UNIVERSIDAD DE BUENOS AIRES

FACULTADES DE CIENCIAS ECONÓMICAS, CIENCIAS EXACTAS Y NATURALES E INGENIERÍA

MAESTRÍA EN SEGURIDAD INFORMÁTICA

TESIS DE MAESTRÍA

Vulnerabilidades del uso de correo electrónico y Desafíos de seguridad para su implementación futura
Protocolos y medios de implementación de correo electrónico seguro

DANIEL ALEJANDRO MALDONADO RUIZ
daniel.a.maldonado@ieee.org

DIRECTOR:
Mg. Ing. Juan Alejandro Devincenzi

2015

Cohorte 2013

DECLARACIÓN

Por medio de la presente, el autor manifiesta conocer y aceptar el Reglamento de Tesis vigente y que se hace responsable que la totalidad de los contenidos del presente documento son originales y de su creación exclusiva, o bien pertenecen a terceros u otras fuentes, que han sido adecuadamente referenciados y cuya inclusión no infringe la legislación Nacional e Internacional de Propiedad Intelectual

_____

Daniel Alejandro Maldonado Ruiz

DNI 95133095

RESUMEN

El presente proyecto incluye un resumen de los conceptos tecnológicos del correo electrónico, haciendo énfasis en sus adaptaciones para transportar multimedios en una plataforma diseñada para transportar texto. Se incluye también un resumen acerca de los sistemas de criptografía de clave pública, desde su origen hasta sus implementaciones actuales. Además una reseña sobre la seguridad y el anonimato en Internet y su creciente importancia en el mundo actual.

Se explican las características de seguridad que, según los conceptos extraídos de la criptografía, deben tener los sistemas de correo electrónico; para explicar brevemente las más importantes tecnologías de protección de correos electrónicos realizando un enfoque en PGP y OpenPGP, sus características, protocolos y vulnerabilidades; estas últimas fruto de la poca evolución del sistema frente a las nuevas formas de utilización de Internet.

Para afrontar estas vulnerabilidades, se plantean las características con las que deberá contar un nuevo protocolo de usabilidad y establecimiento para un sistema basado en OpenPGP, el cual trata de reducir la dificultad de utilización de OpenPGP y asegurar el almacenamiento de los mensajes que se transmiten.

Finalmente se presentan las conclusiones surgidas de esta investigación, poniendo especial énfasis en cómo las características propuestas aumentan la seguridad de las comunicaciones entre usuarios asegurando su privacidad en Internet.

Palabras clave: Correo Electrónico, PGP, almacenamiento, seguridad, privacidad, anonimato, Clave pública, usabilidad.

CONTENIDO:

INDICE DE GRAFICOS:

INDICE DE TABLAS:

INTRODUCCIÓN

Desde el inicio de Internet, la red siempre fue asociada con el correo electrónico, por ser el primer servicio ofrecido a los usuarios y que les permitía hacer lo que en el pasado parecía utópico: enviar mensajes de un lado a otro del planeta y que llegaran casi inmediatamente. Aun cuando hoy en día existen distintos y muy diversos servicios ofrecidos en Internet, es el correo electrónico el primero en ser identificado y con el cual los internautas pueden identificarse para acceder al resto de servicios en la Red.

Sin embargo de la ubicuidad del correo electrónico y su funcionamiento, este sigue utilizando los mismos parámetros y configuraciones básicas con las que fue diseñado y publicado en 1982. A partir de esa fecha y hasta la actualidad, el correo electrónico ha sufrido importantes modificaciones, pero su núcleo de funcionamiento sigue siendo el mismo, lo que lo hace vulnerable a ser atacado.

El presente trabajo pretende realizar, mediante una reseña sencilla del funcionamiento de los sistemas de correo electrónico y de la tecnología de seguridad creada para establecer comunicaciones seguras entre los usuarios, un análisis de las vulnerabilidades inherentes del correo electrónico seguro. A partir de esta información, establecer un nuevo sistema de correo seguro basado en PGP, definiendo qué características deberían poseer para que su utilización y su funcionamiento vayan acorde con la seguridad nativa que pretenden brindar. Un sistema que aproveche eficientemente la tecnología pero con nuevos protocolos de usabilidad y confiabilidad para los usuarios que se comunican vía internet.

Es de vital importancia mantener los protocolos de seguridad de correo electrónico siempre actualizados, ya que sin importar su fortaleza práctica o sus métodos, Internet no para de evolucionar y exige cada vez más sistemas que sean seguros y fáciles de utilizar para los usuarios; que se alejen de los complejos modos de funcionamiento de las edades tempranas

de Internet. Este trabajo apunta a brindar una solución eficiente en ese sentido, en el que mejorar la seguridad no es solo fortalecer un sistema tecnológico, sino también hacerlo más fácil de usar para el usuario común.

Hipótesis

El establecimiento de un nuevo sistema de correo electrónico con seguridad nativa a través de sistemas de encriptación y seguridad como PGP permitirá que las comunicaciones puedan satisfacer los parámetros de seguridad que aplica la criptografía moderna en las comunicaciones brindando seguridad a los usuarios y a sus datos.

CAPÍTULO 1
DEFINICIONES GENERALES

1.1 CORREO ELECTRÓNICO [3][4]

Cuando ARPANET evolucionó en la actual Internet, hace más de 30 años, trajo consigo una aplicación que se convirtió pronto en el sinónimo de Internet en el mundo; el correo electrónico. A pesar que actualmente existan otro tipo de herramientas para comunicación, principalmente basadas en sistemas móviles, se sigue considerando al correo electrónico como parte fundamental y piedra angular de las comunicaciones, consideración que ha permitido que pueda seguir evolucionando y fortaleciéndose a través de los años.

Emula su funcionamiento al del correo postal regular, es decir, no necesita que la otra persona esté disponible al tiempo del envío de un correo electrónico para que pueda recibirlo, con la diferencia que el correo electrónico es casi instantáneo, fácil de enviar y distribuir y principalmente barato. La evolución de su propia tecnología y entorno de funcionamiento le ha permitido crecer desde el envío de textos en código ASCII en sus inicios hasta actualmente soportar el envío de casi cualquier tipo de archivos, sobretodo multimedia y permitir que estos mensajes puedan ser asegurados de tal manera que su contenido solo lo puedan conocer las personas a las que van dirigidos.

1.1.1 FORMATO

Un correo electrónico estándar está compuesto principalmente de tres partes claramente diferenciadas, las cuales son leídas por los servidores de correo electrónico, estas son:

- La envoltura, que contiene la dirección del destinatario, además de la prioridad del mensaje y la seguridad del mismo. Se crea en el servidor de correo, que será explicado posteriormente.
- El encabezado, que contiene la información correspondiente al remitente y que contiene las características de control de envío del mensaje.
- El cuerpo del mensaje, es el mensaje como tal, y contiene la información que se quiere transmitir.

Cuando se diseñó el sistema de correo electrónico se establecieron parámetros de configuración para el establecimiento de un correo, los cuales, con sus modificaciones siguen siendo parte de todos los correos usados actualmente.

1.1.1.1 Formato estándar

Definido en el RFC 822 y actualizado en el RFC 5322, establece las condiciones necesarias que debe cumplir un correo electrónico para que pueda ser leído y enviado por los servidores. Fue escrito en formato ASCII y define los campos que debe tener el encabezado del mensaje para su envío. Cada campo del encabezado, que se especifican en la tabla 1.1, consiste en una línea compuesta por caracteres ASCII. La actualización del protocolo permitió que se establezca una diferencia más clara entre la envoltura del mensaje (proporcionada por el protocolo de transferencia, que será explicado más adelante) y el encabezado del mensaje en sí.

Parámetro	Significado
To:	Dirección (o direcciones) del destinatario principal.
Cc:	Dirección (o direcciones) del destinatario secundario.
Bcc:	Dirección (o direcciones) del destinatario oculto.
From:	Persona que creó el mensaje.

Sender:	Persona que envía el mensaje.
Received:	Parámetro que agrega cada servidor intermedio por el que pasa el mensaje.
Return-Path	Servidores por el que ha pasado el mensaje

Tabla 1.1 Parámetros principales del encabezado según RFC 5322 [3]

El parámetro *Cc:* tiene simplemente una diferencia para la persona que redacta el mensaje, ya que los servidores consideran a todas las direcciones dentro de la misma prioridad de envío. La diferencia que hace *Bcc:* es la de permitir que terceros puedan recibir el mensaje sin que los destinatarios primarios o secundarios tengan conocimiento del envío. Los parámetros *From:* y *Sender:* permiten diferenciar quien crea y envía el mensaje, para que en caso de una respuesta, el servidor pueda direccionar la respuesta a la persona adecuada. Habitualmente contienen la misma dirección.

Los parámetros *Received:* y *Return-Path:* son completados por el servidor de destino, o por los servidores intermedios por los que ha pasado el mensaje para indicar la forma en la que el mismo ha viajado, y el cómo reconstruir un camino de regreso. Adicionalmente los parámetros principales, existen otros no son modificados por el usuario (especificados en la tabla 1.2) y que se definen en el Agente según sus requerimientos o de los destinatarios.

Parámetro	**Significado**
Date:	Fecha y hora de envío del mensaje.
Reply-To:	Dirección a la que se deben enviar las respuestas.
Message-Id:	Identificador único de cada mensaje.
In-Reply-To:	Identificador del mensaje al que éste responde.
References:	Otros identificadores de mensaje relevantes.

Keywords:	Palabras clave seleccionadas por el usuario.
Subject:	Resumen corto del mensaje para desplegar en una línea.

Tabla 1.2 Parámetros adicionales del encabezado según RFC 5322 [3]

A partir del boom multimedia de Internet, se hizo palpable que este formato hacía imposible el envío de otra cosa que no sea texto, por lo que se diseñaron extensiones para que video o imágenes puedan ser enviadas por correo electrónico, surgiendo MIME.

1.1.1.2 Extensiones Multipropósito de Correo en Internet

Las Extensiones Multipropósito de Correo en Internet (*Multipurpose Internet Mail Extensions*, o MIME) fueron creadas como respuesta a las necesidades de la masificación del correo electrónico, tales como el envío de material multimedia y el soporte a idiomas con caracteres que no estaban contenidos en ASCII. Fue definido en los RFC 2045-2047, 4288, 4289 y 2049.

Funcionan bajo el mismo esquema del RFC 5322, modificando el encabezado para permitir establecer una estructura adecuada al cuerpo del mensaje que soporte los tipos de mensajes que no sean necesariamente ASCII. Al mantener la estructura original, permitió escalar el sistema de correo sin tener que migrar el núcleo de la red de correo electrónico que llevaba funcionando durante algunas décadas.

MIME agrega nuevos parámetros al encabezado original, los cuales se muestran en la tabla 1.3, y que especifican que tipo de extensión va a ser utilizada al momento del envío. Si el parámetro de versión va vacío, se considera que el mensaje a enviar es un mensaje ASCII clásico.

Parámetro	Significado
MIME-Version:	Identifica la versión MIME.
Content-Description:	Describe el contenido del mensaje en un formato legible.
Content-Id:	Identificador único de cada mensaje
Content-Transfer-Encoding:	Tipo de codificación del contenido del mensaje.
Content-Type:	Tipo y formato del contenido.

Tabla 1.3 Parámetros de encabezado agregados por MIME [3]

El protocolo de transferencia de correo electrónico estaba diseñado solamente para transportar mensajes en ASCII, y cuyas líneas tenían la restricción de no poder contener más de 1000 caracteres, donde cada carácter se representaba por 7 bits de cada byte. Los formatos binarios (documentos, imágenes, vídeo) están representados por los 8 bits de cada byte, por lo que el desafío en la creación de MIME era que, durante el transporte de los mensajes, estos datos binarios pudieran ser codificados como caracteres ASCII regulares.

Para este cometido MIME cuenta con cinco esquemas de codificación, definidos en el campo *Content-Transfer-Encoding*, donde el más simple es el que debe enviar caracteres ASCII, siempre y cuando estos formen líneas que no superen los 1000 caracteres. A esta codificación MIME la conoce como *7bit*. Cuando esto ocurre MIME no hace ninguna codificación y los envía como un correo regular con formato RFC 5322.

El segundo esquema, conocido como *8bit*, es similar al primero, pero hace uso de los 8 bits, es decir ASCII extendido, manteniendo aún el límite de caracteres por línea.

El tercer esquema, debe ya utilizar una codificación binaria, que no se adhiere ni a la limitación de caracteres o de longitud de línea. Para poder

codificarlos, MIME utiliza un sistema de codificación conocido como *base64*, donde se hacen bloques de 24 bits divididos a su vez en cuatro bloques de 6 bits, donde se representan los primeros caracteres ASCII. Los retornos de carro y avances de línea, que se explicaran posteriormente, permiten que las líneas sean aún más cortas, facilitando la transmisión. Es la forma más extendida de envío de datos binarios, aun cuando actualmente los servidores provean soporte para estos datos. A pesar de su masificación, no es demasiado eficiente.

El cuarto esquema se utiliza cuando se quieren enviar caracteres ASCII mezclados con unos pocos caracteres no-ASCII, y la codificación base64 es ineficiente. Es cuando se recurre a la codificación conocida como entrecomillada imprimible o *Quoted printable*, en inglés, que codifica a los caracteres ASCII por sobre 127 con un signo de igual (=) seguido por la representación hexadecimal del carácter, y así se codifican todos los caracteres de control existentes.

Finalmente, el quinto esquema, conocido como *binary*, queda para datos binarios arbitrarios que no se correspondan a ninguna de las codificaciones anteriores, donde le usuario podría especificar una codificación alternativa para estos datos.

A pesar de haber sido diseñado para correo electrónico, la definición de tipo de contenido de MIME se ha extendido hasta representar una etiqueta que le informa a los navegadores o aplicaciones derivadas que tipo de archivo está siendo descargado o procesado para saber cómo presentarlo. Originalmente se definieron siete tipos, que se indican en el campo *Content-Type* del encabezado y se describen en la tabla 1.4; cada uno con sus subtipos, separados en el encabezado por una diagonal

Tipo	Subtipos	Descripción
text	text plain, html, xml, css	Texto en diversos formatos.
image	image gif, jpeg, tiff	Imágenes.
audio	audio basic, mpeg, mp4	Sonidos.
video	video mpeg, mp4, quicktime	Películas.
model	model vrml	Modelo 3D.
application	application octet-stream, pdf, javascript, zip	Datos producidos por aplicaciones.
message	message http, rfc822	Mensaje encapsulado.
multipart	multipart, mixed, alternative, parallel, digest	Combinación de múltiples tipos.

