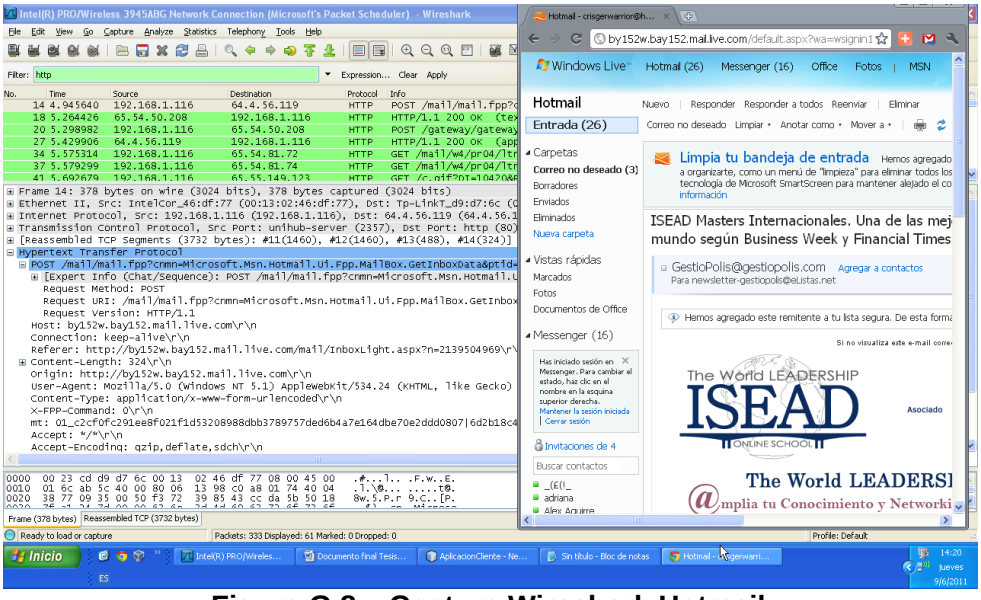


Figura G.8 – Captura Wireshark Hotmail

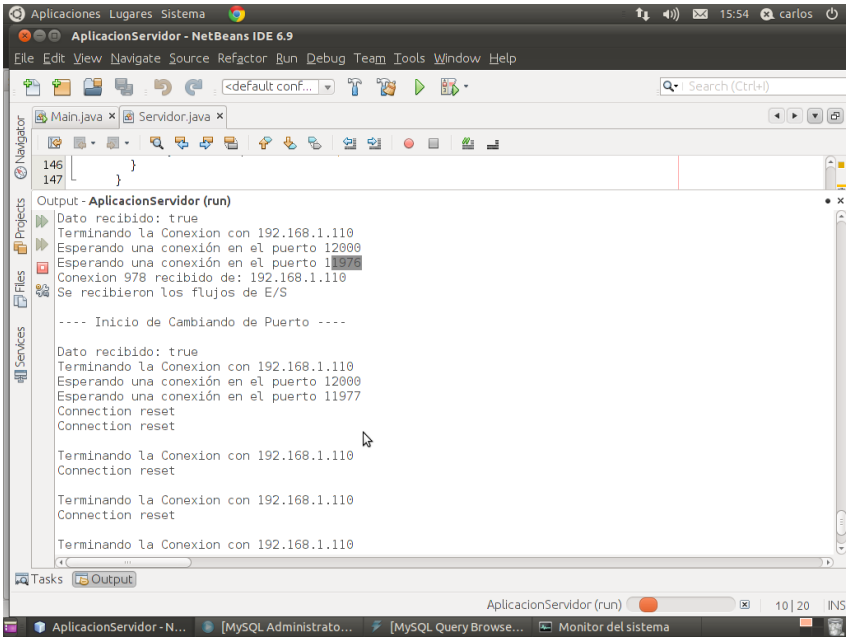


Figura G.9 – Capacidad de conexiones

Figura G.10 – Capacidad: Rendimiento del servidor

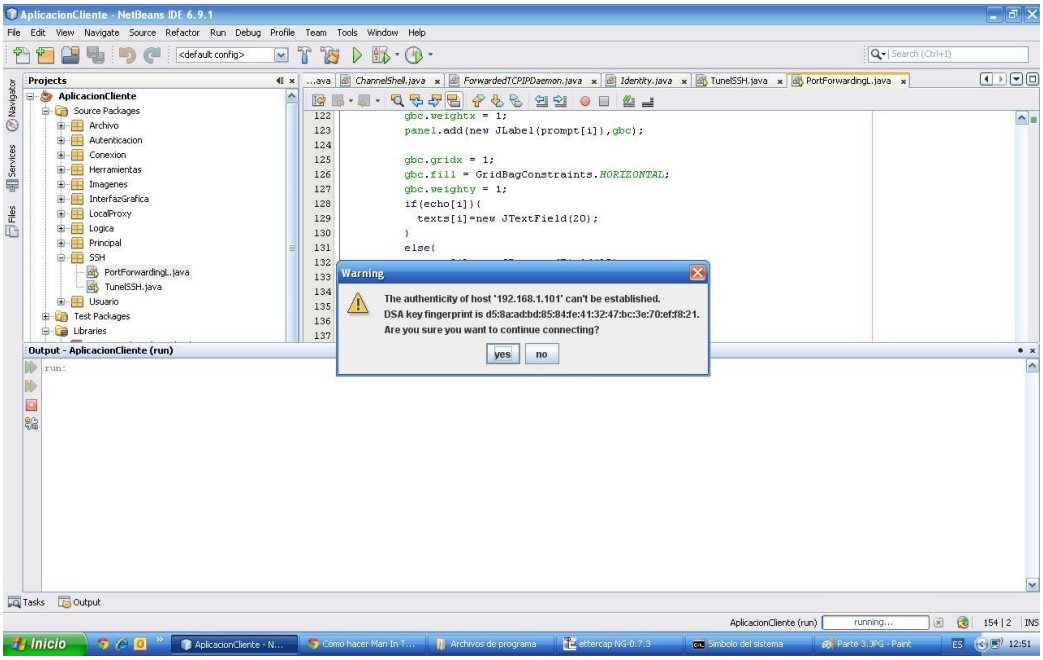


Figura G.11 – Advertencia del servidor a conectar

