



Centro Universitario de la Defensa en la Escuela Naval Militar

TRABAJO FIN DE GRADO

*Diseño de un sistema de mensajería instantánea para su uso
como red de mando cifrada*

Grado en Ingeniería Mecánica

ALUMNO: Roberto Paniagua Ramón
DIRECTOR: Rafael Asorey Cacheda
CURSO ACADÉMICO: 2014-2015

Universida_{de}Vigo



Centro Universitario de la Defensa en la Escuela Naval Militar

TRABAJO FIN DE GRADO

*Diseño de un sistema de mensajería instantánea para su uso
como red de mando cifrada*

Grado en Ingeniería Mecánica
Intensificación en Tecnología Naval
Cuerpo General

UniversidadeVigo

RESUMEN

En este Trabajo Fin de Grado (TFG) se analizan las principales tecnologías de mensajería instantánea y su aplicabilidad en las redes de comunicación de la Armada. El análisis se complementa con la instalación y configuración de un servicio de mensajería instantánea en el cuartel Marqués de la Victoria.

En la primera parte de esta memoria se explica el funcionamiento de los sistemas de mensajería instantánea. Para ello, se introduce la terminología previa necesaria para abordar el análisis de estas tecnologías.

Una de las cuestiones fundamentales en los sistemas de mensajería instantánea es la seguridad. Por este motivo, la mayor parte de estos conceptos introductorios están relacionados con el cifrado de mensajes, la autenticación de usuarios y el establecimiento de conexiones seguras. También se introducen varios de los protocolos empleados, diferenciando entre protocolos abiertos y propietarios.

A partir de los conceptos iniciales, se realiza un estudio del estado del arte sobre el uso de los sistemas de mensajería instantánea y su evolución hasta la actualidad. Se analizan los servicios y aplicaciones de mensajería instantánea bien por su disponibilidad en múltiples plataformas, o bien por usar un protocolo con características especiales de seguridad.

En base al estudio de los sistemas de mensajería instantánea se analiza cómo implementar un sistema de mensajería instantánea en las redes de la Armada. Como resultado, se monta un prototipo de prueba en la red inalámbrica del cuartel Marqués de la Victoria.

Por último, se presentan las conclusiones así como unas posibles líneas de acción futuras.

PALABRAS CLAVE

Mensajería instantánea, *chat*, protocolo, seguridad, red.

AGRADECIMIENTOS

A mis padres, Elena y Javier, por su apoyo incondicional a lo largo de toda mi vida, y en especial, durante mi estancia en la Escuela Naval Militar.

A mi hermana Laura, por ser un pilar fundamental en mi vida.

A mis compañeros de promoción, por haber compartido conmigo tantas experiencias en estos años, y en especial al famoso *G4*.

A mis amigos Marcos, Álvaro y Aníbal, por estar siempre ahí después de tantos años.

A mi director, Rafa, por hacer que este proyecto sea posible.

CONTENIDO

Contenido	1
Índice de Figuras	3
Índice de Tablas.....	4
1 Introducción y objetivos	5
1.1 Motivación	5
1.2 Objetivo.....	5
1.3 Estructura de la memoria	5
2 Conceptos básicos	7
2.1 Mensajería instantánea	7
2.1.1 Concepto	7
2.1.2 Servidor.....	7
2.1.3 Cliente.....	9
2.2 Protocolos.....	9
2.2.1 Protocolos abiertos.....	10
2.2.2 Protocolos privados	11
2.3 Seguridad	11
2.3.1 Parámetros de la seguridad	12
2.3.2 Posibles ataques	12
2.4 Cifrados.....	13
2.4.1 AES.....	13
2.4.2 RSA.....	14
2.4.3 Funciones Hash, SHA.....	14
2.4.4 Diffie-Hellman key Exchange	14
2.4.5 PFS.....	14
2.4.6 Cifrado OTR	14
2.4.7 Cifrado PGP	15
2.5 Conexión	15
2.5.1 SSH.....	15
2.5.2 SSL	15
2.6 Certificados	15
2.6.1 Estándar X.509	15
3 Aplicaciones civiles.....	17

3.1 Estado del arte	17
3.2 Ejemplos de los clientes de IM más característicos	19
3.2.1 Telegram	19
3.2.2 Whatsapp	21
3.2.3 Snapchat.....	22
3.2.4 Facebook Messenger	22
3.2.5 Hangout.....	23
3.2.6 Skype	23
4 Aplicaciones militares (Capítulo omitido en la versión pública de este TFG).....	25
5 Despliegue de un servidor jabber en una red inalámbrica.....	26
5.1 Servidor y cliente elegidos	26
5.2 Implementación en la red inalámbrica del cuartel de alumnos Marqués de la Victoria	30
5.3 Implementación en la red inalámbrica de las Lanchas de Instrucción.....	32
6 Conclusiones y líneas futuras	34
7 Bibliografía.....	35
Anexo I: Topología en redes	37

ÍNDICE DE FIGURAS

Figura 2-1. Funcionamiento IM simple ^[1]	8
Figura 2-2. Funcionamiento del IM multi-servidor ^[1]	8
Figura 2-3. Funcionamiento del IM con comunicación p2p ^[1]	9
Figura 3-1. Cifrado del protocolo MTPProto ^[2]	20
Figura 5-1. Acceso a la consola del servidor desde un ordenador de la red.	27
Figura 5-2. Acceso a la consola del servidor desde el ordenador que hace de servidor.	27
Figura 5-3. Login en el cliente Spark.	28
Figura 5-4. Interfaz del cliente Spark.	29
Figura 5-5. Conversación con el cliente Spark.	29
Figura 5-6. Interfaz del servidor Openfire.	30
Figura 5-7. Captura paquetes XMPP con Wireshark (sin cifrar).	31
Figura 5-8. Captura paquetes XMPP con Wireshark (con cifrado).	32

ÍNDICE DE TABLAS

Tabla 2-1. Puertos utilizados por el protocolo XMPP.....10

Tabla 3-1. Principales aplicaciones de mensajería en el mundo^[5] 19

1 INTRODUCCIÓN Y OBJETIVOS

1.1 Motivación

Las tecnologías de la información y la comunicación han experimentado un crecimiento exponencial en los últimos años. Es por eso que han aparecido nuevos servicios como la mensajería instantánea. En este proyecto se realiza un análisis de los servicios y aplicaciones de mensajería instantánea existentes, tanto en el ámbito civil como en el militar. Como resultado se presenta un estudio de viabilidad para implementar un sistema de mensajería instantánea integrado en aplicaciones de buques de la Armada Española.

1.2 Objetivo

El objetivo de este proyecto es entender cómo funcionan los servicios y aplicaciones de mensajería instantánea más usadas en la actualidad. Para ello se realiza un estudio exhaustivo de protocolos, funcionamiento de servidores, clientes de mensajería y sistemas de seguridad y cifrado.

Además, este TFG plantea una posible implementación de un sistema de mensajería instantánea como red de mando cifrada. Esto implica la realización de un estudio de viabilidad que tenga en cuenta el *software* existente y las necesidades y requisitos de la Armada Española y la OTAN para este tipo de tecnologías.

1.3 Estructura de la memoria

La memoria está organizada en capítulos con la siguiente estructura:

- En el capítulo 2 se definen los principales conceptos relacionados con la mensajería instantánea. Se explica el funcionamiento de servidores, clientes, así como los diferentes protocolos. Además, se hace un especial hincapié en la seguridad, detallándose los principales medios de cifrado y autenticación existentes.
- En el capítulo 3 se presenta el ‘estado del arte’, donde se explica el uso de los servicios de mensajería instantánea en la actualidad. También se incluyen unas pequeñas reseñas históricas de su evolución hasta el día de hoy. Tras esto, se realiza un análisis de las aplicaciones más características existentes en el mercado. Las aplicaciones estudiadas son: Telegram, Whatsapp, Snapchat, Facebook Messenger, Hangout y Skype.
- En el capítulo 4 se definen las condiciones impuestas por la Armada y la OTAN en el despliegue de servicios de mensajería instantánea. Posteriormente, tras un estudio de las redes de defensa, se propone cómo podría implementarse este servicio. Debido a la clasificación de seguridad que posee esta información, la propuesta se hace sobre redes genéricas, muy distintas a las redes reales de la Armada Española.
- La parte práctica del proyecto se encuentra en el capítulo 5. En él se explica cómo instalar y configurar un servidor en la red inalámbrica del cuartel Marqués de la Victoria. Esta instalación se basa en el estudio realizado anteriormente y considera todos los aspectos posibles, tanto militares como civiles. Con esta prueba se simula la implantación de un servidor de mensajería instantánea en una unidad de la Armada, ya sea en tierra o a bordo. En la parte final se propone el despliegue de un servicio de mensajería instantánea en las Lanchas de Instrucción en cuanto cuenten con una red inalámbrica propia.
- Por último, en el capítulo 6 se presentan las conclusiones y se proponen las líneas futuras de trabajo a raíz de los resultados de este proyecto.

2 CONCEPTOS BÁSICOS

2.1 Mensajería instantánea

2.1.1 Concepto

La mensajería instantánea (IM: *Instant Messaging*) surge con la expansión de las tecnologías de las telecomunicaciones. Es una evolución del correo electrónico y permite conectar usuarios en tiempo pseudoreal en cualquier parte del mundo en ambos sentidos. La IM se diseñó inicialmente para el intercambio de texto, aunque últimamente ha incorporado nuevas funcionalidades, como son la gestión de contactos, el envío de ficheros, las aplicaciones compartidas, la entrega de mensajes *offline*, o las videoconferencias, entre otras.

Cabe mencionar la diferencia entre mensajería instantánea y *chat*, que si bien son conceptos muy similares en la actualidad, no lo eran tanto en sus orígenes, ya que la IM se refería a conversaciones privadas entre dos usuarios, mientras que el *chat* se basaba en grupos de usuarios denominados ‘conferencias’. Debido a las prestaciones de los sistemas actuales, a partir de ahora se hablará indistintamente tanto de IM como de *chat*, refiriéndonos siempre al mismo concepto.

La mayoría de los servicios de IM se basan en arquitecturas cliente-servidor, que se definen a continuación.

2.1.2 Servidor

El servidor es el sistema que almacena toda la información de los usuarios. En la mayoría de sistemas, para la comunicación entre usuarios, estos deben estar conectados a un servidor. Su funcionamiento es el siguiente:

- El cliente se da de alta y se autentica en un servidor.