Tabla 1.4 Tipos y subtipos de contenido MIME [3]

Mediante estas extensiones han permitido la extensión del correo electrónico para la distribución de diversos tipos de información que no es necesariamente texto, lo cual también ha abierto una brecha de seguridad, ya que muchos de estos 'programas arbitrarios' pueden contener código malicioso. Cuando un mismo correo tiene distintos tipos de archivos externos, se debe indicar cada uno de ellos por separado, dándole a cada uno su propio encabezado, donde se especifique claramente el *Content-Transfer-Encoding,* como se muestra en la figura 1.1.

```
To: Karen Nobody <karen@192.168.0.35>
From: Nemo <nemo@192.168.0.35>
Subject: Mensaje plano con adjuntos
Message-ID: <55B7B73F.70006@192.168.0.35>
Date: Tue, 28 Jul 2015 14:09:19 -0300
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:38.0) Gecko/20100101
 Thunderbird/38.1.0
MIME-Version: 1.0
Content-Type: multipart/mixed;
 boundary="------------030705050505030203040003"

This is a multi-part message in MIME format.
--------------030705050505030203040003
Content-Type: text/plain; charset=utf-8
Content-Transfer-Encoding: 8bit

En un agujero en el suelo, vivía un hobbit. No un agujero húmedo, sucio,
repugnante, con restos de gusanos y olor a fango, ni tampoco un agujero
seco, desnudo y arenoso, sin nada en que sentarse o que comer: era un
agujero-hobbit, y eso significa comodidad.
Tenía una puerta redonda, perfecta como un ojo de buey, pintada de
verde, con una manilla de bronce dorada y brillante, justo en el medio.
John Ronald Reuel Tolkien.

--------------030705050505030203040003
Content-Type: image/jpeg;
 name="Hobbit.jpg"
Content-Transfer-Encoding: base64
Content-Disposition: attachment;
 filename="Hobbit.jpg"
```


```
/9j/4AAQSkZJRgABAQEASABIAAD/2wBDAAoHBwgHBgoICAgLCgoLDhgQDg0NDh0VFhEYIx8l
JCIfIiEmKzcvJik0KSEiMEExNDk7Pj4+JS5ESUM8SDc9Pjv/2wBDAQoLCw4NDhwQEBw7KCIo
Ozs7Ozs7Ozs7Ozs7Ozs7Ozs7Ozs7Ozs7Ozs7Ozs7Ozs7Ozs7Ozs7Ozs7Ozs7Ozs7Ozv/wAAR
CAMgAf8DASIAAhEBAxEB/8QAHAAAAgMBAQEBAAAAAAAAAAAAAAWQCBQYBBwAI/8QAThAAAgED
AwIEAwcCBAIGBgkFAQIDAAQREiExBUETIlFhBjJxBxQjQoGRsVKhFWLB0TNDJCVjcrLhFhdT
dJLwJjQ1NkRkc4Ki8SdUZYP/xAAaAQADAQEBAQAAAAAAAAAAAAABAgMABAUG/8QAMREAAgIC
AgEEAgEDAwMFAAAAAAECEQMhEjFBBBMiUTJhcRQzgUKhsSMk8FJiweHx/9oADAMBAAIRAxEA
PwDyOJNLDPbNNI6aTpLH13pbBVz65pqNQNwNz7VdiknVhjzBiew7UMr5gMjPvU86T335roAO
Sc7e9YxDwwBjIydtqhhVA527Zo5jLKDsCKWaIljqcnPoKwSQk9PL/rXxJAyfTau+Gq5BG49T
Xz6Yxx2yADWMQC+UY8xoRJByO/Nc1jPc+tERQcMcgdqxiaRgb8HHeuBTI+lTnfc18z5IVSce
5ogIiTYZY9h3rGJaQi6EoqRBQDX0ChCGK5I7GildbDAGPagYHGTJsBwcU0iADb0rmkLj9vrT
(...)
ttv3BpKTp8PGr+9XSj9EXJ9Cst+7n396A1w753NOnp8XOo1IWluq75NUTQtlUQzHYV3w0HJL
H0WrBoYlOBH9M0OSRFxgAfSjZgAjkK4VQgPc80ZLRM+JNIWPoa4Z1C+XmhNK7Ntv9a2wlhHc
JFsgG3tUXvSNtWKS16R8xJ9qgG3zgk5+tCjDqTs0ibk5YV6Knyj6V5lbvi5hRjvqGQN8V6an
A+lJIxh7OUNCozvvxTSzqme+f7VUW5uLMMk9vIugk50/vTySxyxhg2CeRjes0YejnVhtv/pR
hLkE5FVwXw2DKrHPfbepeFO+WGFH96UJZRy++aKJ8VVIt2PlJb6CjI10xGqFuewpWglkJVPF
dJ+mKR03KjBglx66aks0gOGRh9RS0Gx3G/NS1kd9hQUkYjBU5PrU2jkK5CH9qShrCrJg71LW
MZ3zSh8QHHhtke1dHinYI37UKDYxqOaIk2wU0COGZhuhHuaL4Djkrj60riMpBDKVIHIogl22
pcRvjcD96IttKQcMv70OIeQRnzwa54mBSz+NCcuhAPfGa596hOQZFH1NDiHlYOJiRzUhMc5z
tSRkTswNfGUKOa3EPJj6znbeifePU1Vi5A2G59qkL2PGxyfSg4B5ssWn96i1x5RvSBuUOAzY
z71B7tAeaygZzHzKTXGl2pI3CkDB5qQY6dwaPEVyJvJzQtfeoNJnionJGadIRsYSQetE8b0N
KIRg7gVJJUBOph2rcQmQHeg6yT5RmvjJjkE
VqNY0JsDbau/eANywpJ7hV4IyaisoJ9TSuIyY742SDnFfNPgH1pPxk7HNCNwmAcj96HEaxtZ
DzXXkOeQKUFwpG52ztUTODnejxiIyY742SDnFfNPgH1pPxk7HNCNwmAcj96HEaxtZ
```


Figura 1.1 Ejemplo de mensaje con diferentes contenidos multimedia

1.1.2 ARQUITECTURA

Para la construcción de los mensajes tal como se los explico previamente se necesitan sistemas que interactúen con el usuario y a su vez, sistemas que interactúen entre sí para el envío de la información, estas dos sub-estructuras comparten características y establecen un trabajo conjunto mediante distintos de protocolos.

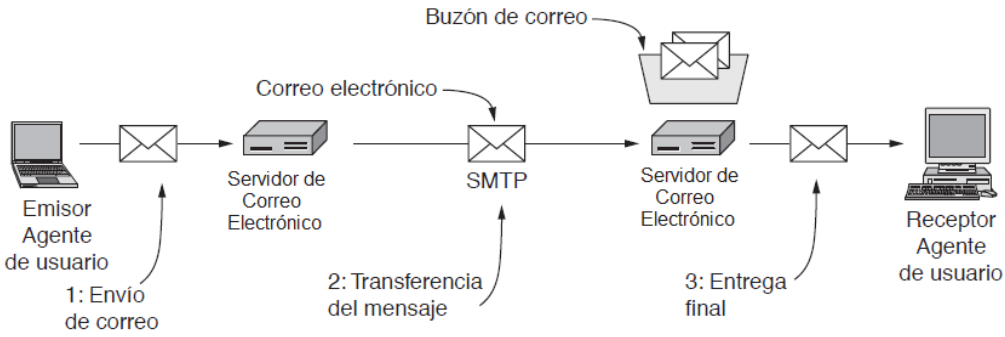


Figura 1.2 Diagrama esquemático de la arquitectura de un sistema de Correo Electrónico [3]

1.1.2.1 Agente de Usuario

El agente de usuario, comúnmente conocido como lector de correo o cliente de correo, es un programa diseñado para redactar, enviar y responder mensajes, así como para leer los mensajes nuevos que han llegado la dirección de correo asignada. El agente de usuario, o UA (User Agent, por sus siglas en inglés) posee además características que le permiten manipular el correo entrante, asignándole etiquetas o re direccionándolo a buzones secundarios, que son mostrados al usuario como carpetas.

El UA tiene además funcionalidades avanzadas según sea su configuración y utilidad, como la de generar respuestas automáticas según el caso y utilidad, como por ejemplo cuando el usuario se encuentra de vacaciones o fuera de su sitio de trabajo. Además permiten configurar las características especiales de RFC 5322 para cada mensaje o grupo de mensajes y administran las seguridades necesarias definidas por el remitente y el destinatario.

Para que el mensaje pueda ser enviado, aparte de estar construido como se explicó previamente, debe tener una dirección de destino. Esta puede representar a un usuario individual o una lista de correo, según estén configuradas. Sin esta dirección el servidor no puede construir el sobre para enviar el mensaje. La forma de direccionamiento más habitual es:

nombre-de-usuario@servidor-de-cuenta.

Donde ‘@’ se usaba como separador para diferenciar el nombre del servidor. Este símbolo se planteó porque ningún nombre o apellido (en inglés) poseía este carácter. Técnicamente se lee:

nombre-de-usuario ‘en’ *servidor-de cuenta*

Considerando que cada nombre de usuario debe ser único dentro de casa proveedor. Una vez definido la dirección del destinatario el mensaje pasa al Agente de transferencia, mejor conocido como servidor de correo.

1.1.2.2 Servidor de Correo (Agente de transferencia)

Para que el mensaje adecuadamente construido llegue hasta el servidor del remitente y a partir de éste al servidor del destinatario es necesario hacer uso un protocolo que es el núcleo del correo electrónico en internet, el Protocolo Simple de Transferencia de Correo (*Simple Mail Transfer Protocol*, o SMTP, por sus siglas en inglés).

SMTP, definido en el RFC 521 y actualizado en el RFC 5321, define la forma de comunicación entre cliente (remitente) y servidor (destinatario) para que los mensajes sean enviados. Como la mayoría de los protocolos de Internet, está escrito en ASCII simple, siendo esta característica, la simplicidad, la que ha permitido la realización de pruebas y depuraciones del protocolo ya que permiten que se hagan manualmente desde una consola conectada al puerto específico, permitiendo encontrar errores de manera más sencilla.

Para transferir un mensaje, el cliente se comunica con el puerto 25 del servidor, que es el que escucha solicitudes SMTP. Previo al envío de mensajes, cliente y servidor realizan el establecimiento de la conexión donde se intercambian los parámetros en común que tienen el cliente y el servidor

para la conexión, además de las direcciones origen y destino del mensaje. El servidor hace uso de la dirección destino para construir el sobre del mensaje y enviarlo a través de internet. Si el cliente tiene más mensajes para enviar, la conexión se mantiene, caso contrario se da por terminada una vez enviado el último mensaje.

El ejemplo de la figura 1.3 muestra como el servidor (S) y el cliente (C) establecen la conexión y transmiten mensajes través de sockets TCP para que el correo pueda ser enviado del cliente al servidor.

El Cliente se comunica con el servidor mediante comandos específicos que sirven para determinar las acciones que requiere del servidor, tales como HELO (abreviatura de *Hello*, el saludo del cliente), MAIL, FROM, RCTP TO, DATA y QUIT; cada uno de ellos puede explicarse por sí mismo. Cada una de estas líneas termina con el comando <CRLF>, donde CR representa el retorno del carro (*Carriage Return*) y LF alimentación de línea (*Line Feeding*), comandos que sirven para emular el funcionamiento de una máquina de escribir y le indican al servidor que la línea a terminado y que comenzará una nueva. El cliente además envía una línea que contiene solo un punto (.) para indicarle al servidor el fin del mensaje. Dentro de la configuración ASCII del protocolo, esta línea final es <CRLF>.<CRLF>. A partir de esta línea el cliente puede iniciar el envío de más mensajes de correo o cerrar la conexión.

```
        S:220 mail.undomiel.com ESMTP Postfix (Ubuntu)
C:EHLO [192.168.0.33]
        S:250-mail.undomiel.com
        S:250-PIPELINING
        S:250-SIZE 10240000
        S:250-VRFY
        S:250-ETRN
        S:250-ENHANCEDSTATUSCODES
        S:250-8BITMIME
        S:250 DSN
C:MAIL FROM:<nemo@192.168.0.35> SIZE=834
        S:250 2.1.0 Ok
C:RCPT TO:<karen@192.168.0.35>
        S:250 2.1.5 Ok
C:DATA
        S:354 End data with <CR><LF>.<CR><LF>
C:From: Nemo <nemo@192.168.0.35>
C:To: Karen Nobody <karen@192.168.0.35>
C:Subject: Mensaje plano
C:X-Enigmail-Draft-Status: N1110
C:Message-ID: <55B7B5AF.5020001@192.168.0.35>
C:Date: Tue, 28 Jul 2015 14:02:39 -0300
C:User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:38.0) Gecko/20100101
C: Thunderbird/38.1.0
C:MIME-Version: 1.0
C:Content-Type: text/plain; charset=utf-8
C:Content-Transfer-Encoding: 8bit
C:
C:En un agujero en el suelo, vivía un hobbit. No un agujero húmedo, sucio,
C:repugnante, con restos de gusanos y olor a fango, ni tampoco un agujero
C:seco, desnudo y arenoso, sin nada en que sentarse o que comer: era un
C:agujero-hobbit, y eso significa comodidad.
C:Tenía una puerta redonda, perfecta como un ojo de buey, pintada de
C:verde, con una manilla de bronce dorada y brillante, justo en el medio.
C:John Ronald Reuel Tolkien.
C:.
        S:250 2.0.0 Ok: queued as DF54A212
C:QUIT
        S:221 2.0.0 Bye
```

Figura 1.3 Ejemplo de un envío de mensaje SMTP

El servidor de correo, en respuesta a los comandos del cliente, devuelve mensajes que están precedidos por un código que muestra su funcionamiento y ocasionalmente una pequeña explicación.

A partir del RFC 5321, SMTP agrego diversas extensiones para corregir ciertas falencias que tenía el protocolo original. A estas extensiones, en general, se las conoce como ESMTP (SMTP Extendido, o Extended SMTP, por sus siglas en inglés). Para invocar estas funciones, en lugar de HELO el cliente envía EHLO, lo que hace que el servidor responda con un set adicional de características, descritas en la tabla 1.5, para que el cliente las conozca y pueda hacer uso de ellas según sus requerimientos.

Característica	**Descripción**
AUTH	Autentificación del cliente.
BINARYMIME	El servidor acepta mensajes binarios.
CHUNKING	El servidor acepta mensajes extensos en partes.

SIZE	Verifica el tamaño del mensaje antes de intentar enviarlo.
STARTTLS	Cambiar a transporte seguro vía TLS.
UTF8SMTP	Direcciones internacionalizadas.

Tabla 1.5 Características adicionales de ESMTP [3]

1.1.3 PROTOCOLOS DE RECEPCIÓN.

Dentro de sus características, SMTP fue diseñado para ser un protocolo de empuje, es decir, no está diseñado para solicitar información de un repositorio. Durante los primeros años del correo electrónico, cuando el agente de usuario y el servidor de correo eran procesos diferentes de un mismo equipo, SMTP 'empujaba' el correo entrante a un archivo específico donde el usuario podía consultarlo. Sin embargo, con la introducción de las computadoras personales y la movilidad los agentes de usuario debían solicitar la información que había sido empujada al servidor de correo, cosa que SMTP no estaba facultado para realizar. Para este cometido se diseñaron protocolos que transmiten los mensajes del servidor hacia el cliente bajo demanda. Entre los más importantes están:

1.1.3.1 POP3

Definido en el RFC 1393, el Protocolo de Oficina de Correos Versión 3 (Post Office Protocol Ver. 3, por sus siglas en inglés) es un protocolo sencillo y limitado de acceso de correo que permite recibir en el agente de usuario los mensajes recibidos. Cuando inicia la conexión a través del puerto 110, envía las credenciales de autenticación al servidor para poder identificar al usuario. Cuando ha sido autenticado, el protocolo procede a 'descargar' todos los mensajes nuevos hacia el agente de usuario, etiqueta los mensajes descargados para su eliminación y maneja estadísticas simples de uso. Cuando el cliente decide terminar la comunicación, el protocolo elimina

del servidor todos los mensajes marcados y descargados, dejando únicamente las copias de los mensajes dentro del agente de usuario.