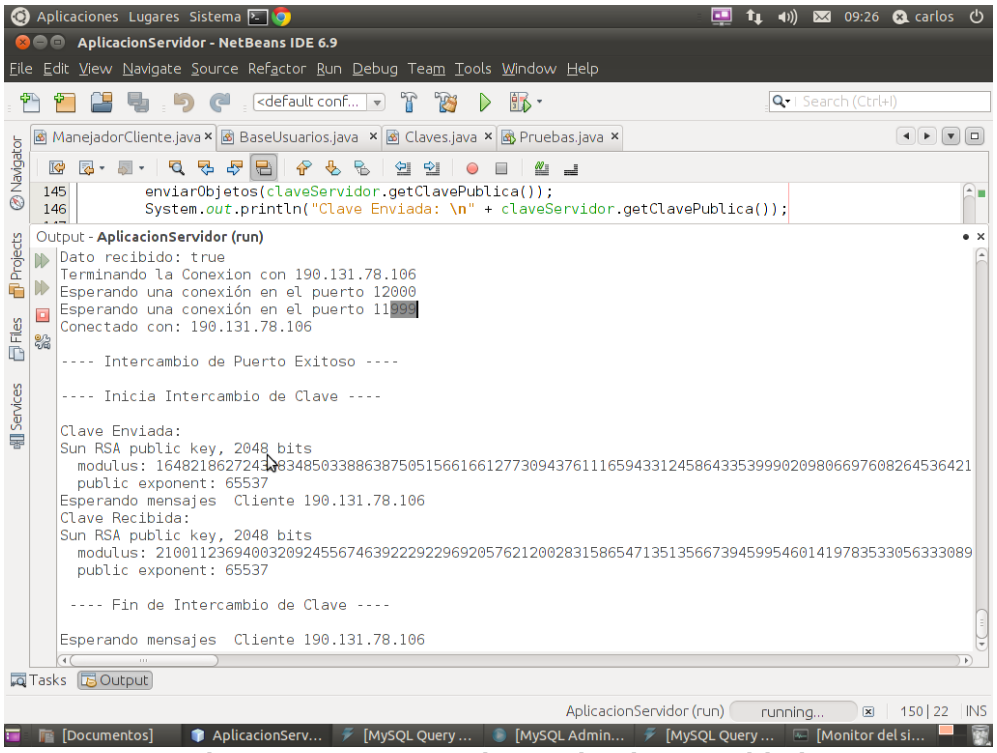


Figura G.12 – Segunda prueba de capacidad

B I B L I O G R A F Í A Y R E F E R E N C I A S

[1]. **Polanco, Leon,** *"Internet",* Wikipedia Español, 5 de Abril del 2011. <http://es.wikipedia.org/wiki/Internet>, [Revisado el: 11 de Mayo del 2011]