- El servidor envía al cliente toda la información relativa a sus contactos, como quién está en línea. Además, el servidor avisa a los contactos de ese usuario de que este se ha conectado.
- Al comenzar una conversación, se envían los mensajes al servidor, quien los reenvía al contacto destinatario.
- Al desconectarse el usuario, se informa a todos sus contactos de dicha acción.

Los servidores pueden funcionar de diversos modos:

- IM simple (un único servidor que provee servicio a todos los clientes).

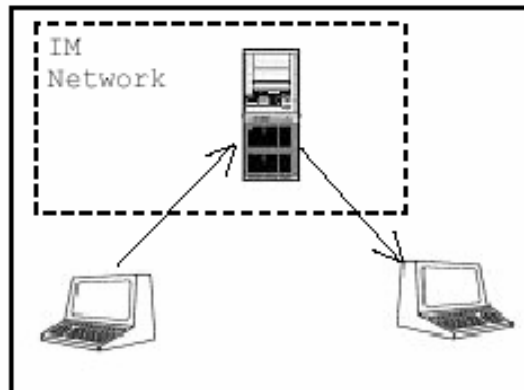


Figura 2-1. Funcionamiento IM simple^[1]

- IM multi-servidor (varios servidores sincronizados entre sí).

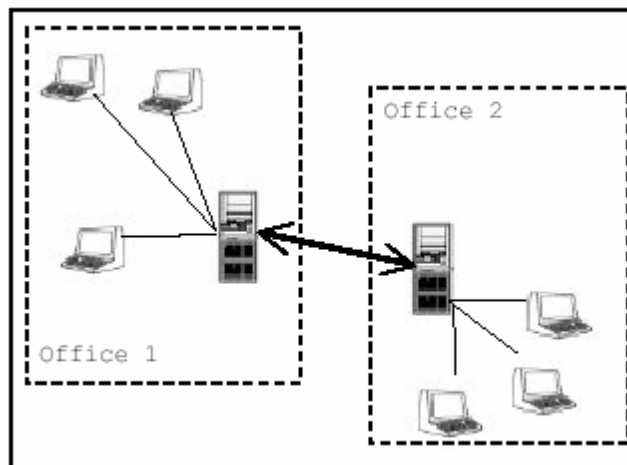


Figura 2-2. Funcionamiento del IM multi-servidor^[1]

- IM por comunicación p2p -*peer to peer*- (se utiliza el servidor únicamente para establecer la conexión inicial).

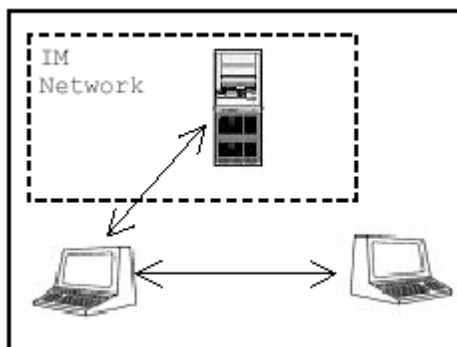


Figura 2-3. Funcionamiento del IM con comunicación p2p^[1]

2.1.3 Cliente

Son las aplicaciones que emplean los usuarios finales. Permiten acceder a todos los servicios del servidor (ver contactos, establecer conexión con ellos, etc). Existen multitud de clientes en función del protocolo en que se basa el *chat*. Así, existen clientes jabber que emplean protocolo XMPP (Spark, Psi, Gajim), clientes con protocolo propietario (MSN, AOL, Yahoo Messenger, Skype), y otros clientes de código abierto modificado (Whatsapp) o nuevos clientes con código único y abierto (Telegram). Para una elección correcta, hay que tener en cuenta diversos factores, como son el propósito del *chat*, el nivel de seguridad necesario, la plataforma de ejecución, etc.

2.2 Protocolos

Existen dos grupos principales de protocolos: los abiertos y los propietarios. Ambos se detallan más adelante.

Además hay otros tipos de *chat*: IM con comunicación p2p y *chats* IRC.

- IM con p2p: Este sistema de mensajería instantánea únicamente requiere un servidor para establecer la conexión inicial, realizándose posteriormente el intercambio de información directamente entre clientes. La auditoría en este tipo de sistemas se realiza mediante *gateways* IM, que actúan como herramientas auditoras. Sin embargo, si se busca controlar todos los elementos en un entorno seguro, donde el tráfico de datos vaya por el servidor para ser comprobado, este tipo de *chats* no son una buena elección.
- *Chats* IRC -*Internet Relay Chat*-. Su funcionamiento se basa en IM multi-servidor, formando una red propia de servidores (IRCNet, EFNet y DALNet). La diferencia básica de este protocolo es que los usuarios no necesitan agregar otros contactos para establecer comunicación con ellos. Basta con estar conectados a la misma red para comenzar una conversación.

Sin embargo, estos tipos de *chat* no son objeto de estudio del presente TFG. A continuación, se definen los principales protocolos abiertos y propietarios.

2.2.1 Protocolos abiertos

Son protocolos con especificaciones abiertas y disponibles para todo el mundo. Normalmente existen implementaciones de estos protocolos basadas también en código abierto. Esto implica que cualquier empresa puede desarrollar un servidor o un cliente mediante el empleo del código fuente. Como contrapartida, esto dificulta enormemente el control y la clasificación de los sistemas de IM ya que existen multitud de combinaciones posibles.

En la actualidad, los dos principales protocolos abiertos son:

- XMPP -*eXtensible Messaging and Presence Protocol*- o protocolo jabber: Protocolo abierto basado en XML (*eXtensible Markup Language: es un tipo de lenguaje desarrollado para almacenar datos de forma legible*), que permite establecer una plataforma para el intercambio de datos. Es un sistema de mensajería instantánea descentralizado, es decir, existe una red de servidores interconectados entre sí conformando un sistema de IM multi-servidor. Al ser un protocolo abierto, una gran cantidad de clientes y servidores lo pueden emplear de forma gratuita. Esto facilita la instalación de servidores propios en redes locales. En lo referente a la seguridad, soporta protocolos como SASL o TLS, y cifrados GPG y OTR. Otras características son la posibilidad de crear conferencias como en los chats IRC. Sus usuarios tienen un nombre con el siguiente formato: *usuario@dominio*, donde el usuario es también denominado Jabber ID o JID. Algunos ejemplos de servidores públicos son *jabberes.com* o *jabber-hispano.com*. En cuanto a clientes, los más conocidos son *Gajim*, *Spark* o *Psi*. El protocolo XMPP permite también la comunicación con servidores que no sean Jabber mediante el empleo de pasarelas. En la capa de aplicación, puede emplear el protocolo HTTP si se encuentra con el puerto del XMPP cerrado. A continuación se detalla una tabla con los puertos empleados por XMPP:

Puerto	Interfaz	Tipo	Descripción
5222	Todas direcciones	Cliente - Servidor	Utilizado por clientes para conectarse al servidor
5223	Todas direcciones	Cliente - Servidor	Utilizado por los clientes para conectarse al servidor por SSL
5269	Todas direcciones	Servidor - Servidor	Utilizado por servidores remotos para conectarse al servidor
9090	Todas direcciones	Consola de Administración	Utilizado para conexiones inseguras a la consola de administración
9091	Todas direcciones	Consola de Administración	Utilizado para conexiones seguras a la consola de administración
7777	Todas direcciones	Transferencia de archivos vía Proxy	Transferencia de archivos en la red XMPP
7070	Todas direcciones	HTTP Binding	Conexiones inseguras de clientes HTTP
7443	Todas direcciones	HTTP Binding	Conexiones seguras de clientes HTTP
5229	Todas direcciones	Flash Cross Domain	Permite a clientes Flash conectarse

Tabla 2-1. Puertos utilizados por el protocolo XMPP

- MTPROTO Protocol (Telegram): Protocolo abierto basado en la API de Java (se puede obtener el código fuente directamente desde su página web^[2]). Este protocolo vincula cada cuenta de usuario a un número. Sin embargo, su principal característica es la posibilidad de realizar cifrado punto a punto, es decir, proporciona un servicio totalmente cifrado desde el usuario originante hasta el destinatario. Para ello emplea una mezcla de diversos protocolos (Diffie-Hellman key exchange + RSA + AES + PSA). Sin embargo, no soporta OTR. Emplea los protocolos TCP y HTTP en el puerto 80. Algunos clientes que emplean MTPROTO, además de Telegram, son *Ultra*, *Sigram* y *ZapZap*. Más adelante, cuando se describa Telegram, se detallan más características de este protocolo.

2.2.2 Protocolos privados

Este tipo de protocolos pertenecen a entidades privadas, que se encargan de desarrollar todo el *software* relacionado con los servidores y los clientes. Al ser privados, los códigos fuente no están disponibles, por lo que su funcionamiento y estructura tan solo pueden conocerse mediante el empleo de ingeniería inversa. La mayoría de estos protocolos se emplean en sistemas de IM públicos, como es el caso de AOL, MSN o Yahoo!.

La mayor parte de los protocolos propietarios existentes se basan en el protocolo SIMPLE/SIP. Este protocolo permite crear, modificar y finalizar sesiones con uno o más usuarios. Incluye la realización de llamadas telefónicas, distribución de ficheros multimedia y videoconferencias.

Sin embargo, aunque el protocolo SIMPLE/SIP es abierto, muchos desarrolladores lo emplean solo como base, por lo que los protocolos privados no son compatibles entre sí. Algunos ejemplos de protocolos privados son:

- SIMPLE/SIP + MSNP, empleado por Microsoft 2007 Communication Server.
- SIMPLE/SIP + XMPP + Protocolo Propietario, empleado por IBM LOTUS SAMETIME.
- BANTU IM, con protocolo propietario.
- WIREDRED E/POP, que usa un protocolo propietario desarrollado por WireRed.

2.3 Seguridad

La siguiente definición de seguridad en aplicaciones *chat* se basa en el documento^[1] del Centro Criptológico Nacional (CCN)^[3] del Ministerio de la Presidencia^[4], titulado “Guía de Seguridad de las TIC. Seguridad en Mensajería Instantánea”. En dicha publicación del CCN se define la seguridad con 5 parámetros, los cuales quedan detallados a continuación.