Un problema existente con esta configuración es la limitación de lectura de los mensajes solamente en el agente de usuario en el que fueron descargados, eliminando la posibilidad de ubicuidad de comunicación. Para resolver este y otros problemas, se creó IMAP.

1.1.3.2 IMAP

Definido en el RFC 3501, el Protocolo de Acceso a Mensajes de Internet (Internet Messages Access Protocol, por sus siglas en inglés), presenta la evolución de POP3 ya que permite mantener copias de los mensajes recibidos en el servidor de correo y en cuantos agentes de usuario estén conectados correctamente al mismo, además de permitir la organización de los mensajes entrantes en distintos buzones o 'carpetas', y manteniendo esta configuración en todos los agentes de correo conectados. Adicionalmente IMAP permite que el usuario reciba extractos de sus mensajes recibidos, como las cabeceras o partes específicas de adjuntos MIME, según sus requerimientos.

Además de estos protocolos, existen protocolos propietarios, como Microsoft Exchange, que cumplen con las mismas funciones, dentro de entornos cerrados y privados.

1.2 ANONIMATO EN INTERNET [1][3][8]

Una de las ventajas de Internet fue la de permitir el libre intercambio de información entre todos los internautas, lo que con el boom de la web hizo posible la creación de foros y de listas de correo electrónico donde las personas intercambiaban opiniones. Es por esto que Internet dejó de ser un

ente meramente tecnológico para tornarse un ente social, donde son las personas las que influyen directamente en la comunicación.

La privacidad se ha tornado un derecho inalienable de las personas y con el auge de internet los conceptos que se manejaban sobre privacidad y anonimato adquirieron nuevas definiciones. Sobre todo si Internet es usado para realizar denuncias de actos de corrupción y de abuso de poder por parte de las autoridades, o denunciar las violaciones de los derechos humanos de perseguidos políticos y disidentes de los gobiernos totalitarios que aún mantienen prisioneros a sus ciudadanos.

Aun así, la privacidad en Internet es un objetivo muy difícil de conseguir, por la cantidad de información almacenada en servidores que, consciente o inconscientemente, hemos entregado y que podría permitirle a otros internaturas y a agencias gubernamentales obtener información acerca de nuestras vidas. A este panorama se añaden los requerimientos cada vez más restrictivos de los estados en cuestiones de información no privada en aras de ‘salvaguardar la seguridad de los estados y sus ciudadanos’. La línea entre seguridad preventiva y vigilancia es realmente muy difusa y delgada.

La criptografía permitió un cambio de enfoque en el manejo de la privacidad, ya que cualquier persona que instale un sistema de cifrado de información (como PGP, que será explicado posteriormente) y hace uso de claves considerablemente fuertes asegurará sus archivos y comunicaciones sin preocuparse de la vigilancia de la que pudiera estar siendo objeto. El control de la tecnología de seguridad por parte de los gobiernos ha hecho que muchas veces soluciones de seguridad no puedan ser exportadas a otros ambientes para que los ‘secretos’ de creación no cayeran en manos indebidas.

La criptografía permite el envío seguro de información entre dos partes, pero ocurre en ocasiones que la mejor forma de mantener la

privacidad es evitar la autenticación, manteniendo la comunicación anónima. El anonimato es usado principalmente para proteger la seguridad de denunciantes de actividades ilegales o activistas dentro de gobiernos represores. Además, es también útil cuando el discutir acerca de tópicos en donde las ideas deben ser privilegiadas por sobre quien las ha expresado. Muchos de las listas de correo de los años 90's que mantenían esta modalidad permitieron la creación de nueva tecnología libre, como por ejemplo Bitcoin.

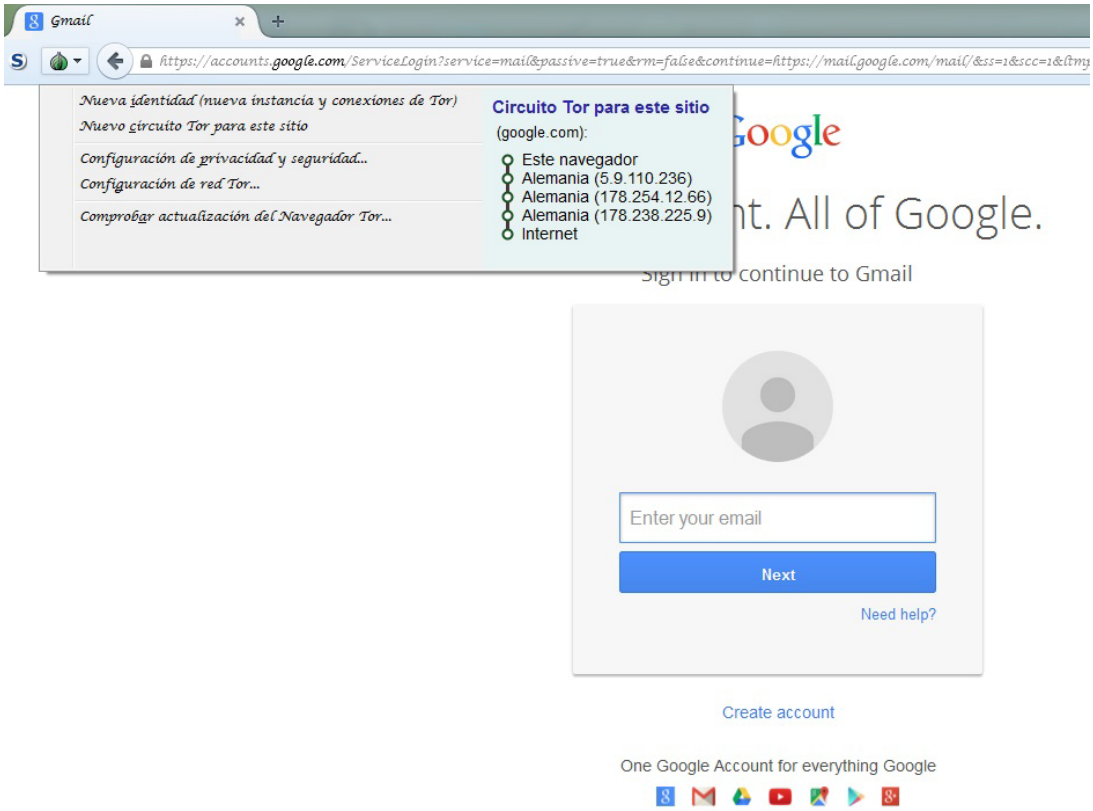


Figura 1.4 Circuito de conexión de TOR para Google (google.com) en Windows

Muchos sistemas de tecnología, además de la criptografía permiten mantener el anonimato en Internet, tales como redes anónimas como TOR [11]. Basado en la navegación por túneles virtuales o capas desarrollado por la marina estadounidense, mantiene las comunicaciones anónimas mientras viajan a través de esta red porque evitan el análisis de tráfico, además de proteger las direcciones origen y destino de los usuarios mediante el uso de sus propias direcciones encriptadas (.onion). Mantener la privacidad y el

anonimato en las comunicaciones es parte fundamental del uso de internet, y los desarrollos de la tecnología indudablemente están encaminados en esa dirección.

CAPÍTULO 2
SOLUCIONES ACTUALES DE CORREO ELECTRONICO SEGURO

2.1 REQUISITOS DE SEGURIDAD [1][5]

Aun cuando se pudiera suponer lo contrario en la actualidad, los sistemas de correo electrónico no implementan la mayoría de características que comprenden la propia definición de seguridad de la información. Incluso los sistemas globales diseñados especialmente solo proveen algunas de estas características.

Dependiendo de las necesidades propias de los usuarios, la implementación de una o varias de estas características en los sistemas de correo electrónico es lo que diferencia lo servicios existentes en internet.

- ***Privacidad:***

Característica que permite mantener la información enviada o distribuida para que sólo la persona autorizada a leerla pueda hacerlo, por lo que se considera que esta información no puede ni debe ser revelada a terceros.

La idea es evitar que fisgones, sistemas espías o nodos de retransmisión configurados puedan acceder al contenido del mensaje, provocando un conflicto de seguridad ya que se asume que el contenido de un mensaje, sea el que fuere, solo pertenece a su destinatario.

- ***Autenticación:***

Capacidad del destinatario de saber efectivamente si el remitente es quien dice ser, comprobando su identidad de manera confiable. La autenticación está fuertemente ligada con la integridad del mensaje, por lo

que habitualmente los proveedores ofrecen ambos servicios juntos. Estos servicios están ligados al uso de claves, sean estas públicas o privadas.

- ***Integridad:***

Comprobación por parte del destinatario que la información enviada no ha sido modificada ni por el remitente ni por terceras partes mientras el mensaje era transmitido.

A pesar que la integridad está ligada a la autenticación, existen escenarios en los que es necesario que el mensaje sea considerado como válido sin necesidad de autenticar al remitente. Sin un sistema criptográfico esto no tiene sentido, pero a través de la clave pública del destinatario, el mensaje puede ser autentificado sin que necesariamente su fuente lo sea.

- ***No Repudio:***

Incapacidad del remitente de negar haber enviado un mensaje. Significa que el destinatario tiene la capacidad de probar, ante terceras personas, que el mensaje que recibió fue en efecto enviado por el remitente.

Esta propiedad de la seguridad no siempre es necesaria, ya que en escenarios donde la información transmitida procede de eventos ilegales (como una investigación periodística) el remitente debería tener la posibilidad de negar el envío de la información si su vida podría estar en riesgo. Es comúnmente aceptado que para probar que el remitente envió el mensaje, el mismo tendría que estar firmado por el remitente. Existen técnicas que permiten enviar un mensaje de manera que el destinatario sepa que el remitente se lo envió, pero que no pueda probarlo ante terceras personas, por negación plausible.

- ***Prueba de envío y Prueba de entrega:***

Notificaciones para el remitente que indican que el sistema envió el mensaje y lo entregó a su destinatario y los tiempos en los cuales se realizaron estos procesos. Habitualmente el sistema informa, mediante

resúmenes cifrados del mensaje que el mismo fue enviado cuando pasa por algún nodo identificador o el nodo final, e informa la entrega cuando el destinatario, mediante su firma, indica que el mensaje fue entregado.

- ***Anonimato:***

Se presentan ocasiones cuando es necesario que el destinatario no conozca al remitente, sin embargo esto no es tan fácil como se aparenta, ya que muchos sistemas incluyen dentro de los metadatos del mensaje la información del remitente, o la capa de red almacena la dirección IP del remitente, haciendo posible por un análisis posterior determinar, si no la identidad, al menos el origen del mensaje, lo cual podría ya ofrecer pistas para conocer la identidad del remitente.

Una de las formas más seguras de asegurar el anonimato de un mensaje es también una de las formas de mejorar la confidencialidad en el flujo del mensaje, y es usar distintos nodos intermedios entre el origen y el destino del mensaje en cuestión, de modo que, cuando llegue al destinatario, éste solo sepa que el mensaje es para el porqué está cifrado por su clave pública, pero que los metadatos ni las direcciones de red puedan proporcionar efectivamente información del remitente original. Cabe aclarar que estos nodos intermedios no añaden ninguna autenticación o información al mensaje en sí, solo habilitan su paso.

- ***Auditoria:***

El manejo de eventos en el momento del uso del sistema que podrían ser de relevancia para la seguridad, como fechas de envío y destinatarios.

- ***Control de uso:***

La capacidad del sistema de correo electrónico de mantener estadísticas de uso, para gestión de usuarios y el uso que le dan a su cuenta de correo.

2.2 PROTOCOLOS SEGUROS [1][3][4][5]

Casi al tiempo del desarrollo del correo electrónico, y a pesar de no estar universalizado, surgió la necesidad de implementar seguridades a los mensajes que se enviaban a través de ARPANET al principio, y luego a través de internet.

Estas formas de seguridad evolucionaron en protocolos que de una u otra manera respondieron a la necesidad de proveer seguridad a los mensajes enviados y por enviar. A continuación se explicaran algunos de los más notorios e importantes.

2.2.1 PEM

Correo con privacidad mejorada (Privacy Enhanced Mail o PEM, por sus siglas en inglés) fue desarrollado a finales de los años 80's y principios de los 90's. Añade autenticación de origen, integridad y cifrado a mensajes compuestos por texto plano, es decir antes del surgimiento de las necesidades multimedia de internet. Está definido por los RFC 1422, 1423 y 1424. Fue diseñado como un sistema añadido al agente de usuario del remitente y el destinatario, asumiendo que el mensaje ira a través de una infraestructura diseñada solo para el envío de mensajes en texto plano.

Su funcionamiento está basado principalmente en PKI, tratando de uniformizar las diferentes Autoridades Certificantes (Certificate Authority o CA, por sus siglas en inglés) de la época en un sistema que permita asegurar las comunicaciones a través de claves públicas y privadas, asumiendo que estas serían certificadas por alguna entidad extremadamente confiable. Todo este sistema fue implementado antes del diseño de PKI como se conoce actualmente, por lo que la infraestructura de certificados que había sido diseñado para PEM resultó inútil para una implementación a gran escala. Lo que hizo que el sistema implementara su propio sistema de jerarquía de certificados que le permitía funcionar. Además, implementó una

solución muy similar a los sistemas de directorio para que los usuarios puedan acceder a las CRL's de PEM basado en mensajes dirigidos a los repositorios establecidos por la jerarquía propia de PEM, pero el sistema como tal no estaba diseñado para entregar tampoco esta información.

Para el envío de mensajes seguros, PEM generaba una clave numérica al azar, conocida como per-clave, con la cual cifraba el mensaje a transmitir y le agregaba control de integridad. Para el envío de esta per-clave, se la cifraba con otra clave conocida como clave de intercambio, que podía ser la clave pública del destinatario, o una clave compartida por los comunicantes que habría sido establecida previamente. Para usar la clave pública, PEM asumía que los comunicantes habían asegurado su propia cadena de certificados para validar la identidad de sí mismos, la cual viajaba en conjunto con el mensaje; mientras que para el uso de clave simétrica, PEM no ofrecía ningún tipo de comunicación adicional de establecimiento. Si quería utilizarse algún sistema simétrico, los comunicantes tenían que encontrar la forma de establecer una clave compartida antes de usar PEM.

Para el cifrado de los mensajes, PEM habilitaba, además de la per-clave, un vector de inicialización de 64 bits escogido aleatoriamente. La razón del vector, al momento de cifrar diferentes mensajes con diferentes per-claves, es la de hacer más fuerte el sistema frente a ataques de repetición, suponiendo que muchos mensajes enviados tuvieran los mismos caracteres de inicio (como un saludo en una carta formal). Además protegía a los mensajes de ataques de suplantación de texto para poder obtener la clave usada en el mensaje. Para mantener la autenticación de fuente y la integridad de los mensajes, PEM habilitaba el uso de MD2 hasta que fue dado de baja, y luego MD5. Cuando utilizaba sistema de claves asimétrico, además añadía la firma del mensaje a través de la clave privada del remitente.