[2]. **Asociación para la Investigación de Medios de Comunicación,** "*Encuesta a usuarios de Internet",* SERSA*,* España Febrero del 2011, pag. 57, <http://download.aimc.es/aimc/navred2010/macro2010.pdf>, [Revisado el: 12 del Mayo del 2011]

[3]. **Seguridad,** "*Los delitos informáticos crecieron",*ElComercio.com, Ecuador 20 de Diciembre del 2010, <http://www.elcomercio.com.ec/seguridad/delitos-informaticos-crecieron_0_393560676.html>, [Revisado el: 19 de Marzo del 2011]

[4]. **Risk, Verizon y USSS.** "*2010 Data Breach Investigations Report",* Estados Unidos 2010, pags 20-37, <http://www.verizonbusiness.com/resources/reports /rp_2010-data-breach-report_en_xg.pdf>, [Revisado el: 15 de Mayo del 2011]

[5]. **Dorgoigne, José y Ateline, Philippe.** "*Redes Informáticas: Conceptos Fundamentales",* ENI, España-Barcelona 2006, Págs. 10-12, [Revisado el: 15 del Mayo del 2011]

[6]. **Cancel, Edwin,** *"Introduccion a las Microcomputadoras?"*, AngelFire.com, 9 de Mayo 2003, <http://www.angelfire.com/pro/edcanj/Micro.htm >, [Revisado el: 15 de Mayo del 2011]

[7]. **EURAM, Informatica,** "*¿Qué es un Sistema operativo?*", Paginas Verdes, Edicion 144, Nicaragua 2 de Agosto 2006, <http://www.euram.com.ni/pverdes/verdes_informatica/informatica_al_dia/que_es_un_so_144.htm>, [Revisado el: 15 de Mayo del 2011]

[8]. **Cassiusfrance,** "*Sobre UBUNTU*", Ubuntu-es, 12 de Julio 2011, <http://doc.ubuntu-es.org/Sobre_Ubuntu>, [Revisado el: 20 de Abril del 2011]

[9]. **Segovia, Sergio,** "*Familia de Protocolos de Internet",* Wikipedia.. 3 de Mayo del 2011, <http://es.wikipedia.org/wiki/TCP_IP>, [Revisado el: 15 de Mayo de 2011]

[10].**Redes y Telecomunicaciones,** "*TCP/IP*", textoscientificos.com, 2 de Octubre del 2006, <http://www.textoscientificos.com/redes/tcp-ip>, [Revisado el: 18 de Mayo del 2011]

[11]. **Maiwald, Eric.** "*Fundamentos de Seguridad en Redes",* Mc GRAW HILL, Mexico 2003, Págs. 4-90,261, [Revisado el: 18 de Mayo del 2011],

[12]. **Emilio, Durón,** "*Protocolo (informática)*", Wikipedia, 27 de Abril del 2011. <http://es.wikipedia.org/wiki/Protocolo_(inform%C3%A1tica)>, [Revisado el: 19 de Mayo del 2011]

[13]. **Ylonen, Tatu y Lonvick, Chris,** "*RFC 4251",* eitf.org, Enero del 2006, <http://tools.ietf.org/html/rfc4251#section-4.5>, [Revisado el: 19 de Mayo del 2011]

[14] **Roger S. Pressman,** *"Ingenieria de Software-Un enfoque práctico"*, McGRAWHILL, Madrid-España 2002, Pags. 181-234, [Revisado el: 23 de Agosto del 2011]

[15]. **Dierks T., Rescorla E,** *"The Transport Layer Security (TLS) Protocol Version 1.1"*, IETF, Abril 2006, <http://www.ietf.org/rfc/rfc4346.txt>, [Revisado el: 5 de Marzo del 2011]