2.3.1 Parámetros de la seguridad

- Autenticidad: Consiste en confirmar la identidad de los usuarios. En otras palabras, ayuda a prevenir las suplantaciones de identidad. La autenticidad la garantizan los algoritmos de cifrado empleados por la aplicación IM.
- Integridad: Es la garantía de que la información no ha sido manipulada, por lo que es confiable. Para obtener integridad en nuestros mensajes, debemos emplear un algoritmo de cifrado que incluya claves únicas generadas por sesión y que estén sincronizadas en los extremos. Además, un cifrado de este tipo también nos otorga confidencialidad en nuestros mensajes.
- Privacidad: Es la capacidad de mantener en secreto toda la información en cualquier fase del funcionamiento del *chat*, es decir, tanto en la transferencia de mensajes como en su almacenamiento.
- Confidencialidad: Se define como el aislamiento de la información a posibles accesos a ella no autorizados. Se diferencia de la privacidad en que esta se centra en el manejo de información a nivel individual, mientras que la confidencialidad se refiere a la naturaleza de la información. Este parámetro se aplica tanto al intercambio no autorizado de información como al acceso a la información producida por el sistema de IM.
- Disponibilidad: Es el parámetro que indica si la información puede recuperarse en un momento determinado. Implica todo lo relacionado con el almacenamiento, tanto cifrado como no, de registros, tanto en el servidor como en los clientes.

En resumen, podemos decir que para que un sistema de IM sea seguro, debe demostrar que quien habla es quien dice ser, que el mensaje no se ha manipulado, que nadie más lo ha accedido, y que solo alguien con los privilegios adecuados puede acceder a los registros de las conversaciones.

2.3.2 Posibles ataques

A continuación se detallan los ataques más comunes así como sus posibles soluciones:

- Suplantación de la identidad: Como su nombre indica, consiste en hacerse pasar por otro usuario. Es un ataque al parámetro de autenticidad explicado en el apartado anterior. Para evitarlo, se emplean diversos protocolos, como son el SSL (que trabaja con certificados digitales) o el TLS (versión renovada del SSL).
- Robo de información: Este ataque se emplea para obtener información, ya sea confidencial o no, con fines de ingeniería social o espionaje industrial, entre otros. Puede llevarse a cabo de múltiples formas:
 - Lectura del tráfico no cifrado en cualquiera de los puntos intermedios en la comunicación.
 - *Man in the Middle*: Un intruso hace creer a cada uno de los usuarios que es el otro usuario para así interceptar la información. Para no ser detectado, el intruso tiene que reenviar el contenido al usuario verdadero tras la interceptación del mensaje.
 - *DNS spoofing*: Un intruso se hace pasar por el servidor mediante la modificación de las tablas DNS para que los usuarios se conecten a él.
 - Control ilegítimo del servidor.

- Acceso a los registros almacenados de forma local por el cliente, ya que estos no suelen guardarse cifrados.

La principal debilidad explotada por este tipo de ataques son los servidores. Es por ello que se hace especial hincapié en sus protocolos de seguridad. Una posible solución es el cifrado de la información. Los principales tipos de cifrado se explican en el siguiente apartado.

- Filtraciones: Consisten en la llegada de información a un usuario que no debería tener acceso a dicha información. Pueden ser de origen malintencionado (en cuyo caso será tratado como un robo de información) o de origen voluntario. La solución para este último tipo de filtraciones es la formación de los usuarios del *chat*, así como la posibilidad de crear un manual de buenas prácticas.
- Denegación de servicio -DoS-: Consiste en interrumpir el funcionamiento del sistema. Estos ataques suelen enfocarse a los servidores. Se soluciona duplicando el servicio mediante la implantación de un sistema IM con multi-servidor, de tal forma que si cae un servidor, exista otro operativo para suplirlo.
- *Malware*: El sistema IM es una fuente de entrada de *software* malicioso en el ordenador. Puede introducirse de diversas formas:
 - Mediante la transferencia de archivos multimedia.
 - Con el acceso a una página *web* no segura difundida a través de un mensaje a modo de enlace de texto.
 - Empleando errores conocidos en el cliente para acceder a las máquinas sin permiso.

Estos ataques se pueden evitar de distintas maneras:

- Mediante el empleo de un antivirus (en casos de usuarios particulares que empleen servicio de IM de forma personal).
- Con el control de los usuarios que tienen acceso al sistema (empleado en *chats* de empresa).
- Asegurando que los puertos empleados por el sistema de IM no queden expuestos a posibles ataques.
- La opción más segura es establecer un sistema EIM -*Enterprise Instant Messaging*- o mensajería instantánea de empresa, que consiste en la creación de un sistema IM en una red interna. De esta forma no se envían ni reciben mensajes desde el exterior del cortafuegos. Además, un sistema EIM permite controlar todos los elementos, es decir, tanto en el servidor como en los clientes.

2.4 Cifrados

Como se ha visto en el apartado de seguridad, los ataques más delicados son los del robo de información, cuya solución era la implementación de cifrados. A continuación se citan los principales medios de cifrado, así como sus principales características, y ejemplos prácticos donde pueden ser encontrados.

2.4.1 AES

AES -*Advance Encryption Standard*-. Su funcionamiento básico se basa en la modificación de matrices, ya sea mediante la alteración de *bits*, filas, realizando multiplicaciones, etc. Puede tener una longitud de clave de 128, 192 o 256 *bits*. Consume pocos recursos, tanto de *software* como *hardware*.

Es por eso que ha permitido su empleo a gran escala. Para una buena seguridad, se aconseja emplear una longitud de clave de 192 bits como mínimo.

2.4.2 RSA

RSA -*Rivest, Shamir y Adleman*-. Es un sistema de cifrado de clave pública. Consiste en un sistema de claves públicas (todos los usuarios tienen las claves públicas de todos) y privadas (un usuario tiene solo su clave privada). El algoritmo se basa en la factorización de números enteros, con una longitud de cifra de mínimo 1024 *bits*. Además del cifrado de mensajes, sirve para firmar digitalmente, ya que si un usuario cifra un mensaje con su clave privada, demuestra su autenticidad cuando el destinatario lo descifra con la clave pública del originante.

2.4.3 Funciones Hash, SHA

Una función *hash* es una construcción criptográfica que comprime y cifra los mensajes. Son de bajo coste computacional y de memoria. El ejemplo más claro de función *hash* es SHA.

SHA -*Secure Hash Algorithm*-. Es una función *hash* de cifrado utilizada principalmente para la verificación de la integridad y la autenticidad de un mensaje. Se empleaba la función SHA-1 de 160 *bits*, pero en el año 2005 la resistencia de este algoritmo se vio comprometida. Por este motivo se desarrolló SHA-2, que lo forman las funciones SHA-256, SHA-384 y SHA-512, de 256, 384 y 512 *bits* respectivamente.

2.4.4 Diffie-Hellman key Exchange

Es un protocolo empleado principalmente para el intercambio de claves simétricas para crear claves de sesión. Este algoritmo se basa en operar número primos con otros números de forma aleatoria. Romper la clave es inviable computacionalmente hablando. Sin embargo, este protocolo es muy susceptible a ataques *Man in the Middle*, por lo que suele ir acompañado de otros protocolos que verifican la autenticidad de los usuarios.

2.4.5 PFS

PFS -*Perfect Forward secrecy*-. No es un sistema de cifrado en sí, sino que se trata de una propiedad de los sistemas criptográficos. Su funcionamiento se basa en la actualización de las claves en el eje temporal para evitar que estas sean descifradas.

2.4.6 Cifrado OTR

OTR -*Off The Record Messaging*- Mecanismo de cifrado comúnmente utilizado por los sistemas de mensajería instantánea. Emplea una combinación de AES (para el cifrado simétrico), Diffie-Hellman, para el intercambio de claves, y la función SHA-1. Además, las claves AES poseen la

propiedad PFS. Cabe destacar que versiones OTR anteriores a la 3.1 no tienen firmas digitales, por lo que es vulnerable a ataques *Man in the middle*.

2.4.7 Cifrado PGP

PGP -*Pretty Good Privacy*-. Algoritmo criptográfico basado en una mezcla de cifrado simétrico y asimétrico. Se genera una clave de sesión aleatoria en función de la interacción del usuario con el ratón y el teclado (parte simétrica). Esta clave es cifrada con la clave pública del destinatario para que este la descifre con su clave privada (parte asimétrica). El estándar de cifrado más empleado en el mundo, OpenPGP, se basa en esta tecnología.

2.5 Conexión

2.5.1 SSH

SSH -*Secure Shell*- Protocolo que nos permite acceder a una máquina remota a través de la red. Es básico para la gestión de claves (como las claves RSA), ya que permite el transporte de datos de forma segura mediante túneles. Es recomendable el empleo de la versión SSH-2, ya que la versión 1 tiene problemas de seguridad conocidos.

2.5.2 SSL

SSL -*Secure Sockets Layer*-. Protocolo criptográfico sobre la capa de transporte (TCP) que permite establecer conexiones seguras de forma sencilla y transparente a las aplicaciones. Proporciona autenticación y privacidad de la información. Para conseguirlo emplea certificados digitales X.509. En cuanto al cifrado, soporta varios protocolos: AES, RSA, Diffie-Hellman, funciones de la familia SHA, entre otras. Las versiones más recientes de SSL se denominan TLS.

2.6 Certificados

2.6.1 Estándar X.509

Un certificado es un documento que asocia una persona física o entidad a una dirección digital de Internet. En el estándar X.509, una autoridad de certificación emite un certificado asociando una clave pública a una persona o entidad con una dirección DNS o dirección de correo electrónico. Así, dicha dirección de Internet queda validada como segura y auténtica. En función del nivel de documentación requerida para la emisión del certificado, existen varios niveles, siendo el 1 el de más bajo nivel y el de clase 4 el más restrictivo.

3 APLICACIONES CIVILES

3.1 Estado del arte

La mensajería instantánea surge como evolución al correo electrónico tal y como se explica en el apartado 2.1. Estas tecnologías permiten conectar a personas en cualquier lugar del mundo y en cualquier momento y se han visto potenciadas con el desarrollo de los dispositivos móviles. De hecho, a día de hoy es prácticamente impensable imaginar un contexto en el que no se pueda realizar una comunicación de forma instantánea desde el móvil. La compartición de fotos, música, ubicaciones, ficheros y conversaciones de *chat* en tiempo real han sustituido casi en su totalidad a las llamadas telefónicas, las cuales tenían un coste considerable y ofrecían poca privacidad. Ahora es posible comunicarse de forma gratuita sin que las personas cercanas se enteren de la conversación. Como consecuencia, estas tecnologías han contribuido a una sociedad hiperconectada.