PEM fue diseñado para viajar a través de la red como un correo electrónico ordinario, solo definiendo dentro del mensaje los campos que

hacían parte del mensaje cifrado, haciendo uso de la combinación de caracteres base64, para uniformizar los mensajes y evitar que el sistema destino identifique erróneamente los caracteres del mensaje. La cabecera del mensaje no es tomada en cuenta para el cifrado, el cuerpo se empaqueta dentro de líneas de comandos específicas como se muestra en la figura 2.1, y a partir de estas las características que deberá procesar el programa del destinatario.

```
From: Alice
To: Bob
Subject: Keep this proposition private!
Date: Fri, 01 Apr 94 10:10:31 -0400

-----BEGIN PRIVACY-ENHANCED MESSAGE-----
Proc-Type: 4,ENCRYPTED
Content-Domain: RFC822
DEK-Info: DES-CBC,31747476B4831B1D
Originator-ID-Asymmetric: MEMxCzAJBgNVBAYTAlVTMSYwJAYDVQQKEx1EaWd
 pdGFsIEVxdWlwbWVudCBDb3Jwb3JhdGlvbjEMMAoGA1UECxMDTEtH,02
Key-Info: RSA,Pv3W7Ds86/fQBnvB5DsvUXgpK7+6h5aSVcNeYf9uWtly9m2VHzC
 v2t7A6qgbXtIcf4kaMj1FL2yl9/N9mWpm4w==
MIC-Info: RSA-MD5,RSA,FUiVRM3x5Ku0aZveGIJ1hv/hi3Iowpm1iypd9VP7MGw
 PPQra+42TkbR/2jhnqXyVEXLaJ7BSyNhBh/9znIUj5uk0N7IXeBxX
Recipient-ID-Asymmetric: MEMxCzAJBgNVBAYTAlVTMSYwJAYDVQQKEx1EaWdp
 dGFsIEVxdWlwbWVudCBDb3Jwb3JhdGlvbjEMMAoGA1UECxMDTEtH,05
Key-Info: RSA,dpUp7/QoY9YOZzZCVcIwxIDMN0WbGCFAGN3T+xlV/0pBu2n5+x8
 PmBvMUN0NcBi5vtqBS4cfmgShiK0I4zu05Q==

21OHDuHTP5BABnlsqENz1WVerZxxWo2AsPHhm2SIz9qpLMvxT/x0+8UEf8dDsTDr
wbQo9/x+
-----END PRIVACY-ENHANCED MESSAGE-----
```

Figura 2.1 Ejemplo de mensaje enviado vía PEM. [4]

Como la figura 2.1 muestra, las líneas de empaquetamiento comienzan con el carácter (-), por lo que cualquier línea interna que tuviera el carácter guion como parte del mensaje se edita para que el destinatario no tenga conflictos de lectura, como ejemplifica la figura 2.2. Esta modificación del mensaje permitía también, de una manera indirecta y poco eficiente, evitar que el remitente de forma maliciosa incluya en el cuerpo del mensaje los caracteres propios de PEM enviando el mismo sin cifrado.

```
- --Sincerely, Alice--  En Lugar De  --Sincerely, Alice--
```

Figura 2.2 Modificación de caracteres en PEM para envío de mensajes. [4]

Dentro de los límites del mensaje PEM, todos los campos del mensaje son texto, algunas veces especificado en la cabecera como cifrado, otras como codificado, pero para el envío el formato texto de los sistemas de correo electrónico original se mantiene. El usuario a simple vista no podría diferenciar el mensaje cifrado del codificado, ya que ambos para el envío se muestran como ilegibles, pero son los parámetros propios de cada mensaje los que hacen que el sistema interprete el mensaje y lo vuelva texto legible directamente o previa la utilización de una clave. Las tablas 2.1, 2.2, 2.3 y 2.4 ejemplifican los nombres y las características de estos campos.

Campo	Descripción
Proc-Type: 4,ENCRYPTED	Tipo de mensaje PEM (tipo, versión)
Content-Domain: RFC822	Tipo de mensaje enviado
DEK-Info: DES-CBC (16 dígitos hexadecimales)	Algoritmo de cifrado del mensaje
Originator-Certificate: (ilegible)	Certificado codificado del remitente (opcional)
Originator-ID-Asymmetric: (ilegible),(numero)	Identificador del remitente ID (usado solo si el certificado del remitente no existe)
Key-Info: RSA,(ilegible)	Información de clave para un remitente secundario (Si fuera necesario)
Issuer-Certificate:(ilegible)	Secuencia de certificados del remitente.
MIC-Info: RSA-MDX,RSA, (ilegible)	Algoritmo de extracto de mensaje, algoritmo de cifrado de extracto del mensaje, sistema de Hash codificado y cifrado
Recipient-ID-	ID de cada destinatario y su información

Asymmetric: (ilegible),(numero) *Key-Info:* RSA,(ilegible)	de clave.
Línea en blanco	Línea en blanco
(Ilegible)	Mensaje codificado y cifrado.

Tabla 2.1 Características de un mensaje cifrado con Clave Publica [5]

Campo	Descripción
Proc-Type: 4,MIC-ONLY O MIC-CLEAR	Tipo de mensaje PEM (tipo, versión)
Content-Domain: RFC822	Tipo de mensaje enviado
Originator-Certificate: (ilegible)	Certificado codificado del remitente (opcional)
Originator-ID-Asymmetric: (ilegible),(numero)	Identificador del remitente ID (usado solo si el certificado del remitente no existe)
Issuer-Certificate:(ilegible)	Secuencia de certificados del remitente.
MIC-Info: RSA-MDX,RSA, (ilegible)	Algoritmo de extracto de mensaje, algoritmo de cifrado de extracto del mensaje, sistema de Hash codificado y cifrado
Línea en blanco	Línea en blanco
(Texto)	Mensaje (codificado si es del tipo MIC-ONLY)

Tabla 2.2 Características de un mensaje codificado (MIC-ONLY) o codificado con texto plano adicionado (MIC-CLEAR) con Clave Pública. [5]

Campo	Descripción
Proc-Type: 4,ENCRYPTED	Tipo de mensaje PEM (tipo, versión)
Content-Domain: RFC822	Tipo de mensaje enviado
DEK-Info: DES-CBC (16 dígitos hexadecimales)	Algoritmo de cifrado del mensaje
Originator-ID-Symmetric: (Identificador de entidad),(Autoridad Emisora),(Versión)/(Expiración)	Identificador del Remitente
Recipient-ID-Symmetric: (Identificador de entidad),(Autoridad Emisora),(Versión)/(Expiración) *Key-Info:* DES-ECB,RSA-MDx,(16 dígitos hexadecimales),(32 dígitos hexadecimales)	Identificador e información de clave para cada destinatario
Línea en blanco	Línea en blanco
(Ilegible)	Mensaje codificado y cifrado.

Tabla 2.3 Características de un mensaje cifrado con Clave Privada [5]

Campo	Descripción
Proc-Type: 4,MIC-ONLY O MIC-CLEAR	Tipo de mensaje PEM (tipo, versión)
Content-Domain: RFC822	Tipo de mensaje enviado
Originator-ID-Symmetric: (Identificador de entidad),(Autoridad Emisora),(Versión)/(Expiración)	Identificador del Remitente

Recipient-ID-Symmetric: (Identificador de entidad),(Autoridad Emisora),(Versión)/(Expiración) *Key-Info:* DES-ECB,RSA-MDx, (16 dígitos hexadecimales), (32 dígitos hexadecimales)	Identificador e información de clave para cada destinatario
Línea en blanco	Línea en blanco
(Texto)	Mensaje (codificado si es del tipo MIC-ONLY)

Tabla 2.4 Características de un mensaje codificado (MIC-ONLY) o codificado con texto plano adicionado (MIC-CLEAR) con Clave Privada. [5]

Aunque PEM fue descontinuado a mediados de los años 90, sentó las bases de la forma en la que se debían proteger los correos electrónicos, formatos que se usan hasta la actualidad, como en S/MIME.

2.2.2 DKIM

El sistema de Correo Identificado por Claves de Dominio (DomainKeys Identified Mail, o DKIM, por sus siglas en inglés) es un sistema diseñado para mantener seguro el correo electrónico mediante el uso de claves de dominio de manera que para el usuario de correo la seguridad sea completamente transparente. Depende de un dominio de autenticación que asigna una clave pública y una clave privada a cada dominio administrativo donde se generan correos electrónicos, que podrían ser proveedores de Internet, corporaciones o sistemas masivos de correo, como Gmail. Es utilizado habitualmente como filtro de Spam o de suplantación de identidades de dominio.

Definido por el RFC4686, se diferencia de otros sistemas de seguridad en que el usuario no maneja sus claves sino que el propio sistema lo hace, ya que cuando el usuario envía un nuevo mensaje este se firma con

la clave privada del dominio administrativo remitente, y se verifica mediante la clave pública del mismo, la cual es obtenida por el dominio administrativo del destinatario mediante interacciones DNS. Esta firma no solo incluye el contenido del mensaje, sino el contenido de ciertas cabeceras del RFC5322.

Este tipo de seguridad pretende eliminar el riesgo inherente de la característica ‘cualquiera-a-cualquiera’ que tiene el correo electrónico, lo que ha generado problemas de spam y suplantaciones varias de identidad. Al ser el dominio administrativo el que firma los mensajes, se asegura que la identidad del remitente sea verificada y no pueda ser suplantado externamente, ya que el remitente no conoce ni maneja la clave privada.

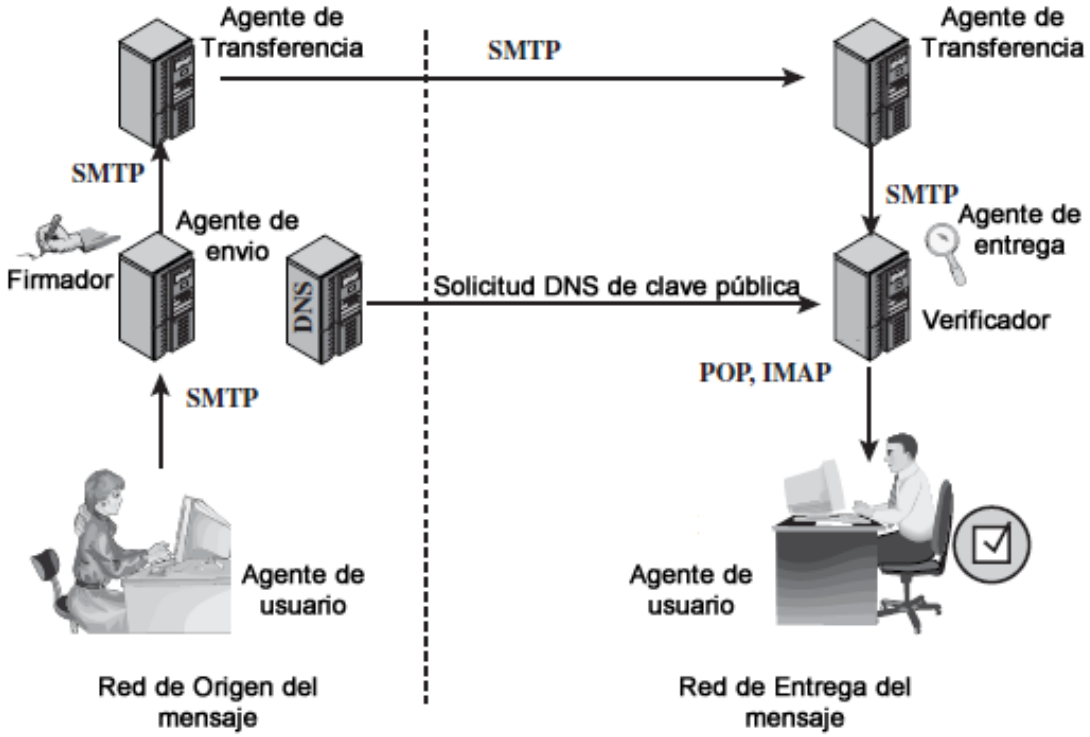


Figura 2.3 Ejemplo de envío y recepción de mensaje a través de DKIM [1]

Este tipo de cifrado para el usuario es completamente transparente, ya que la firma y la verificación se realizan en los agentes de envío y entrega de mensajes, es decir directamente en el servidor, lo que hace que el usuario no tenga ni siquiera que instalar ningún programa adicional a su agente de usuario habitual. Adicionalmente el Módulo Administrativo de Dominio (Administrative Management Domain, o ADMD por sus siglas en

inglés) determina la validez de la firma asociada al mensaje o determina el tipo de firma que debería ser adecuado para un mensaje en cuestión a fin de determinar la ‘reputación’ de un remitente. Si la firma ha sido considerada válida, esta información se añade al sistema de filtrado de mensajes, lo que aumenta la confianza en determinado remitente. Si la firma falla, o no se ha encontrado ninguna firma para el dominio en cuestión se busca información sobre el firmante para asignarla al sistema de filtrado de mensajes, pudiendo ser declarada como fraudulenta y bloqueada.

DKIM añade a la cabecera de correo electrónico el campo *DKIM-signature* antes de ser enviado, lo que le permite al Agente de recepción final verificar la firma del mensaje. Para que la cabecera sea agregada, el mensaje pasa por un proceso conocido como canonización, agregando mínimos cambios dentro de la cabecera y el mensaje, que consisten básicamente en el manejo de los espacios en blanco y los saltos de líneas, para estandarizar los mensajes y las cabeceras según como las definió DKIM.

```
From: Nemo <nemo@192.168.0.35>
To: Karen Nobody <karen@192.168.0.35>
Subject: Mensaje plano DKIM
X-Enigmail-Draft-Status: N1110
Message-ID: <55B7B5AF.5020741@192.168.0.35>
Date: Tue, 04 Ago 2015 19:25:49 -0300
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:38.0) Gecko/20100101
 Thunderbird/38.1.0
DKIM-Signature: v=1; a=rsa; c=relaxed/relaxed; t=y;
        d=undomiel.com; s=20120113;
        h=mime-version:from:to:subject:message-id:date:content-type;
        bh=zVFRCEuC9plKORHTj3ZigqRIg2MaRprOMmorEC2AHfs=;
        b=MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDMAiN9eMP+bniSsNHZJAUdR6Z7oT
         pn4B+hRfb8+/GvccRQpQTpVCa8uDp94gikPA28PpQhsftIskm4Ffr3l5bKr/a09doMgT
         4yOKY9AL5gJZRcbV0xYXiflPYoszyd8yRmDX3tw6xPRTe3bmq5RNU2VrRbrS7+Ts2GE9
         k46oOwmELJeb8Tdnv3txYohPKrZv1UZbTolj8jZZXx+Rn6b6xkCn+mZ74CWemv2KET6h
         +tJuN/WQIDAQABphdg==
MIME-Version: 1.0
Content-Type: text/plain; charset=utf-8
Content-Transfer-Encoding: 8bit

En un agujero en el suelo, vivía un hobbit. No un agujero húmedo, sucio,
repugnante, con restos de gusanos y olor a fango, ni tampoco un agujero
seco, desnudo y arenoso, sin nada en que sentarse o que comer: era un
agujero-hobbit, y eso significa comodidad.
Tenía una puerta redonda, perfecta como un ojo de buey, pintada de
verde, con una manilla de bronce dorada y brillante, justo en el medio.
John Ronald Reuel Tolkien.
.
```

Figura 2.4 Ejemplo de mensaje enviado por un servidor de correo con una firma DKIM

Esta cabecera incluye varios parámetros, como los que se aprecian en la figura 2.4, y son:

- v; versión de DKIM

- a; algoritmo para generar la firma, como sha-1 o sha-256
- c; método de canonización del mensaje
- d; nombre de dominio para identificar a la persona u organización responsable de la firma, también conocido como Identificador de Dominio Firmante (SDID o Signing Domain Identifier, por sus siglas en ingles)
- s; verificador de uso de la firma adecuada durante el momento de su verificación.
- h; identifica los campos de la cabecera que serán firmados
- bh; hash del cuerpo del mensaje canonizado.
- b; los datos de firma expresados en base64.

Para la implementación de DKIM, el sistema de dominio debe contar con su propio servidor DNS que administre la clave pública del dominio y propague la misma dentro de su red para que el servidor de correo pueda hacer uso de la misma. Para que los mensajes puedan ser firmados, la clave privada estará en el servidor de correo, mientras que la clave pública puede o no estar en el mismo, dependiendo de la configuración de cada sistema de dominio.

2.2.3 S/MIME

MIME Seguro, o S/MIME, es una adecuación de seguridad al formato MIME de correos electrónicos, surgido principalmente del mundo comercial y empresarial para el manejo de información confidencial y está referenciado en los RFC 3370, 3850, 3851 y 3852. Está diseñado para ofrecer al usuario la capacidad de firmar y/o cifrar los mensajes.

Al igual que PEM, S/MIME puede enviar información en 3 modos distintos;

- Información cifrada (figura 2.5); consistente en información cifrada de cualquier tipo y las llaves de cifrado del contenido para los destinatarios.

```
To: Karen Nobody <karen@192.168.0.35>
From: Nemo <nemo@192.168.0.35>
Subject: Mensaje Cifrado S/MIME
Message-ID: <55BAE083.1030508@192.168.0.35>
Date: Thu, 30 Jul 2015 23:42:11 -0300
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:38.0) Gecko/20100101
 Thunderbird/38.1.0
MIME-Version: 1.0
Content-Type: application/pkcs7-mime; name="smime.p7m"; smime-type=enveloped-data
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename="smime.p7m"
Content-Description: Mensaje cifrado S/MIME

MIAGCSqGSIb3DQEHA6CAMIACAQAxggLCMIICvgIBADCBpTCBlzELMAkGA1UEBhMCQVIxFTAT
BgNVBAgMDEJ1ZW5vcyBBaXJlczEVMBMGA1UEBwwMQnVlbm9zIEFpcmVzMREwDwYDVQQKDAhV
bmRvbWllbDENMAsGA1UECwwETWFpbDEVMBMGA1UEAwwMS2FyZW4gTm9ib2R5MSEwHwYJKoZI
hvcNAQkBFhJrYXJlbkAxOTIuMTY4LjAuMzUCCQCd6HvxI2FLqTANBgkqhkiG9w0BAQEFAASC
AgCGPjkV+1ybkqxsMXRUA6JWjjDRJs82Be+CVrrqjRziVoCxfMpLswfOpGzYB4aVRSRE36ua
9ToH/XMx68hPQ+ddlY2AgClfM7hSfJULvdvKQ4wV4Y5hf8mvXcYKR9MDRiC5cZkhDAJJt3dX
xhOULzrdWDBRfcIVf9bxf60x9O3NXoYy/5Ci4wWxPBsfWt+rfpCuVKeSuk6xMLRQ8tCO4Z1o
MMdvra4bt+8P2PExh3EkVrRGgaJnd5USLjdBe5Min3gAxAOMb1lgdRJOEgfEYhSTiwE5tMvN
G/RMav8hNabTh7UIF7hvA6xJIROl/ncMjdkU1cOD4d9Eqht2uUgY733Wsz8d+N4hnZbRp6P8
azXRpixIB2QJgw8AEBc6eyubi0SI9p6/yXsiIyHuqO8YwGAV8Xyjl5FmVhXjhguOx8a39M6k
H3dbOXen4KvHBFtPAcS3oOiwAwv2UOCos3GB+9f/1SplPugqvHAXzkvMhSfiRWNtVlhIXDUe
ix0yEum9oUClRpWDydBXWikLr+wjm7ZDlCUERmWugO4RT2bJ3W579sdBItrKNtOQHElM5VmC
hOSwgOMKJf1h8iNVzr69W5ngy5XUlAmRxLzPRlhGmHO9X5ltJ/UTgC7naF67AADVyF6DeOEP
nTvOXTXH5iivE17794Z+ean8VqnM48ESxNM1zDCABgkqhkiG9w0BBwEwFAYIKoZIhvcNAwcE
CBNdsLPpYliKoIAEggIYoSjmniw933CJnlB7augDAvXerugLhcLv5vpwef62fEgzZc5B+kLm
baIYaqBDQJpqtm3Ki37DeyWWYJHZUHEja/7pjnCTlGB8AxqB/mtL+rLMSG+OvBi+SsZbqBmH
Vj8SYi796PiYkKAxYr6q8xPkJsgtWs/tDXndmlO8sCSQOq3ZMMsYHoGALOMJ/cJ7y3SozwrI
JDLuFJbMkmsbumqefsVUWFUDSCNPjnu7U5nhNtd4t/mOQtuK1sOywXgsJWdMHa5Srr/TOfnQ
Nqqrx+6J1s9GjcFg6KRVFzyY2IkBO7BExYDUdJ6ouL8GlukeA6Bhfk6XXHqiaYfXK5z+e+8m
lhLQurhmyiCjLOzWRTaCWNbtYPrNRf0y767upa8LVnmkLQhLI2r40Ohf0RyvZ1lClIOw4YnL
pvT8uewqm1TzHOeETxP5+QQYvfAPVfKGMQsyUMctlxMPZcXRrK6U+CqaEa+QJQDO2bc6/Ntc
WvLoJ+cHyh9t+ArqOL4Ak5yqOfBge9znkOPNMlXIwYwkbMMIcWX5Ew+yzjVIqhMX7aEZ7NYa
iAtxjuDAdI1TAUt7h5OvlmH1nnY8DndXavHdySP8zzDGFCUmzrCkVtEjVEGCAYPWCobMKNRg
kK5rKPWqwloyaWn5vupcLf12FylhxUUjgHKsDGwZXrolmYwZX8m9VmHsbcNhUfyV1gKz8qwF
JRhvFKcSe+NkFe4ECLstDXtIg2VoAAAAAAAAAAAAAA==
.
```

Figura 2.5 Ejemplo de mensaje S/MIME cifrado

- Información firmada (figura 2.6); formada por el extracto del mensaje firmado por la clave privada del remitente adicionado al contenido en plano del mensaje. Puede ser leida por todos los destinatarios, pero solo comprobada por los destinatarios con capacidad S/MIME

```
To: Karen Nobody <karen@192.168.0.35>
From: Nemo <nemo@192.168.0.35>
Subject: Mensaje Firmado S/MIME
Message-ID: <55BADC82.7090306@192.168.0.35>
Date: Thu, 30 Jul 2015 23:25:06 -0300
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:38.0) Gecko/20100101
 Thunderbird/38.1.0
MIME-Version: 1.0
Content-Type: multipart/signed; protocol="application/pkcs7-signature"; micalg=sha-512;
boundary="------------ms050803090705000706040300"

This is a cryptographically signed message in MIME format.

--------------ms050803090705000706040300
Content-Type: text/plain; charset=utf-8
Content-Transfer-Encoding: quoted-printable

En un agujero en el suelo, viv=C3=ADa un hobbit. No un agujero h=C3=BAmed=
o, sucio,
repugnante, con restos de gusanos y olor a fango, ni tampoco un agujero
seco, desnudo y arenoso, sin nada en que sentarse o que comer: era un
agujero-hobbit, y eso significa comodidad.
Ten=C3=ADa una puerta redonda, perfecta como un ojo de buey, pintada de
verde, con una manilla de bronce dorada y brillante, justo en el medio.
John Ronald Reuel Tolkien.


--------------ms050803090705000706040300
Content-Type: application/pkcs7-signature; name="smime.p7s"
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename="smime.p7s"
Content-Description: Firma criptográfica S/MIME

MIAGCSqGSIb3DQEHAqCAMIACAQExDzANBglghkgBZQMEAgMFADCABgkqhkiG9w0BBwEAAKCC
C1wwggWoMIIDkAIJAPIL5DGxBCT9MA0GCSqGSIb3DQEBCwUAMIGVMQswCQYDVQQGEwJBUjEV
MBMGA1UECAwMQnVlbm9zIEFpcmVzMRUwEwYDVQQHDAxCdWVub3MgQWlyZXMxETAPBgNVBAoM
CFVuZG9taWVsMQ0wCwYDVQQLDARNYWlsMRQwEgYDVQQDDAtOZW1vIE5vYm9keTEgMB4GCSqG
SIb3DQEJARYRbmVtb0AxOTIuMTY4LjAuMzUwHhcNMTUwNzI3MTcxOTQ4WhcNMjUwNzI0MTcx
OTQ4WjCBlTELMAkGA1UEBhMCQVIxFTATBgNVBAgMDEJ1ZW5vcyBBaXJlczEVMBMGA1UEBwwM
QnVlbm9zIEFpcmVzMREwDwYDVQQKDAhvbmRvbWllbDENMAsGA1UECwwETWFpbDEUMBIGA1UE
AwwLTmVtbyBOb2JvZHkxIDAeBgkqhkiG9w0BCQEWEW5lbW9AMTkyLjE2OC4wLjM1MIICIjAN
BgkqhkiG9w0BAQEFAAOCAg8AMIICCgKCAgEAmMGmcX1glgWXW9BjIDj/XIxStBQbDq9Qpu3I
TOn6XsPYrKiCvKKvkOrayU6eCbfwAhCfbP5IkLupIBDa9f1OiJMdDRMuf6JHFJ7Xkl0vqNBD
cNlCDrzGT0AIa93W6c/2hHy+aSgtV2eEOI2smtqEG/aM82G0AzPV8cBhPxo2b6QEHlJNIOqC
tphcmwTqB29Z2Hpa0OtsDgOu1uHtKBGLgBadlRwiu5++Tmfuz2OnBs9zboUJNV8VtYsrk3Pz
fnu3KpJqOxb1/8eDCstig3vTOTeFF5WVrVeRM9v1ajKhg153DUTgvCCl+x/bgY6KHzh4H7LB
y+cyOL/OxArSbrGXaKarc1GiMC7eXNzM/1oYooR4Jm3ZwjndgCoI2Dbwlp+euFkF592atEQJ
557EC/EhAeXu81z/OeFuimLKgOoTusp9vxYDUieYsVUYKX//Kxec6c8EMaZhZV8OsiXS3B/i
dQ3G/8OWwvh5m+kwwTUhtwlbhUng9qK5EKbFU2D6gfSyPyHSURl12QAkySCqOjh22Gy6WEZ4
obdGaJwJ3XWj2PLV9TNgcstOpn1fY85Oqqin1yN1jov7vW1JKuxodbkaMqzyUX6v9tIO7dTa
fOmppmLco2CUZNhANJtwuWq3FC2J1fWcjq3m+eGI2aMQXfWgMuzPv5ywKsrJ2vwOqw0bV3sC
(...)
BgNVBAMMC05lbW8gTm9ib2R5MSAwHgYJKoZIhvcNAQkBFhFuZW1vQDE5Mi4xNjguMC4zNQIJ
APIL5DGxBCT9MA0GCWCGSAFlAwQCAwUAoIICazAYBgkqhkiG9w0BCQMxCwYJKoZIhvcNAQcB
MBwGCSqGSIb3DQEJBTEPFw0xNTA3MzEwMjI1MDdaME8GCSqGSIb3DQEJBDFCBEDYYIpuOip/
zw0fpTwKE2u3Uwu+zdU3TjsAJjPfS55NC+5ek0InnRwabkpOgcOpw8F9NKxydOT5w7JTyddV
j5kvMGwGCSqGSIb3DQEJDzFfMF0wCwYJYIZIAWUDBAEqMASGCWCGSAFlAwQBAjAKBggqhkiG
9w0DBzAOBggqhkiG9w0DAgICAIAwDQYIKoZIhvcNAwICAUAwBwYFKw4DAgcwDQYIKoZIhvcN
AwICASgwgbYGCSsGAQQBgjcQBDGBqDCBpTCBlzELMAkGA1UEBhMCQVIxFTATBgNVBAgMDEJ1
ZW5vcyBBaXJlczEVMBMGA1UEBwwMQnVlbm9zIEFpcmVzMREwDwYDVQQKDAhvbmRvbWllbDEN
MAsGA1UECwwETWFpbDEVMBMGA1UEAwwMS2FyZW4gTm9ib2R5MSEwHwYJKoZIhvcNAQkBFhJr
YXJlbkAxOTIuMTY4LjAuMzUCCQCd6HvxI2FLqTCBuAYLKoZIhvcNAQkQAgsxgaiggaUwgZcx
CzAJBgNVBAYTAkFSMRUwEwYDVQQIDAxCdWVub3MgQWlyZXMxFTATBgNVBAcMDEJ1ZW5vcyBB
aXJlczERMA8GA1UECgwIVW5kb21pZWwxDTALBgNVBAsMBE1haWwxFTATBgNVBAMMDEthcmVu
IE5vYm9keTEhMB8GC5qGSIb3DQEJARYSa2FyZW5AMTkyLjE2OC4wLjM1AgkAneh78SNhS6kw
DQYJKoZIhvcNAQEBBQAEggIAVQFQceEd9b63JkFIAVVuoT4XisYb62/y0ZUfgDS8EfQXnt5
BSya35CUtp2S8U/5hllvIuuhs9qK8d/L8SWX3XzJIeZJzozGeXT7O6KPgtAWwbOcUQ3HOZ4M
UdHyoqJKzGEPv1GfqUdHEAPqUWwmNOfd0bOxFxRiUMTu9s0lzyLd2TG8YzTaRN10TSNz1i2c
lohNXsOHPlvQJC5xAsBXMKoRFN5g5gQcpUwOWFTsr6rMnOeDVOsrLV5eseO2PHiwLA1/ff8i
XXXqJSkjTQY5qeE0bnOP5mafUawfGzoKSglOYOF6rZeBrc2YrQB6O8mVQUIZzbxIs5knecmm
zIFeIa5LbDNTlRLUcPNaPOdquG8egRYo+O1plxSwZmdFFjr8P4t6B+pBy1J621xtFTNIUdTV
R1HFst4eRbc6mLFaWKuf9WIgZ1hF5VENcSjSh/4ZQu27xz853dw//KYVwInX9NhGlkH2B9zO
5o9X4NDwO/yH8+zI7sNXqfGs+ujQv+dsGa9nPumES/ZP7beWngToKHA+1jk8UUSgEE5WQwed
Zde8AqvRv4Ao9mX5XOwiPH6YPCgHESfSUzCkz4OTQDzRBt2LwGoVYG1LMuqR67liL4TI2v59
/vYULATVf2cEWt3ZDxVaxmfjomlBjNzec9SCAz/NOo3CLciu5iEdZecOzicAAAAAAAA=
--------------ms050803090705000706040300--
.
```

Figura 2.6 Ejemplo de mensaje S/MIME firmado

- Información firmada y cifrada (figura 2.7); formada por información de los dos tipos anteriores, que se envía anidada en un solo mensaje.

Figura 2.7 Ejemplo de mensaje S/MIME cifrado y firmado

Adicionalmente a RSA, S/MIME hace uso de una variante de Diffie-Hellman, conocida como ElGamal para el cifrado de las claves de sesión y el uso del algoritmo SHA-1 para la firma de la información. S/MIME está diseñado, además, para decidir que algoritmo de seguridad utiliza para cada caso, determinando sus propias capacidades y las capacidades del destinatario del mensaje y decidiendo con esta información si es aceptable que se realice un envío de información asegurada débilmente. Antes de establecer la conexión, S/MIME sigue los siguientes pasos en secuencia para decidir cómo enviar la información:

- Si el agente de envío posee enlistadas sus capacidades de descifrado para un determinado destinatario, escogerá siempre la de más alta seguridad, primera en la lista.
- Si el remitente no posee esta información, pero ha recibido anteriormente mensajes del destinatario, entonces utilizara el mismo tipo de cifrado utilizado anteriormente.