En el ámbito empresarial, las aplicaciones de IM permiten interconectar a los trabajadores de forma rápida, sencilla y barata. Sin embargo, la gran conectividad de estas tecnologías supone un *hándicap* en ocasiones, ya que a la vez que conecta con compañeros de trabajo, clientes o proveedores, es posible estar conectados con amigos y familiares, lo que supone una bajada en el rendimiento de la compañía. Por esta razón, cada vez son más las empresas que optan por la instalación de pequeñas redes locales en las que conectar a sus trabajadores para aumentar la productividad en los mismos. La principal diferencia con su antecesor, el correo electrónico, es que la conversación se produce en tiempo real, se pueden crear grupos de trabajo, y las numerosas extensiones actuales permiten añadir multitud de nuevas funciones.

Los orígenes de la IM se remontan a mediados de la década de los 60. Se utilizaba principalmente para notificar eventos relacionados con tareas específicas. A continuación se detallan los hitos más importantes en la historia y evolución de la IM:

- 1970: Se crea el protocolo p2p, el cual permite a los usuarios enviar y recibir mensajes tras haberse conectado al mismo ordenador.
- 1980: Surge CB Simulator, el primer servicio de *chat* en línea utilizado de forma masiva.

- 1984, PlayNet: Servicio en línea que permitía el intercambio de mensajes mientras los usuarios tenían partidas de juegos de mesa. Su faceta más destacada es el énfasis en la interfaz local y en el requerimiento de instalar un cliente en el PC local.
- 1985, Quantum Link: Control Video Corporation adquiere la licencia de PlayNet y lanza una versión alternativa (Q-Link).
- 1989: Jarkko Dikarinen crea el IRC. Este tipo de *chat* permite comunicación instantánea entre usuarios dentro de grupos, es decir, charlas grupales, sin necesidad de registros previos.
- 1990: IRC se convierte en una herramienta empleada no solo por usuarios privados, sino también por autoridades. Numerosos desarrolladores comienzan a imitar esta herramienta.
- 1991: La ex CVC, en la actualidad America Online, lanza AOL para DOS. Este lanzamiento busca el perfeccionamiento de la mensajería instantánea.
- 1994: Aparece MTalk, la primera aplicación VoIP que permite hacer llamadas de voz a través de Internet.
- 1995-96: Microsoft, Radvision, Intel y la UIT impulsan la estandarización de la tecnología VoIP. No sería hasta casi 8 años después cuando se empezaron a ver resultados comercialmente viables.
- 1996: Aparece ICQ de la mano de Mirabilis. Es un mensajero gratuito que sienta las bases de la mayoría de aplicaciones de IM que le sucedieron.
- 1997: AOL presenta AIM, que incluía textos en tiempo real, salas de *chat* y transferencia de archivos.
- 1998: AOL adquiere ICQ y patenta la tecnología de red que permite que los usuarios se vean y hablen entre sí cuando están conectados.
- 1999: Microsoft lanza MSN Messenger, que se convertiría en el mensajero instantáneo más popular. Existían diversas versiones que permitían conversar con usuarios de AIM, así como personalizar la apariencia de la interfaz de la conversación y ver una lista de contactos.
- 2000: Aparece Jabber, basado en el protocolo abierto XMPP. Fue el elemento necesario para la comunicación entre clientes con diferentes protocolos.
- 2001: Ericsson, Nokia y Motorola inician el proyecto Wireless Village para el desarrollo de especificaciones universales para la mensajería instantánea móvil.
- 2002: Yahoo y AOL presentan mensajería instantánea para dispositivos móviles basados en una interfaz SMS.
- 2003: Aparece la aplicación VoIP más popular, Skype.
- 2005: Surge Google Talk de la mano de Google, basado en el protocolo XMPP. Esta aplicación integra el servicio de correo electrónico de Gmail.
- 2005: MSN pasa a denominarse Windows Live Messenger, el cual ofrece mensajería fuera de línea y muchas más opciones de personalización.
- 2008: La red social Facebook presenta su *chat* integrado.
- 2009: Dos ex empleados de Yahoo! Fundan Whatsapp.
- 2011: Facebook lanza su aplicación de mensajería para móviles con sistemas iOS, Android y Blackberry.
- 2012: Facebook lanza su aplicación de mensajería para escritorio de forma oficial. Este cuenta además con funciones propias de la red social.
- 2014: Facebook compra Whatsapp por la cantidad de 19 mil millones de dólares.

Para hacernos una idea de la enorme importancia que hoy en día tienen estos tipos de tecnologías, se muestra a continuación una tabla con datos de algunas de las principales aplicaciones de mensajería instantánea:

PRINCIPALES APPS DE MENSAJERÍA EN EL MUNDO				
App	Usuarios (millones)	Propietaria	Principales Mercados	Servicios
Whatsapp 	465	Facebook	EEUU, Europa y América Latina	Mensajería
Line 	400	Naver Corp. (NHN)	Japón, Taiwán y Sudeste asiático	Mensajería, juegos, llamadas, stickers
WeChat 	355	Tencent	China	Mensajería y servicios varios
Viber 	300	Rakuten	Japón	Mensajería, llamadas, stickers
Kakao Talk 	133	Kakao Corp.	Corea	Mensajería, llamadas, stickers

Tabla 3-1. Principales aplicaciones de mensajería en el mundo^[5]

3.2 Ejemplos de los clientes de IM más característicos

A continuación se detallan los clientes de mensajería instantánea que poseen alguna característica de especial interés que ha motivado su estudio en el presente proyecto.

3.2.1 Telegram

Es una aplicación de mensajería instantánea aparecida en agosto de 2013. Puede ejecutarse en cualquier plataforma (iOS, Android, Windows Phone, y en ordenadores Windows, OSX y Linux). Su funcionamiento se basa en la nube. Es decir, todos los mensajes se almacenan en un servidor para poder reanudar una conversación desde cualquier dispositivo. A diferencia de otras aplicaciones, es totalmente gratuita y sin fines comerciales.

Su funcionamiento se basa en un protocolo propio pero abierto, el MTProto Protocol, para permitir el desarrollo de nuevos clientes compatibles. Sin embargo, a diferencia de lo que pasaba con el protocolo Jabber, los desarrolladores de Telegram no ofrecen todo el código fuente, ya que al ser un *chat* basado en la nube, no quieren que su red presente bifurcaciones debido a la creación de servidores por parte de otros usuarios.

Telegram permite la transferencia de cualquier tipo de fichero con un tamaño máximo de 1,5 GB. Para establecer la comunicación con otro usuario, basta con tener su número de teléfono o buscarlo por su apodo en Telegram. A diferencia de otros sistemas de IM, no permite realizar llamadas ni

videollamadas. Se pueden crear grupos de hasta 200 usuarios, y enviar mensajes de difusión a un número máximo de 100 contactos. En cuanto al transporte, puede ser por HTTP o por TCP.

Una de las principales características de Telegram es su seguridad. Al basarse en un protocolo abierto, están dispuestos a que cualquier experto en la materia revise su seguridad. Para establecer las conexiones, se realiza un cifrado servidor-cliente y otro cliente-cliente basado en AES256 y RSA2048. Dicho cifrado funciona de la siguiente manera:

MTPROTO encryption

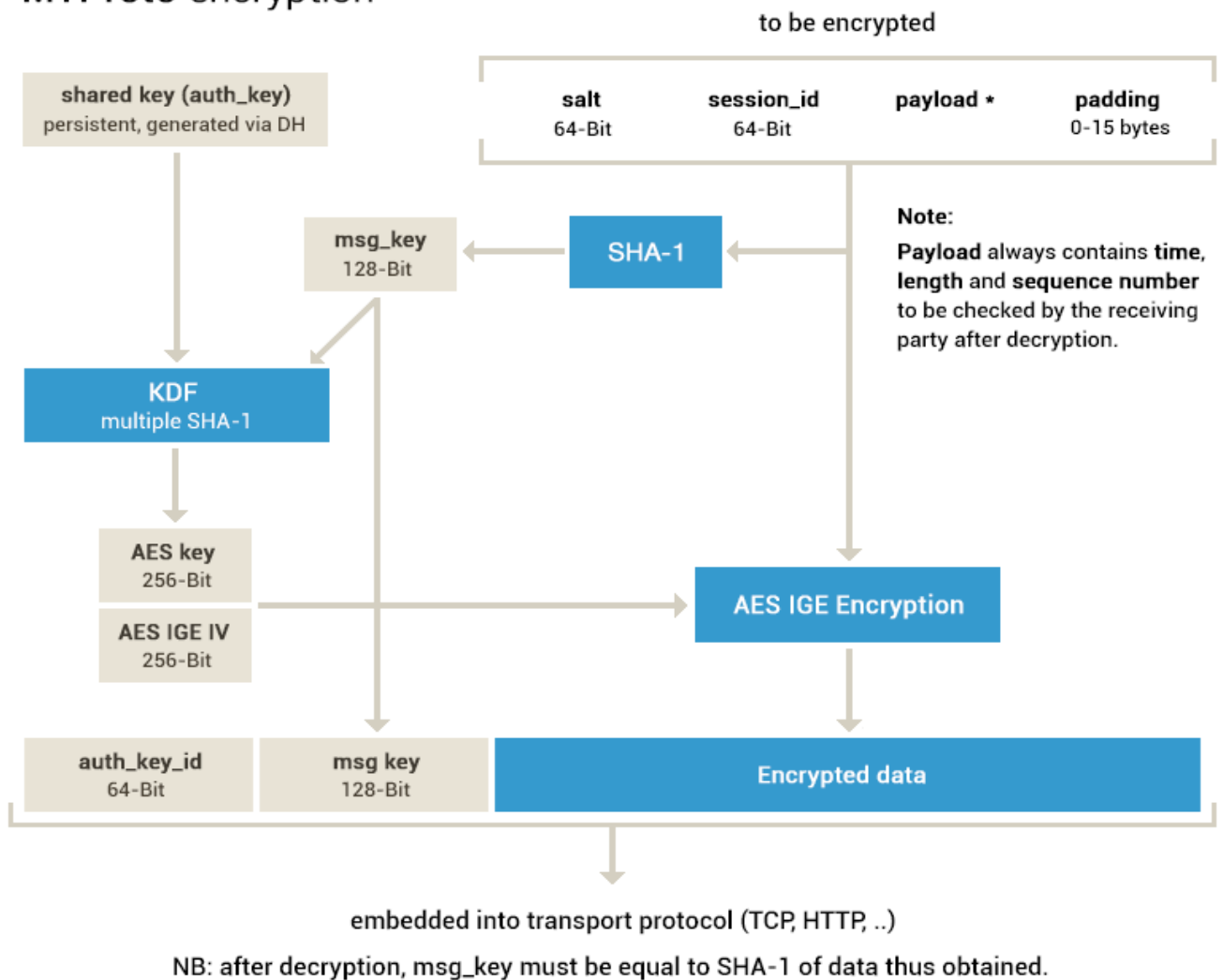


Figura 3-1. Cifrado del protocolo MTPROTO^[2]

Además, Telegram ofrece la posibilidad de crear los denominados “*chats* secretos”. La diferencia de estos *chats* radica en que no deja huella en el servidor. Es decir, este no almacena copia de las conversaciones de un *chat* secreto (A cambio se pierde la opción de reanudar la conversación desde otro dispositivo). Los *chats* secretos ofrecen cifrado extremo a extremo, mediante el protocolo Diffie-Hellman. También se elimina aquí la opción de reenvío de mensajes y ofrece la posibilidad de configurar el mensaje de tal forma que se autodestruya una vez haya sido leído.