- Si el remitente no tiene ningún conocimiento sobre las capacidades de cifrado del destinatario, utilizara 3DES para el envío del mensaje.
- Si, adicionalmente, el remitente considera que el destinatario no soporta 3DES, el sistema deberá utilizar RC2[1] de 40 bits, aun cuando RC2 sea considerado inseguro desde 1997[2].

Para el caso de varios destinatarios, el agente de envío realiza esta evaluación para todos, y establece la seguridad del mensaje en aquella que sea soportada por el destinatario con menor capacidad de cifrado.

De la misma manera que MIME definió tipos específicos de contenidos para el envío y recepción de información no consistente en texto plano, S/MIME definió dos nuevos tipos para seguridad exclusivamente.

Tipo	Subtipo	Parámetros S/MIME	Descripción
Multiple	Firmado		Mensaje firmado-plano en dos partes, el mensaje y la firma.
Aplicación	pkcs7-mime	signedData	Mensaje S/MIME firmado.
	pkcs7-mime	envelopedData	Mensaje S/MIME cifrado.
	pkcs7-mime	degenerate signedData	Mensaje con certificados de Clave Pública únicamente
	pkcs7-mime	CompressedData	Mensaje S/MIME comprimido
	pkcs7-signature	signedData	Tipo de contenido con una parte firmada de un mensaje del tipo Múltiple.

Tabla 2.5 Tipos de Contenido S/MIME [1]

[1] Cifrado de bloque de clave simétrica diseñado por Ron Rivest en 1987.

[2] John Kelsey, Bruce Schneier, David Wagner: **Related-key cryptanalysis of 3-WAY, Biham-DES, CAST, DES-X, NewDES, RC2, and TEA**. ICICS 1997: 233–246

- pkcs7-mime (ejemplificado en las figuras 2.5 y 2.7); usada para cada una de las cuatro categorías definidas por S/MIME de la tabla 2.5. Define al contenido etiquetado como un objeto y lo transforma a base64 para que pueda ser enviado como un mensaje MIME ordinario. Para obtener la información de vuelta, el destinatario vuelve a convertir el mensaje en base64 a un objeto cifrado, el cual puede ser descifrado con su clave privada.
- pkcs7-signature (ejemplificado en la figura 2.6); utilizada para enviar mensajes firmados, en los cuales se incluye embebidos el extracto del mensaje cifrado para comprobar la autenticidad, todo esto una vez más agrupado en bloques de información y codificado en base64. Adicionalmente este tipo se utiliza para enviar mensajes que deben llevar su contenido sin cifrar adicionalmente, para aquellos destinatarios que no tengan capacidad S/MIME. Estos reciben un mensaje ordinario de correo electrónico, que viene con un archivo adjunto llamado smime.p7s, en el cual están las seguridades del mensaje.

El manejo de PKI de S/MIME se realiza casi en su totalidad en cooperación de VeriSign a través de una variante de X.509 que añade adicionalmente una red de confianza entre usuarios para la compartición y comunicación de certificados, muy parecido al sistema desarrollado previamente por PEM. A través de estas características, S/MIME puede generar, registrar e invalidar certificados de sus usuarios de confianza, además de utilizar los sistemas de VeriSign para asegurar la identidad de los comunicantes.

Figura 2.8 Ejemplo de un certificado S/MIME visto desde el cliente Thunderbird

2.3 OTROS MODELOS DE CORREO SEGURO

Adicionalmente a los protocolos seguros de correo electrónico, existen implementaciones, sobretodo en internet que hace uso de las propiedades del RFC 5322 o de otros sistemas de direccionamiento de internet para hacer que los usuarios que envían mensajes puedan mantenerse seguros o anónimos, según sean sus necesidades.

2.3.1 CORREO ELECTRONICO ANÓNIMO [9][10]

Está basado en la idea que el destinatario no tiene ni puede tener idea de quién es el remitente, ya que hace uso de técnicas de distracción y de ocultamiento de su identidad y su dirección IP. Este tipo de servicios

pseudoanonimos, conocidos como *remailers*, permiten la comunicación bidireccional entre dos usuarios de internet sin que estos, técnicamente, sepan quién es su interlocutor. Para habilitar la comunicación, el servidor anónimo asigna un alias a cada comunicante, basado en una tabla de identidades, la cual está relacionada con el alias que estos servicios ofrecen, para poder mantener una comunicación fluida. Para responder un mensaje, el destinatario debe conocer el alias del remitente, con lo cual las identidades están protegidas.

Según su configuración y prestaciones, se han dividido en algunos tipos, pero de manera general son:

- Remailer tipo 0; que remueve las cabeceras de identificación de RFC 5322 y envía el mensaje al destinatario. Aunque los servidores puente no conocen la dirección IP origen del remitente, este sistema es vulnerable a escuchas durante el trayecto del mensaje, donde el intruso podría determinar origen, destino y contenido.

- Remailer tipo 1; donde los correos electrónicos están conformados por un grupo anidado de mensajes cifrados, que se envían a través de caminos combinados. Cada uno de estos caminos descifra un mensaje y obtiene un conjunto de características propias del correo que se está enviando, además del mensaje cifrado anidado. Para habilitar la respuesta el remitente recibe un alias dentro de la red de caminos combinados, para que el destinatario pueda responder de la misma manera.

- Remailer tipo 2; conocido como MixMaster, encripta el mensaje con diferentes claves simétricas y se lo envía por una red similar a la del remailer de tipo 1. La diferencia radica en que el mensaje antepone una cabecera adicional al mensaje, que contiene la dirección final y el camino a recorrer, un identificador de paquete y de mensaje y las claves necesarias para descifrar los paquetes

adicionales. Esta cabecera se cifra con la clave simétrica del camino a recorrer, y luego al conjunto se lo cifra múltiples veces con las claves simétricas de los caminos previamente recorridos.

El destinatario recibe el primer paquete y lo descifra, para obtener las claves de los paquetes restantes y poder re ensamblar el mensaje enviado y poder contestarlo o reenviarlo, si fuera el caso.

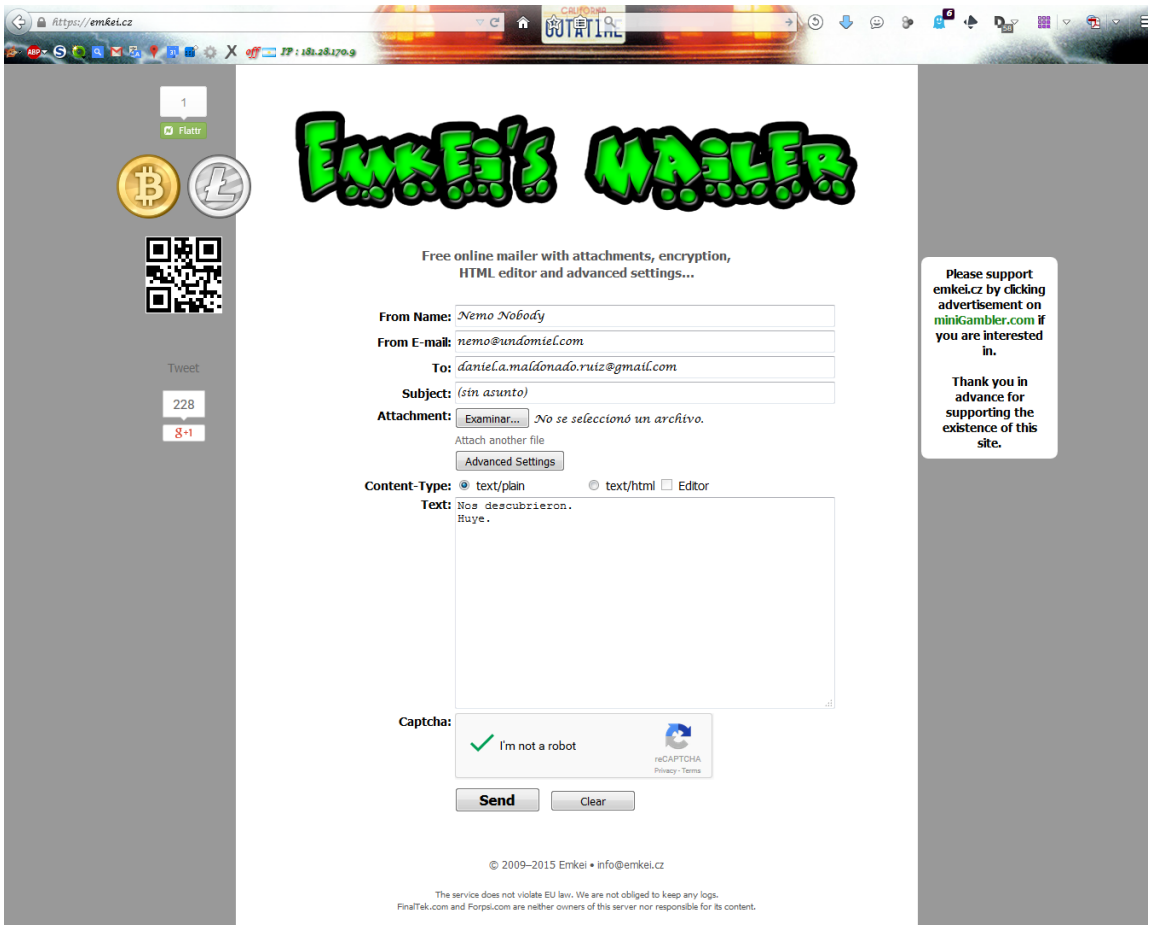


Figura 2.9 Ejemplo de un remailer tipo 0 gratuito con sistema ‘on-the-fly’

```
Delivered-To: daniel.a.maldonado.ruiz@gmail.com
Received: by 10.31.69.13 with SMTP id s13csp561894vka;
        Wed, 5 Aug 2015 10:08:14 -0700 (PDT)
X-Received: by 10.181.12.111 with SMTP id ep15mr465105wid.15.1438794494481;
        Wed, 05 Aug 2015 10:08:14 -0700 (PDT)
Return-Path: <nemo@undomiel.com>
Received: from emkei.cz ([2a01:5e0:36:5001::21])
        by mx.google.com with ESMTP id ba10si11309298wib.29.2015.08.05.10.08.13
        for <daniel.a.maldonado.ruiz@gmail.com>;
        Wed, 05 Aug 2015 10:08:13 -0700 (PDT)
Received-SPF: fail (google.com: domain of nemo@undomiel.com does not designate 2a01:5e0:36:5001::21 as permitted sender)
client-ip=2a01:5e0:36:5001::21;
Authentication-Results: mx.google.com;
       spf=fail (google.com: domain of nemo@undomiel.com does not designate 2a01:5e0:36:5001::21 as permitted sender)
smtp.mail=nemo@undomiel.com
Received: by emkei.cz (Postfix, from userid 33)
        id 3E239D572C; Wed,  5 Aug 2015 19:08:13 +0200 (CEST)
To: daniel.a.maldonado.ruiz@gmail.com
Subject: (sin asunto)
From: "Nemo Nobody" <nemo@undomiel.com>
X-Priority: 3 (Normal)
Importance: Normal
Errors-To: nemo@undomiel.com
Reply-To: nemo@undomiel.com
Content-Type: text/plain; charset=utf-8
Message-Id: <20150805170813.3E239D572C@emkei.cz>
Date: Wed,  5 Aug 2015 19:08:13 +0200 (CEST)

Nos descubrieron.
Huye.
```

Figura 2.10 Ejemplo de cabecera de un mensaje enviado desde un remailer tipo 0

La figura 2.10 muestra cómo, a pesar de que en el remailer permite asignar al remitente cualquier nombre de dominio que este escoja para proteger su anonimato, el correo realmente se envía con un identificador propio del dominio del remailer (en este caso emkei.cz), que sirve únicamente para referencia interna del envío. Sin embargo, si el remitente no ha especificado una dirección de correo válida para recibir una respuesta, el sistema de correo del destinatario asumirá que la dirección alias es la dirección real del remitente y tratará de enviar los mensajes a esta dirección (en el caso del ejemplo, a nemo@undomiel.com); si el dominio no existe, todas las respuestas se perderán. Existen servicios que ofrecen capacidades avanzadas de Tipo 0 o funciones de tipo 1 y 2, la mayoría con una suscripción y un pago mensual.

Estos sistemas suelen ser vulnerables al spam, ya que aunque no se conozca su identidad, el alias de un usuario puede ser conocido por *spamers,* además que los sistemas de remailers no implementan directamente controles frente a correo electrónico no solicitado, dejándole esa labor al usuario. Adicionalmente, este tipo de comunicaciones pueden ser espiadas por terceros, que pueden leer los mensajes en transmisión y determinar las rutas de comunicación u obtener las tablas de correlación de identidades y alias para conocer las identidades reales de los comunicantes.

2.3.2 CORREO ELECTRONICO SOBRE REDES ANONIMAS[9][11][12]

Las redes anónimas permiten que no solo los mensajes vayan seguros, sino que las direcciones de los comunicantes vayan protegidas. Para este cometido, la más conocida de todas es la Red Onion, que es un conjunto de túneles virtuales superpuestos en capas, como una cebolla (onion en inglés), en un concepto similar a los caminos combinados de los remailers del tipo 1 y 2. Para hacer uso de este sistema, el usuario debe contar con un enrutador Onion y un proxy Onion, los cuales determinan y traducen las direcciones de la red para que el usuario pueda comunicarse. Dado el sistema en capas de TOR (The Onion Router, o el enrutador Onion, en inglés) las direcciones para comunicarse van cifradas con las claves de los enrutadores, lo que hace que sean casi inteligibles para las personas, por esta razón son conocidas como direcciones .onion.

Cuando es necesario establecer una comunicación, se configura una red onion para este cometido en el momento de iniciar la comunicación. El enrutador onion descifra su parte de la red, obtiene la información necesaria sobre enrutamiento y claves necesarias y vuelve a cifrar la información embebida para enviarla al siguiente enrutador. Mientras la información viaja a través de la red TOR viaja cifrada, y de los enrutadores finales a los usuarios la información vuelve a estar descifrada y legible.

Cada uno de los sistemas de correo dentro de TOR deben tener necesariamente una conexión con la red IP, para que los mensajes generados desde TOR puedan ser entregados a servidores mainstream como Gmail. En la figura 2.11 se muestra la cabecera de un mensaje enviado desde un servicio TOR (bitmessage.ch) en el que se puede diferenciar el servicio IP (mail.bitmessage.ch) y el servicio onion (bitmailendavkbec.onion) que utiliza para enviar mensajes por fuera de la red

TOR, así como el uso de un sistema DKIM para certificar los mensajes enviados.

```
Delivered-To: daniel.a.maldonado.ruiz@gmail.com
Received: by 10.31.69.13 with SMTP id s13csp622679vka;
        Wed, 5 Aug 2015 12:32:06 -0700 (PDT)
X-Received: by 10.194.23.194 with SMTP id o2mr23759095wjf.63.1438803126569;
        Wed, 05 Aug 2015 12:32:06 -0700 (PDT)
Return-Path: <BM-2cWSyjhvpuP1WxDS1nxVXuZ458tT165zx6@bitmessage.ch>
Received: from mail.bitmessage.ch (mail.bitmessage.ch. [146.228.112.252])
        by mx.google.com with ESMTPS id p9si12053547wiv.76.2015.08.05.12.32.05
        for <daniel.a.maldonado.ruiz@gmail.com>
        (version=TLSv1.2 cipher=ECDHE-RSA-AES128-GCM-SHA256 bits=128/128);
        Wed, 05 Aug 2015 12:32:06 -0700 (PDT)
Received-SPF: pass (google.com: domain of BM-2cWSyjhvpuP1WxDS1nxVXuZ458tT165zx6@bitmessage.ch
designates 146.228.112.252 as permitted sender) client-ip=146.228.112.252;
Authentication-Results: mx.google.com;
       spf=pass (google.com: domain of BM-2cWSyjhvpuP1WxDS1nxVXuZ458tT165zx6@bitmessage.ch
designates 146.228.112.252 as permitted sender) smtp.mail=BM-
2cWSyjhvpuP1WxDS1nxVXuZ458tT165zx6@bitmessage.ch;
      dkim=pass header.i=@bitmessage.ch
dkim-signature: v=1; a=rsa-sha256; d=bitmessage.ch; s=mail;
        c=relaxed/relaxed; q=dns/txt; h=From:Subject:Date:Message-ID:To:MIME-Version:Content-
Type:Content-Transfer-Encoding;
        bh=v3ftokytydyF5/ifmfr5UaRyXXunAlHVfxLaV6Uw2R8=;

b=W02ie8RV/3PDx1qJFCwJKD3kBQK/sC3SnPEfLTFFUOn20uYQHK9DO8B39YkThCZToiqrANmpfBYLuCaYkBrvsW0HpQZKYI
Yfkz3M/iWKichy2UQ2Or/y3o5CF3TAUbSgpNeCZLppMwg8eH1N7k5poruL73UE6pucEyuJiKY/HDs=
Received: from bitmailendavkbec.onion (BITMESSAGE [127.0.0.1])
        by mail.bitmessage.ch with ESMTPA
        ; Wed, 5 Aug 2015 21:31:29 +0200
MIME-Version: 1.0
Content-Type: text/plain; charset=US-ASCII;
 format=flowed
Content-Transfer-Encoding: 7bit
Date: Wed, 05 Aug 2015 12:31:27 -0700
From: Nemo Nobody <BM-2cWSyjhvpuP1WxDS1nxVXuZ458tT165zx6@bitmessage.ch>
To: daniel.a.maldonado.ruiz@gmail.com
Subject: (no subject)
Message-ID: <c8bffee3bbadac91c8d291baa2e3c5a0@bitmessage.ch>
X-Sender: BM-2cWSyjhvpuP1WxDS1nxVXuZ458tT165zx6@bitmessage.ch
User-Agent: Roundcube Webmail/1.0.2

Corre
```

Figura 2.11 Ejemplo de cabecera de un mensaje enviado desde la red TOR hacia una casilla de correo de Google

Este mismo servicio dentro de la red TOR utiliza únicamente direcciones .onion para establecer la conexión, la cual por configuración del sistema es dinámica. Este tipo de correos, al no salir hacia la red IP, no son susceptibles de ser espiados, ya que no pueden ser espiados mientras están siendo transportados o en los enrutadores de paso entre una red y otra, volviéndolos un sistema seguro de comunicación. En la figura 2.12 se muestra un ejemplo de cabecera de mensaje enviado entre dos servicios TOR, en donde se puede apreciar el uso único de direcciones .onion.

```
Return-Path: n.nobody1628@sigaint.org
Delivered-To: BM-2cWSyjhvpuP1wxDS1nxVXuZ458tT165zx6@bitmessage.ch
Received: from mx1.sigaint.org (BITMESSAGE [127.0.0.1])
        by mail.bitmessage.ch with ESMTPS (version=TLSv1.2 cipher=ECDHE-RSA-AES256-GCM-SHA384 bits=256);
        Wed, 5 Aug 2015 22:20:13 +0200
Received: from sigaintevyh2rzvw.onion (localhost [127.0.0.1]);
        by localhost (OpenSMTPD) with ESMTP id a61cbe0c;
        for <BM-2cWSyjhvpuP1wxDS1nxVXuZ458tT165zx6@bitmessage.ch>;
        Wed, 5 Aug 2015 20:20:16 +0000 (UTC)
Received: from 127.0.0.1 (HTTP authenticated user n.nobody1628)
        by localhost with HTTP;
        Wed, 5 Aug 2015 20:20:16 -0000
Message-ID: <d0f506a32af16fcbf5d3f4c534c5090b.webmail@localhost>
Date: Wed, 5 Aug 2015 20:20:16 -0000
Subject: (sin asunto)
From: n.nobody1628@sigaint.org
To: BM-2cWSyjhvpuP1wxDS1nxVXuZ458tT165zx6@bitmessage.ch
MIME-Version: 1.0
Content-Type: text/plain;charset=iso-8859-1
Content-Transfer-Encoding: 8bit
X-Priority: 3 (Normal)
Importance: Normal
```

Figura 2.12 Ejemplo de cabecera de un mensaje enviado dentro de la red TOR

Adicionalmente a todos los sistemas y protocolos tratados, existe uno que fue desarrollado por un hacker en 1991, para prevenir que su información fuera vulnerada por los gobiernos. A este sistema se le denominó PGP, y será explicado en el siguiente capítulo.

CAPÍTULO 3

PGP

En 1991, Phil Zimmermann, preocupado por la creciente intromisión del gobierno de los Estados Unidos en el correo electrónico de los ciudadanos, desarrolló un sistema que unificaba la criptografía de clave pública y las firmas digitales con protocolos de seguridad conocidos y probados, y hacía posible la transmisión segura de información a través de la internet insegura, y la llamó Privacidad Bastante Buena (Pretty Good Privacy, o PGP por sus siglas en inglés).

A pesar de haber sido difundido universalmente para asegurar correo electrónico, su diseño le permite proteger y asegurar la integridad de cualquier tipo de archivo, sea este para su transmisión o para su almacenamiento. Parte de su fortaleza se sustenta en que es un software de código abierto, lo cual hace que pueda ser modificado para determinadas funciones específicas, además de por defecto hacer uso de protocolos de cifrado que no han sido objeto de desarrollo por parte de alguna entidad gubernamental, sobre todo la de los Estados Unidos.

3.1 FUNCIONAMIENTO [1][3][4]

PGP utiliza conocidos algoritmos de seguridad, como SHA-1, IDEA, CAST o RSA de tal manera que con ellos ensambla un sistema de seguridad el cual puede ser invocado de acuerdo al uso que se le va a brindar. En el caso de correo electrónico, fue desarrollado de tal manera que los usuarios, según la versión que estén utilizando, puedan escoger que algoritmos de seguridad van a implementar.

En general, PGP ofrece servicios de seguridad, descritos en la tabla 3.1, que se integran el uno con el otro para el envío de un mensaje por medio electrónico. Al mensaje en cuestión M, PGP le aplica un hash SHA-1,

y con la clave privada D_A proporcionada por RSA encripta el hash obtenido. Con el hash cifrado y el mensaje original se construye un nuevo mensaje concatenado, M1, que es comprimido mediante un programa ZIP, que utiliza el algoritmo Ziv-Lempel[3].

Al mensaje comprimido, MZ1, se lo cifra mediante una clave IDEA generada a partir de una entrada que PGP le pide ingresar al remitente. Los datos ingresados, añadidos a los patrones de tecleo, forman la clave pseudo-aleatoria de 128 bits K_M con la que se cifrará el mensaje comprimido y que servirá como 'clave de sesión' única para la comunicación. RSA también cifrará la clave de sesión con la clave pública E_B del destinatario antes de enviarla. Estos cifrados se concatenan y se convierten a un texto base64, para que pueda ser enviado por correo electrónico. Para descifrar el mensaje que fue enviado, el destinatario debe realizar el proceso inverso al del remitente, y con la clave de sesión K_M, descifrar el mensaje para poder leerlo. La figura 3.1 ilustra como todos los algoritmos se concatenan al formar el mensaje PGP que finalmente se envía, mientras que la figura 3.2 ilustra mediante un modelo como se ensambla el mensaje PGP a partir de las concatenaciones de los algoritmos utilizados.

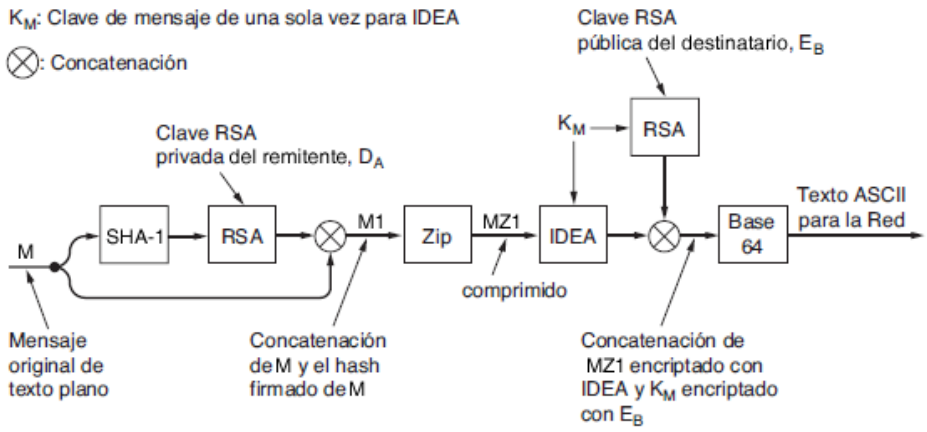


Figura 3.1 Diagrama de procesamiento del mensaje del lado del remitente. [3]

[3] Jacob Ziv & Abraham Lempel. "A Universal Algorithm for Sequential Data Compression". IEEE Transactions on Information Theory 23 (3): pags. 337–343. Mayo 1977. doi:10.1109/TIT.1977.1055714

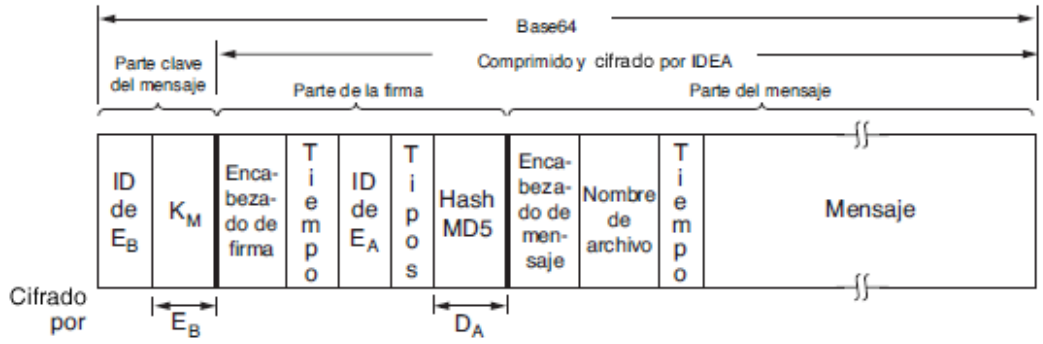


Figura 3.2 Modelo de mensaje PGP [3]

Función	Algoritmo	Descripción
Firma Digital	RSA/SHA	Se genera un hash mediante SHA-1 y se cifra con la clave privada RSA del remitente, y se lo adjunta al mensaje.
Cifrado de mensaje	CAST o IDEA con ElGamal o RSA	A través de una 'clave de sesión' generada, se cifra el mensaje, y luego se cifra la clave de sesión con la clave pública RSA o ElGamal del destinatario
Compresión	ZIP	Se comprime el mensaje para almacenamiento y/o transmisión.
Compatibilidad de envío	Conversión a base64	Para proveer las facilidades necesarias de transmisión por correo electrónico.

Tabla 3.1 Guía de servicios ofrecidos por PGP [1]

A pesar que RSA es el algoritmo más lento de todo el conjunto, el hecho que solo deba cifrar el hash SHA-1 y la clave de sesión, hace posible que se lo pueda usar con claves de 2048 bits, haciendo mucho más segura la transmisión de los datos. Bajo esta configuración, PGP ofrece integridad, confidencialidad, compresión y autenticación mediante firma digital, lo que lo hace un sistema robusto de seguridad.

Análogamente a S/MIME, PGP y sus versiones soportan dos formas de envío de mensajes;