3.2.2 Whatsapp

Whatsapp es un servicio de mensajería instantánea fundado en 2009 por dos exempleados de Yahoo!. En la actualidad, cuenta con más de 500 millones de usuarios en todo el mundo, principalmente en Europa, India y Latinoamérica.

Se basa en el protocolo XMPP, aunque tiene sus propios servidores y el código propio de la aplicación no es abierto. Al basarse en XMPP, todos los usuarios necesitan un JabberID o JID como se explicó anteriormente. En el caso particular de Whatsapp, este JID es *nºde teléfono@s.whatsapp.net*.

La autenticación con el servidor se realiza de forma transparente para el usuario. Este solo tiene que introducir la contraseña la primera vez que se conecta al servidor. La contraseña se genera con la función *hash* MD5 de la dirección MAC para iPhone, y con la misma función del código IMEI para móviles con sistema operativo Android. Dicha contraseña se envía por SMS al usuario.

El funcionamiento del sistema es el siguiente:

- El usuario envía un mensaje, que se almacena en el servidor.
- El servidor envía un asentimiento al usuario originante, el cual ve un símbolo “*check*” en el mensaje como muestra de que el mensaje ha llegado al servidor.
- Cuando el usuario destinatario se conecta, el servidor envía el mensaje.
- Al recibir el mensaje, el destinatario envía un asentimiento al servidor, el cual lo notifica al originante (doble *check*).
- Por último, el servidor borra el mensaje.

Hay que destacar que los mensajes se envían cifrados (originalmente se enviaban como texto plano). Sin embargo, se cifran con SHA1, cuya seguridad está comprometida en la actualidad. Por lo tanto, es potencialmente posible descifrar mensajes interceptados. Otro fallo de seguridad es el acceso a los servidores, el cual es bastante sencillo, ya que en ellos no existe ningún tipo de protección. Además, hay que recordar que los mensajes quedan almacenados hasta que el proceso de entrega finaliza (o hasta un máximo de 15 días si la entrega no se produce). Es por tanto que cualquiera con conocimientos puede acceder a los mensajes en el servidor (también podría acceder a la agenda de contactos de cualquier usuario). A pesar de que la empresa asegura que no guarda conversaciones, analistas afirman que sí guardan conversaciones en sus servidores durante un determinado tiempo. También almacenan los números de teléfono de las agendas (de forma cifrada), aunque dichos números no sean usuarios de Whatsapp. Respecto a las contraseñas de autenticación, cabe destacar que en el iPhone se almacena sin cifrar (es decir, accediendo al teléfono otra persona podría suplantar ese usuario) y en Android se almacena cifrada pero fácilmente descifrable. En cuanto al transporte de datos, los mensajes se envían por TCP, mientras que los archivos multimedia y el cálculo/intercambio de contraseña de autenticación se realiza por HTTPS.

Como características principales de esta aplicación de cara al usuario tenemos: intercambio de imágenes, audios, vídeos, ubicaciones GPS, y mensajes de voz. Ofrece la posibilidad de enviar mensajes de difusión, y de crear un grupo de *chat* de hasta 100 usuarios. El tamaño máximo de los ficheros multimedia es 10 MB. Respecto al proceso de agregar usuarios, es una acción transparente

para el usuario, ya que el servidor gestiona la agenda de contactos de todos los usuarios, que se utiliza para agregar automáticamente a los contactos con Whatsapp.

3.2.3 *Snapchat*

Es una aplicación de mensajería instantánea desarrollada por unos estudiantes de la universidad de Stanford como proyecto de fin de carrera en abril del 2011. Actualmente está valorada en más de 10 mil millones de dólares y cuenta con más de 100 millones de usuarios. Puede ejecutarse en móviles iOS y Android.

Su principal función es el envío de fotografías, pero lo que la diferencia de otras aplicaciones es que dichas imágenes se ‘borran’ tanto en el móvil receptor como en el servidor en un intervalo de 1 a 10 segundos, que elige el originante del mensaje. La autenticación de usuarios se realiza mediante SHA-256 y el cifrado de todos los datos se realiza con AES. Este cifrado es cliente-servidor. Su protocolo es privado y propietario, por lo que no se puede ver ni modificar su código fuente, con lo que su seguridad no está acreditada.

La esencia de esta aplicación es el borrado automático de los datos enviados, lo que a priori podría ser interesante para el envío de información militar clasificada en el desarrollo de un sistema IM para la Armada. Sin embargo, existen algunos fallos de seguridad conocidos en esta aplicación.

Al tratarse de código cerrado, estas vulnerabilidades no pueden ser demostradas. Sin embargo, se ha comprobado que tanto los servidores como los móviles Android (en iOS aún no se ha demostrado) guardan de forma permanente una gran cantidad de metadatos de los archivos, con lo que alguien con suficientes conocimientos podría recuperar, en este caso, las fotos. Es cierto que es un proceso complejo y lento de hacer, pero igualmente posible, con lo cual, es un sistema no válido para el envío de información confidencial. Además, la propia empresa reconoce que si el receptor no ve el mensaje, este se almacena hasta un máximo de 30 días en los servidores, lo que aumenta la probabilidad de interceptación de dicha información.

3.2.4 *Facebook Messenger*

Es una aplicación de mensajería instantánea asociada a la mayor red social del mercado. Actualmente cuenta con más de 1350 millones de miembros registrados, con lo que el *chat* tiene potencialmente este mismo número de usuarios. Este sistema de IM está basado en un código abierto, el MQTT, que es un protocolo muy sencillo que corre sobre TCP/IP. Dicho protocolo está optimizado para dispositivos móviles con poca capacidad de procesamiento y ancho de banda limitado.

“Facebook Messenger” *chat* se ha estudiado en este TFG por ser una de las redes sociales más populares, pero en aspectos de seguridad no ofrece las garantías necesarias para proporcionar un servicio de IM en la Armada. El protocolo MQTT no cifra los mensajes pero sí emplea SSL para la autenticación. Es un servicio de IM multiplataforma y, al igual que Telegram, Facebook almacena en sus servidores las conversaciones. Para eliminar dicha información de los servidores, basta teóricamente con seleccionar la opción ‘Eliminar conversación’ en el *chat*, para que desaparezca de todos los dispositivos y servidores.

Cabe destacar que Facebook Messenger puede correr en un cliente jabber, ya que tienen algunos servidores XMPP conectados con su red de servidores MQTT. En este caso particular, podría iniciarse sesión en el *chat* de Facebook con un cliente jabber, como Spark o Psi, mediante una conexión al servidor *chat.facebook.com*. Es decir, el JID sería: *nombre_de_usuario@chat.facebook.com*.

3.2.5 Hangout

Originalmente, Google poseía tres aplicaciones de mensajería instantánea: Google Chat, Google Voice y Google+ Hangouts, que empleaban XMPP. En la práctica, el empleo de XMPP por parte de Google supuso un gran impulso para la red de servidores Jabber. Sin embargo, en marzo de 2013, Google unificó estas aplicaciones en Hangout, dejando de lado el protocolo XMPP y empezó a emplear un protocolo privado y propietario.

El *chat* ha perdido en seguridad, ya que antes aceptaba, por ejemplo, cifrado OTR. De hecho, la aplicación actual cuenta con una opción denominada OTR, pero que poco tiene que ver con el cifrado. Esta opción permite no almacenar la conversación. La conversación no se guarda en el dispositivo del usuario emisor ni del receptor, pero no necesariamente en los servidores de Google. Además, esta opción está desactivada por defecto (para permitir sincronizar dispositivos). Para activar la opción OTR, hay que indicarlo contacto por contacto. Si no se activa esta opción, todas las conversaciones se guardan en claro en los servidores.

El protocolo propietario de Hangout tiene muchas carencias en lo que a seguridad respecta: no es un protocolo abierto que se pueda revisar, las conversaciones no se cifran, los usuarios no son autenticados, etc. Es decir, en este aspecto, Google ha dado un paso atrás en seguridad al abandonar XMPP. La única seguridad es la seguridad que ofrezca la red, pero el protocolo es totalmente transparente en seguridad.

3.2.6 Skype

Es la aplicación más popular que ofrece servicio VoIP. Emplea un protocolo privado, actualmente propiedad de Microsoft, que compró Skype en 2011. Lo más destacable de esta aplicación es que se basa en p2p, es decir, solo se conecta con el servidor al iniciar la conexión y se realiza el intercambio de información entre usuarios de forma directa sin el paso intermedio por el servidor. Además, este protocolo emplea, en su versión gratuita, cifrado AES de 256 bits tanto en voz, como en archivos multimedia enviados y en el texto de los mensajes. La versión de pago cuenta con el algoritmo RSA de 2048 bits para el cifrado. Ambas versiones emplean RSA de 1024 para la negociación de claves de sesión al inicio de la conexión. Sin embargo, el cifrado de la conferencia se realiza con RC4 (algoritmo muy simple sobre el que se basan otros más complejos como RSA, SSL y TLS) debido a la necesidad de fluidez en la comunicación.

Se puede considerar uno de los protocolos gratuitos más seguros, ya que no almacena conversaciones en el servidor (conexión p2p) y toda la información va cifrada. Sin embargo, es un protocolo privado, con las desventajas que ello supone. Además, al ser un servicio tan extendido, es

objeto de numerosos ataques. Cabe mencionar que la Agencia de Seguridad Nacional estadounidense (NSA) monitoriza las conversaciones de Skype mediante el sistema de vigilancia electrónica PRISM.

4 APLICACIONES MILITARES

**Capítulo omitido en la
versión pública de este TFG
por contener información de
carácter confidencial**

5 DESPLIEGUE DE UN SERVIDOR JABBER EN UNA RED INALÁMBRICA

5.1 Servidor y cliente elegidos

Tras el estudio realizado a lo largo del TFG, y tras considerarse todas las variables, se ha optado por emplear el protocolo XMPP para la implementación de un servidor en una red inalámbrica. Al tratarse de un protocolo abierto, no es necesario desarrollar ningún servidor propio ni una aplicación *chat*. Además, como se ha visto anteriormente, este es un protocolo ya empleado en el ámbito OTAN con el J-Chat de la NS-WAN (que no era más que el cliente civil Psi empleado en el ámbito militar). Es decir, que al implementar un *chat* basado en jabber, se podría inter-operar con el resto de *chats* de la alianza (ya se vio que una de las principales características de este protocolo es la comunicación entre diferentes clientes).

Según un estudio^[1] del Centro Criptológico Nacional, el mejor servidor jabber, en lo que ha seguridad se refiere, es Sun Java System IM. Sin embargo, debido a su sencillez en la administración, se ha optado por el servidor OpenFire, el cual se encuentra en 3ª posición en dicho estudio, siendo considerado un 14% más inseguro que Sun Java System IM. Esta inseguridad queda subsanada al ejecutar el servidor sobre una red segura, como sería en el caso de su implementación en la Armada, y no sobre la red inalámbrica.

Para la elección del cliente, se probaron los más usados: Psi, Spark, Gajim y Jabber MomentIM. Todos ellos ofrecen las mismas características, si bien Psi y Spark tienen una interfaz más fácil e intuitiva.

Con todo esto, para implementar un servidor jabber en una red inalámbrica, se ha optado por el tándem Openfire-Spark, ambos desarrollados por Ignite Realtime. A continuación se detallan sus principales características:

- Servidor Openfire: Servidor de código abierto basado en XMPP. Su principal característica respecto a otros servidores es su interfaz gráfica, mediante la cual somos capaces de

acceder a la consola de administración del servidor. Esta conexión se realiza por el puerto 9090 en HTTP, o por el 9091 en HTTPS. A continuación se muestra el acceso a la interfaz:



Figura 5-1. Acceso a la consola del servidor desde un ordenador de la red.

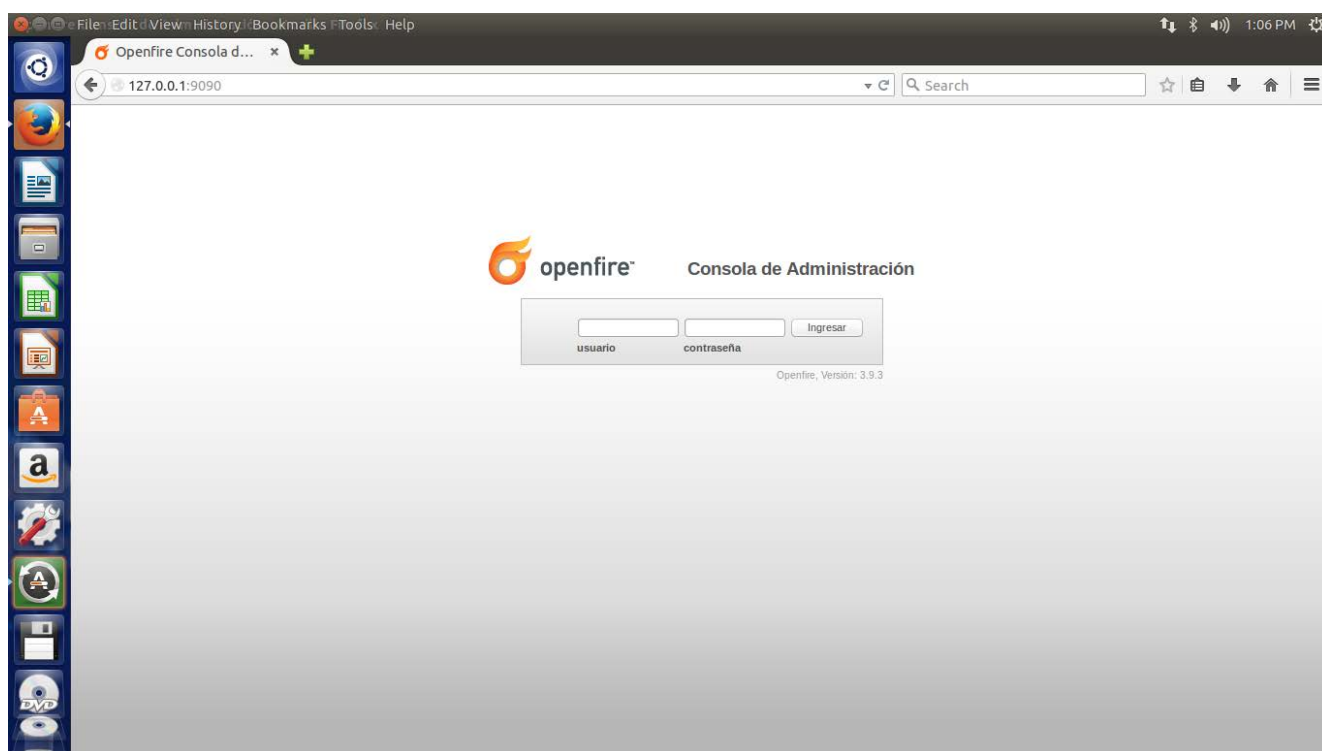


Figura 5-2. Acceso a la consola del servidor desde el ordenador que hace de servidor.

Como se puede observar, basta con introducir en la barra de direcciones del navegador la dirección IP del servidor y el puerto para conectarnos. De forma interna (desde la máquina que actúa de servidor), introduciríamos 127.0.0.1:9090, que es la IP que toma la máquina de forma interna.

Las otras características del servidor Openfire son: interfaz para agregar extensiones, empleo de SSL/TLS, almacenamiento no cifrado de las conversaciones en el servidor, creación de conferencias, generación de estadísticas, y una manera muy intuitiva de gestionar conversaciones y usuarios.

Corre en Windows, MAC OS y Linux, y puede descargarse gratuitamente desde su página web^[9]

- Cliente Spark: Es un cliente Java de código abierto desarrollado por IgniteRealtime, mismo desarrollador que Openfire. Su interfaz es muy intuitiva, facilitando personalizar muchas características del *chat*. Permite la instalación de diversas extensiones, como el del cifrado OTR. Permite guardar copia en local de las conversaciones, según lo requiere la JUTAC en la X-JT-01, si bien cabe destacar que esta información se guarda sin cifrar. Además, permite tener varias conversaciones simultáneas de manera muy cómoda. A continuación se muestran diversas capturas de pantalla del cliente Spark:



Figura 5-3. Login en el cliente Spark.



Figura 5-4. Interfaz del cliente Spark.

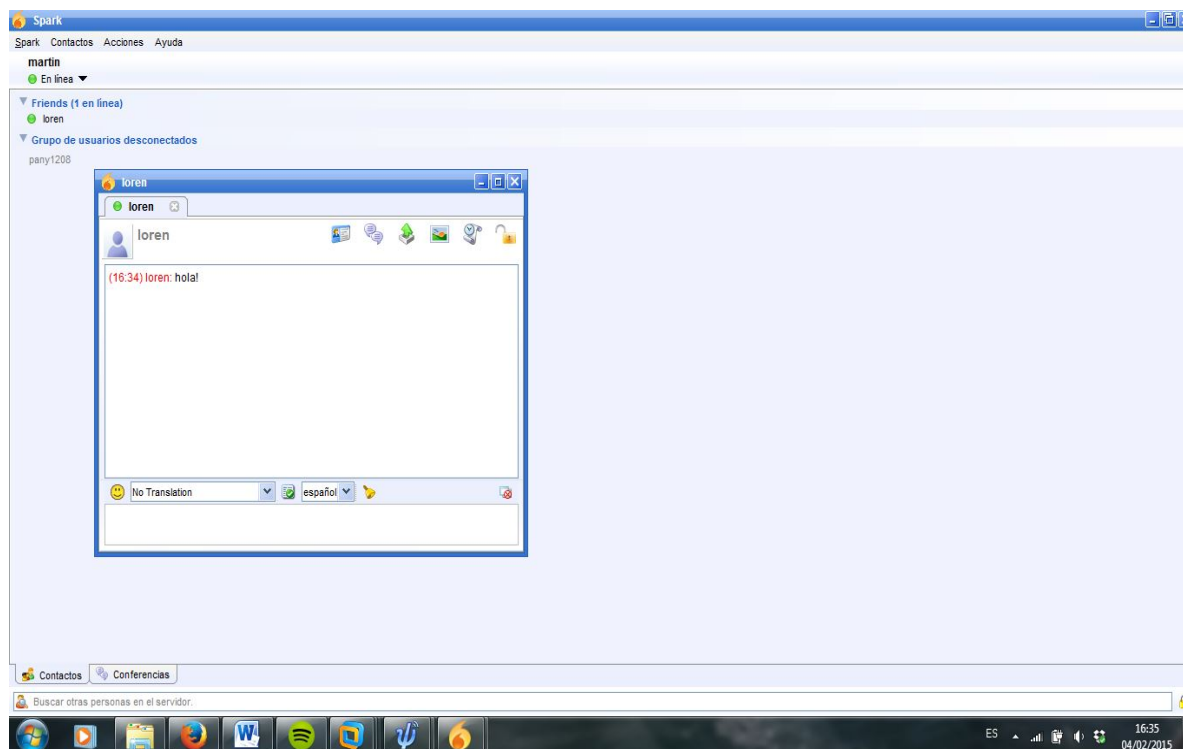


Figura 5-5. Conversación con el cliente Spark.

Esta aplicación también puede ser descargada gratuitamente de la página de su desarrollador^[9].

En resumen, se dispone de un sistema XMPP con el servidor-cliente Openfire-Spark, ambos del mismo desarrollador, IgniteRealtime, con lo que se tiene la máxima funcionalidad, en lo que a compatibilidad se refiere, en el *chat*. Si bien puede generar dudas el hecho de que las conversaciones se almacenan sin cifrar, la seguridad en este caso la otorga la red sobre la que se ejecuta el *chat*.

Con todo esto, se decide instalar un servidor jabber en la red inalámbrica del cuartel. En el siguiente apartado se detallan los pasos seguidos para dicha instalación.

5.2 Implementación en la red inalámbrica del cuartel de alumnos Marqués de la Victoria

Para la instalación del servidor, se optó por el sistema operativo Ubuntu, cuya última versión se descargó desde su página web^[10]. Dicho sistema operativo se instaló en una máquina virtual empleando el programa VMware, descargado de la página web oficial^[11].

Los pasos realizados para la instalación de Openfire en Ubuntu son los descritos en su página web de Internet^[12]. En resumen, hay que instalar la última versión de Java de Oracle, así como abrir los puertos que empleará el servidor (tabla 2-1).

Una vez instalado el servidor, accedemos a la consola de administración para configurarlo, como se mostró en la Figura 5-2. Al ser la primera vez que configuramos el servidor, es necesario fijar algunos campos que serán los que tome el servidor, como la IP a la que tendrán que conectarse los clientes, el idioma de la interfaz, o la configuración de la base de datos. Una vez fijadas estas características, accedemos a la página principal de configuración de Openfire, que se muestra a continuación:

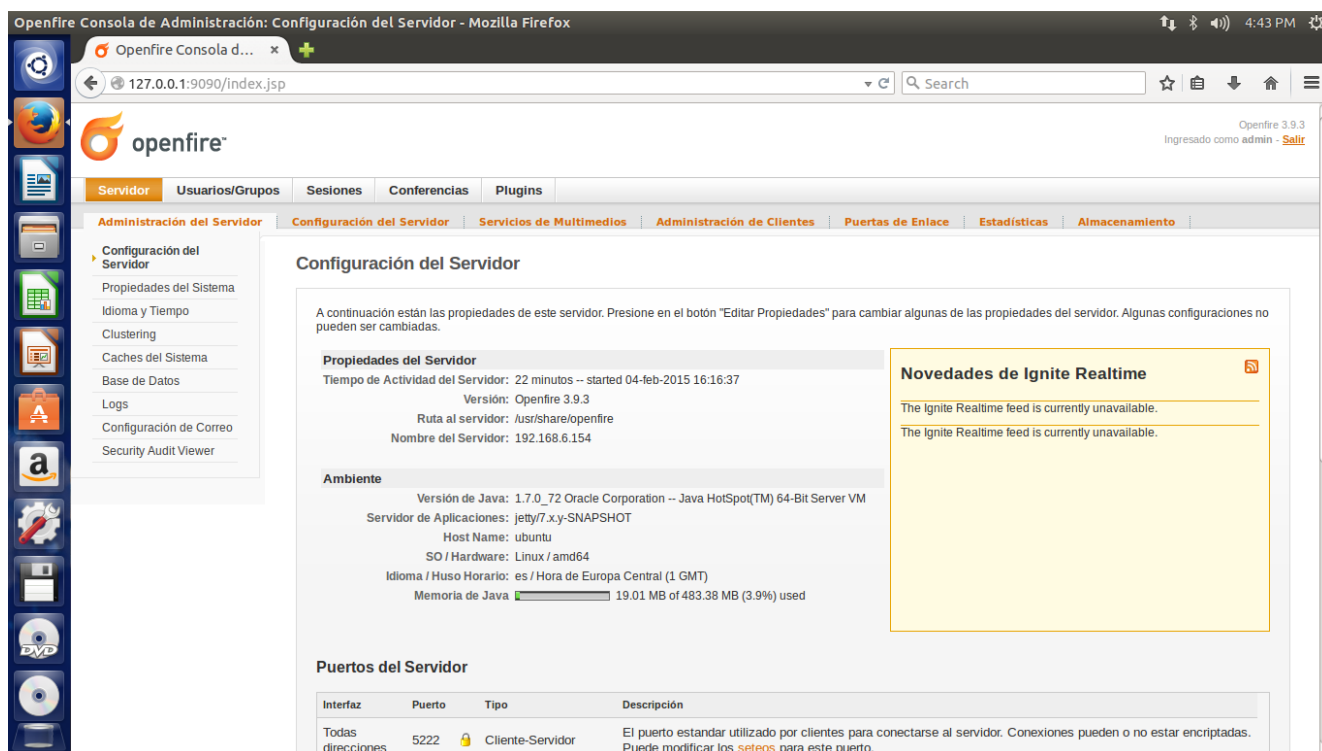


Figura 5-6. Interfaz del servidor Openfire.

Navegando por las diversas pestañas de la interfaz, se pueden configurar numerosas opciones del servidor, así como gestionar usuarios o conferencias, visualizar estadísticas de uso del servidor, entre

otras cosas. Al no ser el objetivo de este proyecto, no se adjuntan más capturas de la interfaz de Openfire mostrando todas las características citadas.

Lo que sí es importante es la gestión de usuarios. En este caso, y para realizar diversas pruebas, se crearon 4 usuarios con sus respectivas contraseñas. A la hora de realizar el *login* en el cliente Spark, lo único a tener en cuenta es introducir en el campo ‘Servidor’ la IP de la máquina virtual donde se encuentra instalado el servidor. En este caso particular, y como se aprecia en la Figura 5-3, esta dirección IP es 192.168.6.154, con lo que nuestros usuarios tendrán un JID del estilo: *nombre@192.168.6.154*.

A continuación se instaló la herramienta Wireshark para capturar los paquetes de la red, y de este modo analizar los paquetes generados por el *chat*. Dichos paquetes están en formato XML, por lo que son sencillos y fáciles de interpretar. Una de las pruebas realizadas consistió en desactivar el cifrado de los mensajes, para comprobar que con Wireshark era posible ‘leer’ dichos mensajes mediante captura. Aquí se muestran las pruebas obtenidas:

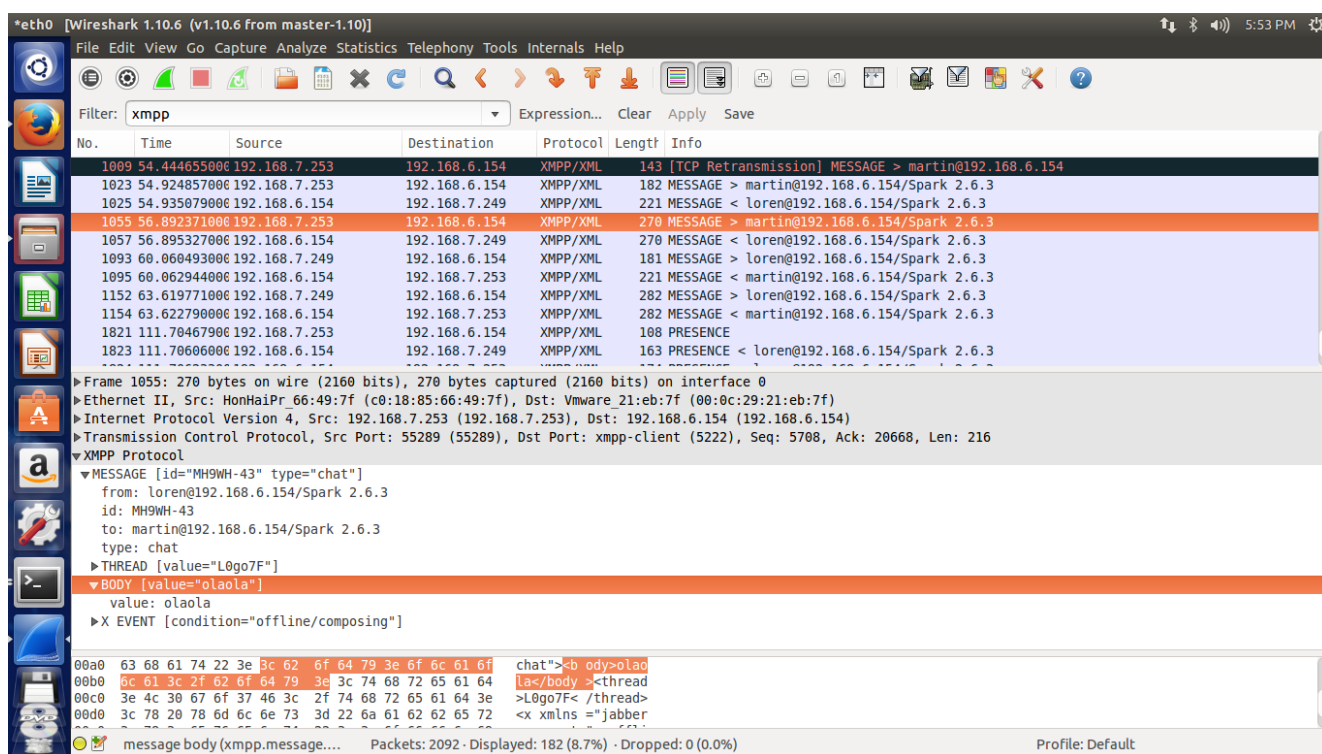


Figura 5-7. Captura paquetes XMPP con Wireshark (sin cifrar).

Como se observa en la Figura 5-7, es posible leer perfectamente el mensaje enviado (resaltado en naranja). Además, se puede ver quién es el originante y el destinatario del mensaje. En la misma captura se pudieron ver todos los mensajes que intercambia el servidor con sus clientes: estado de cada usuario -Mensaje PRESENCE en la figura 5-7; *handshake* inicial al establecerse la conexión con el servidor; desconexión del usuario; etc.

Es decir, que al desconectar la opción de cifrado TLS en la consola de administración del servidor, cualquiera que analice la red puede leer los mensajes en claro. A continuación, volvemos a realizar una captura con Wireshark, habiendo activado de nuevo el cifrado TLS. Estos son los resultados obtenidos:

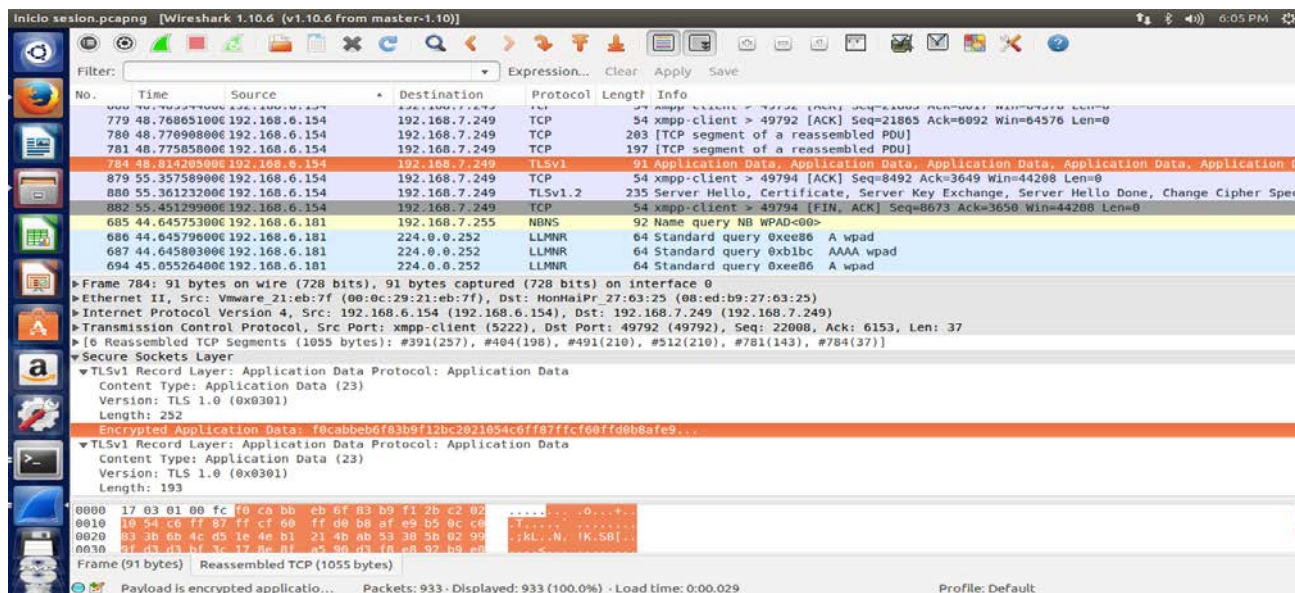


Figura 5-8. Captura paquetes XMPP con Wireshark (con cifrado).

Se observa que ya no obtienen paquetes XMPP, sino paquetes TLS (del cifrado empleado por el servidor), los cuales no se pueden leer en claro. Los únicos paquetes puramente XMPP que se obtienen son los de inicio de sesión, cuando ambas partes (servidor y cliente) negocian el empleo de TLS durante la sesión. Esto se soluciona si se establece la conexión a través del puerto 5223.

Por tanto, al activar la función de cifrado TLS en el servidor, no solo se hace más difícil la captura de paquetes XMPP (ya que estos se ven como paquetes TLS), sino que además no es posible leerlos. Es decir, se demuestra que el cifrado realmente funciona, por lo que finalmente se obtiene un servidor con las siguientes características principales:

- Dirección IP del servidor: 192.168.6.154
- Base de datos: Interna.
- Cifrado TLS requerido en el puerto 5222.
- Creación y gestión de usuarios únicamente desde el servidor.
- Certificado digital autofirmado (al tratarse de un chat en área local).
- Copia de conversaciones en el servidor durante 6 meses.

5.3 Implementación en la red inalámbrica de las Lanchas de Instrucción

Para implementar dicho servidor en la futura red inalámbrica que dispondrán las Lanchas de Instrucción, tan solo se deben considerar algunos aspectos adicionales a los ya señalados a lo largo de este trabajo.

En primer lugar, se debe destacar que lo óptimo sería implementar un servicio de IM multi-servidor para evitar la caída del *chat* si el servidor de tierra falla. Para esto hay que tener en cuenta que los servidores se conectan a través del puerto 5269 (Tabla 2-1), por lo que habría que habilitarlo.

Además, en la consola de administración del servidor debe configurarse una lista blanca (lista con los servidores a los que sí se puede conectar nuestro servidor). Otro aspecto a considerar es la base de datos: para que varios servidores compartan una base de datos, no puede ser interna como en el caso del servidor del cuartel. Como solución, se propone el empleo del *software* mysql^[17].

En cuanto a los dispositivos a emplear para el *chat*, se recomienda el empleo de dispositivos móviles, debido a la dificultad de llevar un ordenador por persona a la lancha. En dicho móvil habría que instalar un cliente que acepte el protocolo XMPP y sea capaz de conectarse a un servidor privado. Durante este TFG se realizaron pruebas con ChatSecure (tanto en iOS como en Android), siendo estas totalmente satisfactorias. No obstante, existe una amplia variedad de clientes con este fin.

Por último, habría que considerar la creación de salas de conferencias (fácilmente configurables desde la consola de administración del servidor) para que la información llegue, con un solo mensaje, a todos los destinatarios que realmente la necesiten, en lugar de tener que enviar un mensaje a cada uno de ellos por separado.

6 CONCLUSIONES Y LÍNEAS FUTURAS

Tras el estudio realizado se llega a la conclusión de que el *chat* es una herramienta de trabajo muy útil en los contextos adecuados. Permite conectar multitud de usuarios de una forma fácil de administrar, otorgando una inmediatez en la información que el sistema de mensajes actual de la Armada no ofrece.

Para un futuro, se recomienda implementar este servicio en la Armada, sustituyendo el que ya existe, de tal forma que sea una herramienta totalmente integrada con la institución. Para ello será necesario el desarrollo de nuevas doctrinas que regulen todo lo referente a la mensajería instantánea. Esto supondrá toda una revolución en las comunicaciones, y según podemos ver en el mundo civil, esta es la dirección a seguir si la Armada quiere seguir a la vanguardia de la tecnología.

Para lograr la completa instalación de una red de mando basada en el chat, es aconsejable empezar instalándolo en pequeñas redes (dentro de una fragata, dentro de un Cuartel General), de tal forma que la adaptación a este sistema sea lo más suave posible, y así asegurar un correcto empleo de este servicio.

7 BIBLIOGRAFÍA

Referencias citadas en el proyecto:

- [1] CNN STIC-440
- [2] <https://telegram.org/>
- [3] <https://www.ccn.cni.es/>
- [4] <http://www.mpr.gob.es/Paginas/index.aspx>
- [5] <http://www.neomobile-blog.com/es/wp-content/uploads/2014/04/Informe-2014-mensajeria-instantanea-neomobile.pdf>
- [6] X-JT-01 (A) CAMBIO 2 (sin verificar)
- [7] ANEP-73 Edition 2 (*Allied Naval Engineering Procedures*)
- [8] ACP-176 SP NAVY SUPP-1 CAMBIO 2
- [9] <http://www.igniterealtime.org/downloads/>
- [10] <http://www.ubuntu.com/download>
- [11] <http://www.vmware.com/es>
- [12] <http://lasegundapuerta.com/>
- [13] <http://www.armada.mde.es/>
- [14] <http://redestipostopologias.blogspot.com.es/2009/03/topologia-de-redes.html>
- [15] <http://eurasian-defence.ru/node/3378>
- [16] CNN STIC-437
- [17] <http://www.mysql.com/>

Para ampliar información de algún capítulo:

Toda la información general recogida en el capítulo 2 de este proyecto, ha sido principalmente obtenida de las referencias 1 y 16.

La información relativa a los chats civiles ha sido obtenida de las páginas oficiales de cada aplicación, siendo ampliada con múltiples fuentes de Internet (foros, monografías, proyectos de fin de carrera, etc).

El capítulo 4 se basa en las referencias 6, 7 y 8.

ANEXO I: TOPOLOGÍA EN REDES

A continuación se detallan las principales topología en redes que nos podemos encontrar, así como sus principales ventajas e inconvenientes:

- Redes bus: Se caracteriza por tener un único canal de comunicación (bus), al cual quedan conectados los nodos de la red.
 - *Ventajas*: Fácil implementación y crecimiento; muy económica; arquitectura simple.
 - *Desventajas*: Longitud del bus limitado; problema en el bus afecta a toda la red; colisiones de mensajes.

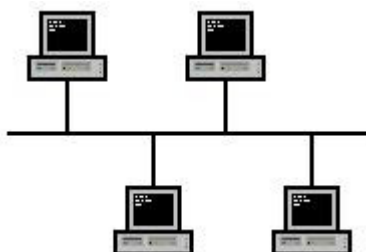


Figura A1-0- 1. Red bus.^[14]

- Redes estrella: Todos los nodos de la red están conectados a un nodo central
 - *Ventajas*: Fácil de administrar; todos los nodos comunicados de forma conveniente.
 - *Desventajas*: La red se viene abajo en caso de fallo del nodo central (fácil de atacar); más costosa; mayor cantidad de cable.

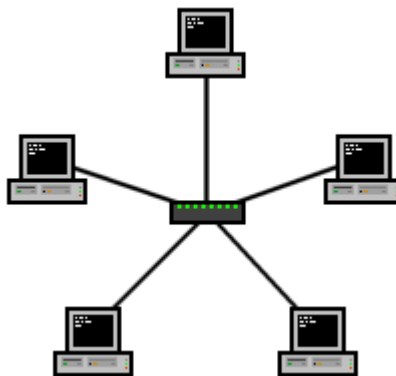


Figura A1-0- 2. Red estrella.^[14]

- Red en anillo: Cada nodo está únicamente conectado con el anterior y el siguiente.
 - *Ventajas*: Arquitectura simple; fácil implementación y crecimiento.
 - *Desventajas*: El canal se degrada en redes grandes; si falla un nodo se va abajo toda la red.

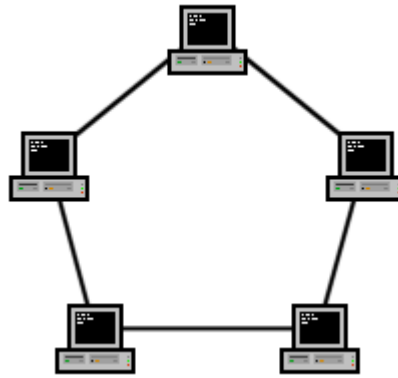


Figura A1-0- 3. Red en anillo.^[14]

- Red en malla: Cada nodo está conectado a todos los nodos. Si está completamente conectada, aseguramos la comunicación sin interrupciones.
 - *Ventajas:* La caída de un nodo no afecta al resto de la red; ofrece una fiabilidad y redundancia superior a otras tipologías.
 - *Desventajas:* Más costosa y compleja (de ahí su empleo en redes inalámbricas)



Figura A1-0- 4. Red en malla.^[14]

- Red árbol: Cada nodo tiene un enlace troncal desde el que se ramifican los demás nodos. Puede verse como una combinación de redes estrella.
 - *Ventajas:* Gran interconectividad entre nodos; red muy bien jerarquizada.
 - *Desventajas:* Arquitectura compleja y de difícil crecimiento; la caída de un nodo de alta jerarquía incomunica todos los que tiene por debajo.

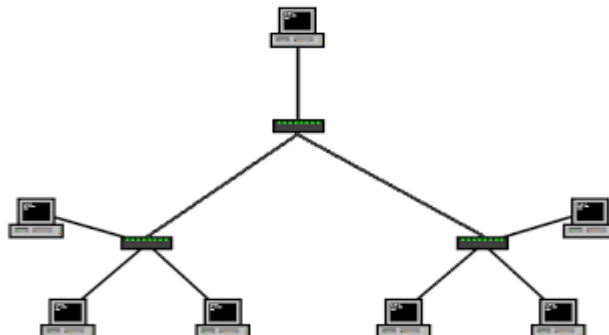


Figura A1-0- 5. Red árbol.^[14]