- Información cifrada (figura 3.3); consistente en información cifrada codificada en base64.

```
To: Karen Nobody <karen@192.168.0.35>
From: Nemo <nemo@192.168.0.35>
Subject: Mensaje Cifrado PGP
Message-ID: <55BADA4C.4030001@192.168.0.35>
Date: Thu, 30 Jul 2015 23:15:40 -0300
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:38.0) Gecko/20100101
 Thunderbird/38.1.0
MIME-Version: 1.0
Content-Type: multipart/mixed;
 boundary="------------070204020909000406070704"

This is a multi-part message in MIME format.
--------------070204020909000406070704
Content-Type: text/plain; charset=utf-8
Content-Transfer-Encoding: 8bit

-----BEGIN PGP MESSAGE-----
Charset: utf-8
Version: GnuPG v2

hQEMA3bfJpNqVd9QAQf+KTCyMbOLTkiYXjHl4HshoAg0xMmnHpIPObQ2ZhC6+iyo
FWZMOy+nH1HCLfTOymgaw9JLLHf1sGSn9og0VYDzwHUp1eo39y8W5DnA6iReOjNN
IoSsZfmaHlgeU7rN8XNZBLmLFm9OZA0VOMjw97JSu588IFr68+MhHrgbjfRGCwA+
ZS0AdlAuCrQus53uTBarbYqHyx8OfSiVilBxonjTPO7KggrJzRTLIr2ZWI3yT0ZE
SXRDZrFzuQAuhGXx9wWTE0YIMFimqMVI7jdnchDMoCPmEamp+vA1/zTS87AuqUVi
BWzCewfuVfMAYkBJ40hlFixHyGkSkx5DFSr/I7inGYUBDAMht5pfJuawVQEIANqy
e8bsDD+MNA5w7KY5lGjTsGXOqTL/Zock2q6fP0pUjjWXBFXvG4Wws1PqUD39sipG
mlIgehHFinMtoFygaC0pAZcM3YU33p0VT9EStmqpjwgDA3jmPxPF9KZl7/ZAwc2k
N+LXNJeHG3W2WDEhWZvC0oftuII7PVYdW4OdfcXw6jAWBkmzcI3cs/RfagMGLXCX
Vo1H66tj85d2hEgkM+Lsgj6UpFuwAy+XwxA3YtnQbVbl5SNlv8hpnSwLU3FEoi37
YSvLvoJEw5XmWzK9AVvOUCpwDk0PyxJ26lgfk7fvfq8lt4G3C4MH7/zThzIgUUnz
CYj0US4wf6UlEUa9TQnSwJABeUBVpz6etpjO3pcQE9TCnuB1lJGluvXlWNt7LwgG
4W0YJF6VhrEUsTDi54k0kbvzvaUYKX4fnbDQRsDp0vEXu39oN6wS7AKuQJeUIc5f
nEecuwSSJ4iPlvmwtQL2LCvGvd9WSQLTa7r9yEqDJ78d/ePaRPRPpaMofLNlieLa
NF6jqF8e0mH7Zh+GBNYoqigxLjE9rZEqbvqRvorA/SHQBB+uh1NaaW0gO20GeH2R
Hre+GPaOfqNVP2bbwge3QkjE1Wa3BPdnZJEEG5P+nk3vXM87di+H5PkaUxz4aX8B
9YdJZM+TOD/vUIx+DM8DrEsJ8TeH3UdLbuGWQONsJNkDO6K8dwdWeSID0Y9yX7ng
JdqqTmMSDMZnDCGVnhHLUjTwj/lW66xZNfwJk5IVpP0WTbqsswhJbYl1kJRrkSeB
bePbo6EqlhUWJJ2bwwBpHJo=
=tyUk
-----END PGP MESSAGE-----

--------------070204020909000406070704
Content-Type: application/octet-stream;
 name="0x7615AB30.asc.pgp"
Content-Transfer-Encoding: base64
Content-Disposition: attachment;
 filename="0x7615AB30.asc.pgp"

hQEMA3bfJpNqVd9QAQgArdoAhF2yI+N9CQtYZol2rq0alb4OuvjvVHFPfUazBuTetehUUTyt
t8KIPjW8Ze5g3/hqaV7BsyPCVzwxS6CWpE2yKwef6hlLvZpA/2FhVd/bv+PTUrKLsG5C3vO8
t2kobsRigNGjBYMXV/rtDSW+U06VJsWAeLaG23INyAB9Gajt3UGpzSNxkaqLQiuhQCTURMHF
EkKloXfElAG9EWjJyxEph+r8TskSTInt5cU2GJU1B6kB2jDSEVOzs1trKPtP6RRQAMpkZxpw
1kH13APgY6vMVMYyi6st9Xei5rc6Bc5VLbQockycB61QuYT+EkRk6OwJvOz2eOmetRInH/ix
74UBDAMht5pfJuawVQEH/Rir7CIuSDYYT99IVXJEuzK+sq8iJTC7bOym1X3vDVXz8MF33LjI
mf6/ouRA1xVLu59rv+LMZGb2njse4aQB8COR2hwSnDJ5cA2IX2NZf6uRniwmDXeM4k0RnJbG
(...)
3YummWQkGimv7LYnmyuTRJfbeLOxfcNDqDVqeqAfonPhJ9d9WOENpBxjhHcYpAe7G16YnFYY
rHTAyBailre+rehBMzxOAy+2FW+hJZc2vuRB2FLZEwkzwYRNTOScOOFEixLCQWu/waRfoQN4
8ne4Wuy2oeVh3NFqBFVXmytEcuxO7uEw3GIO3N8MB22ouy1UVf7UcLXXuix/y4CQUudMa+rL
/AgEG47Lpef2FNf2tYRJCutsRf6UUR7WQ2u/g4yo7o3RdH18GMJJytHntZzMGvEBEhaE8Ym9
bzLe98yxO9GWTPqjzHMKp11VqghC8kZSQTLLIhITJdZAzPZxx2/FeR8F7PjRedMnY+yy7xz7
TN6W7MqApStkdwOTOPKzGPnuD/tiFkyU3aaRA8uUA5GGO/G+I4xpVUD3xKR9Y+OS4WS3suoy
zzbpOvIrkRi4FHeVz7uPuvSd+QvyEPiuiGfES5cR+qYqsFdfoU76BDB9H73abBgycKnEEvN8
LSsyP2Gd728K7ZuYAQ/vUSMeSXgGC0T5569GsXOY/4COMzzBZnL9WGDr9zkY9OFNtNlpzohe
gM5Sfu9FPioY5/4hCkHFzTij
--------------070204020909000406070704--
.
```

Figura 3.3 Ejemplo de mensaje PGP cifrado; que lleva adjunta al mensaje la clave pública del remitente como archivo independiente

- Información firmada (figura 3.4); formada por el extracto del mensaje firmado por la clave privada del remitente adicionado al contenido en plano del mensaje. Puede ser leída por todos los

destinatarios, pero solo comprobada por los destinatarios con capacidad PGP. Además de la firma en base64 en el cuerpo del mensaje, PGP envía un archivo con la firma para comprobaciones del sistema.

```
To: Karen Nobody <karen@192.168.0.35>
From: Nemo <nemo@192.168.0.35>
Subject: Mensaje Firmado PGP
Message-ID: <55B7E848.4080501@192.168.0.35>
Date: Tue, 28 Jul 2015 17:38:32 -0300
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:38.0) Gecko/20100101
 Thunderbird/38.1.0
MIME-Version: 1.0
Content-Type: multipart/mixed;
 boundary="------------050107090203030806010606"

This is a multi-part message in MIME format.
--------------050107090203030806010606
Content-Type: text/plain; charset=utf-8
Content-Transfer-Encoding: 8bit

-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA256

En un agujero en el suelo, vivía un hobbit. No un agujero húmedo, sucio,
repugnante, con restos de gusanos y olor a fango, ni tampoco un agujero
seco, desnudo y arenoso, sin nada en que sentarse o que comer: era un
agujero-hobbit, y eso significa comodidad.
Tenía una puerta redonda, perfecta como un ojo de buey, pintada de
verde, con una manilla de bronce dorada y brillante, justo en el medio.
John Ronald Reuel Tolkien.
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v2

iQEcBAEBCAAGBQJVt+hHAAoJEHKJozX/xFcqCwkH/RaNfmC8Pj6Q5wWcG8sq/lDH
g2EUcbds33GRCOkTA/DMbWtbBCo5hDSCpbT/B1DZvxl0fjDrJLMvYbsvCSow0i1j
qur+k2ZmzjedOIAReuchPLxOt385Sn0OUVvPfOcjjuBSqOikmlnyYnEU1ic6Cd8K
6pIiZtmpKxhmu9kaFzuIvpzNUgeRQUqEGg3GhiqKCw0VbqIWbf0xyc2F4mRRWUdW
n87Z1VfXZ7J6LZiMQcpDrBTx3XfJMWSqMJNvOxPh1FP02DPKQeqcEUOFfgbLV6uF
mdCkCOCW9F67EItXvGrK1p+cKN5c/oIsBWmoPQ83H7X5B9TllPgIwTdmVTlLJbg=
=oXyh
-----END PGP SIGNATURE-----

--------------050107090203030806010606
Content-Type: application/pgp-keys;
 name="0xFFC4572A.asc"
Content-Transfer-Encoding: 7bit
Content-Disposition: attachment;
 filename="0xFFC4572A.asc"

-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2

mQENBFW3wgIBCAC385uh6I6M/og6r22Ou5URLnfnFWR0Cmmfgh7L+cuKZQKp6Z71
83PUDkZJHAH/3UQO0AMdHTmla16ZYIPmvSuJHN4pHEX3xxygN2BXt+otZuLd06eK
ww/abpn0+2u0/kqMXB2SZIn+i1gmDKspHYbL84//lydeCS72THNGQP1l+w9BTeHf
KE4xGmUF7q27JvPJP9tN6M4xo7ejB09mRQbM6f04XGMIiHsOj+luhl5+mTfc9ixF
mup9DMa5LmM5ZUdvvlsJQlY9GPCo7Tt3Pz9K6z8zoilOrqvj9Mi7Rlb+ZhtCEckM
OvMbkaDvTNyGEbDs/Mdj39PExSVNtTHgdtQxABEBAAG0Kk5lbw8gTm9ib2R5IChV
bmRvbWllbCkgPG5lbW9AMTkyLjE2OC4wLjM1PokBOQQTAQgAIwUCVbfCAgIbDwcL
CQgHAwIBBhUIAgkKCwQWAgMBAh4BAheAAAoJEHKJozX/xFcqhIoH/jB9GpN5Rs+E
grnWYrkT6HlKv6UH046ZKCjg3zO6eKEfRIn4fyqmCrEI5p7CNRsPmKbvRKoXyUct
ZBXTahJQk8OdEfXbnwDJ9o3In5cIOoBYVSA/NJ64k/bH5MDjBybj01eKKDua/OG2
zMuFZWybnVHgj3JqlT1m/aIRMhOjbRDBp5rTaQbXGTSy2sX/iB5HvtpLfjv8lG++
Sy9ukDk8OeaKDDjvKJhNRHxX3Kgk4uLssmkhai7cL6QqjvS9b7vwn+exxHKKFEHM
rN1hPRScHzRYSoz7vVPpXDTGGALbyl6pP6b2cGT4Ni6GPy47023jUOvj1ZMkywre
71BWCDbJ5Tg=
=6FH5
-----END PGP PUBLIC KEY BLOCK-----

--------------050107090203030806010606
Content-Type: application/octet-stream;
 name="0xFFC4572A.asc.sig"
Content-Transfer-Encoding: base64
Content-Disposition: attachment;
 filename="0xFFC4572A.asc.sig"

iQEcBAABCAAGBQJVt+hIAAoJEHKJozX/xFcqY2sIAI3YNn4Kgke/HkXSj6YzAmXELdq4dE6b
YznCsHi7xEc+z4HG8tdOPC0soMnczAIjIIs3Q1yNovdZ1FgIv9qgql8gzH7tsEcH4sPOS5A+
DNT2RQ7XHGAxK0A0aBwRUyrTMG/hjCvBB5/SNqTkBGA68iKTjQCFpFsXz2v5xX8W5FpjKthB
LPwy9wkV4wK3HWuZuO6n6FYxw0CLiNN9LBaXloZeRg6qGMZLZuvllCfNbNsx7Zrz5sM3rJk+
oTLKVKi8yG1eYK1IfBgqbzD6ylzV/lxYPa75pdXCuVNGmBv+1X5uGBOJpV53PzvULMFbm6fW
Ul9Bgm+5iDzFZV9IS5j2Rrw=
--------------050107090203030806010606--
.
```

Figura 3.4 Ejemplo de mensaje PGP firmado; que lleva adjunto la clave pública del remitente y la firma del mensaje como archivos independientes

3.1.1 ANILLOS DE CLAVES

Para que el sistema pueda reconocer y administrar las claves públicas y privadas con las que tiene que interactuar al enviar mensajes, se diseñaron dos sistemas conjuntos que se almacenan en el terminal del usuario, conocidos como anillos de claves. El anillo personal de usuario, conocido como anillo de claves privadas almacena los pares de claves generados por el usuario para su uso particular. La segunda estructura, conocido como anillo de claves públicas almacena las claves públicas de los nodos con los que el usuario se comunica. A fin que el remitente conozca que claves debe utilizar al enviar un correo seguro, a cada una de las claves almacenadas en los anillos se le asigna un identificador, conocido como KeyID (Identificador de Clave o Key Identificator, en inglés).

Ambos anillos están estructurados como una base de datos, con el KeyID como identificador único, y en el que, como una tabla de datos, almacena la información propia de cada clave.

El anillo de clave privada contiene en cada 'fila':

- Marca de tiempo (Timestamp); que almacena la fecha y la hora de la creación de las claves.
- KeyID, identificador de clave, los 64 bits menos significativos.
- Clave Pública (Public Key); la clave pública correspondiente al par.
- Clave Privada (Private Key); la clave privada correspondiente al par, se la almacena cifrada.
- Identificador de Usuario (UserID); identificador del poseedor de la clave, habitualmente es un correo electrónico, pero también puede ser un alias con el que asociar cada par de claves.

Para proteger las claves privadas del usuario el anillo las almacena cifradas desde el momento de su creación. El cifrado de la clave privada se realiza con el mismo protocolo con el que se cifró el mensaje enviado, es

decir IDEA o CAST-128. Para su almacenamiento, el usuario debe elegir una clave de cifrado, y a través de SHA-1, se genera un hash de la clave de cifrado, con lo que la clave, sin importar su longitud, se descarta y se utilizan los 160 bits del hash SHA-1 como clave final para el protocolo de cifrado, que luego de cifrar la clave en cuestión se descarta y la clave privada cifrada se almacena en el anillo.

Cuando el usuario requiere el uso de su clave privada para firmar un mensaje, el sistema le solicita la clave de cifrado en texto plano, y realiza el proceso inverso para descifrar la clave privada almacenada; esto es obtener el hash SHA-1 de la clave proporcionada por el usuario y utilizar estos 160 bits sobre la clave cifrada. Este sistema ha probado ser lo suficientemente eficiente para almacenar y asegurar las claves privadas.

El anillo de clave pública contiene en cada 'fila':

- Marca de tiempo (Timestamp); que almacena la fecha y la hora de la creación de las claves.
- KeyID, identificador de clave, los 64 bits menos significativos.
- Clave Pública (Public Key); la clave pública correspondiente al par.
- Confianza del propietario (Owner Trust); nivel de confianza que el usuario le asigna al dueño de la clave pública para que pueda certificar otras claves públicas.
- Identificador de Usuario (UserID); identificador del poseedor de la clave, habitualmente es un correo electrónico, pero también puede ser un alias con el que asociar cada par de claves.
- Legitimidad de clave (Key Legitimacy); nivel de confianza que tiene la clave de un usuario, a mayor nivel de confianza, la asociación entre esta clave y su poseedor será mayor.
- Firma (Signature); firmas de certificación de confianza que posee una clave, sirve para asegurar el valor de legitimidad.

- Firmas de confianza (Trust Signature); indica al usuario el nivel de confianza del firmador para certificar otras claves públicas. Está ligado al campo Owner Trust.

Los campos Key Legitimacy, Signature Trust y Owner Trust comparten una estructura conocida como Byte de Alerta de Confianza (Trust Flag Byte, en inglés), que contiene los valores explicados en la tabla 3.2 y que se configura cuando una nueva clave pública es almacenada en el anillo.

Confianza asignada por el dueño de la clave pública, definida por el usuario	**Confianza asignada por la relación entre la clave pública y el identificador del propietario, calculada por PGP**	**Confianza asignada por firmas de certificación.**
Campo OWNERTRUST: - Confianza indefinida. - Confianza desconocida. - No usualmente confiable para certificar otras claves. - Usualmente confiable para certificar otras claves. - Siempre confiable para certificar otras claves. - Confianza máxima.	Campo KEYLEGIT: - Confianza desconocida o indefinida. - Propietario no confiable. - Confianza incipiente en el propietario. - Total confianza en el propietario.	Campo SIGTRUST: - Confianza Indefinida - Usuario desconocido. - No usualmente confiable para certificar otras claves. - Usualmente confiable para certificar otras claves. - Siempre confiable para certificar otras claves. - Confianza máxima.
Bit BUCKSTOP; Configurado si la clave aparece en el anillo de clave privada.	Bit WARNONLY; Configurado si el usuario quiere ser advertido cuando la clave utilizada no está completamente validada.	Bit CONTIG; Configurado si la firma llega adicionalmente una certificación de confianza máxima en un anillo de clave pública contiguo.

Tabla 3.2 Contenido del Byte de Alerta de Confianza [1]

Cuando una nueva clave pública es almacenada en el anillo de clave pública, PGP le asigna a la clave un valor de confianza asociado con el de su propietario. Si el usuario almacena una clave pública de su propiedad en

el anillo de clave pública, está automáticamente adquiere la máxima confiabilidad posible, caso contrario el usuario deberá asignar niveles de confianza para cada clave pública. A esta nueva clave ingresada, automáticamente se le asignaran una o más firmas de certificación, en función de su propietario; si es conocido o no. En base a las firmas de cada clave, se establece el valor de la legitimidad de clave, y PGP lo calcula en función a la suma de los 'pesos' de cada firma de certificación para determinar su confianza dentro del sistema. Los valores de estos campos son periódicamente modificados según como el usuario y su red de confianza evolucionen al utilizar el sistema. Las figuras 3.5 y 3.6 muestran los valores de anillos públicos y privados en sistemas Windows y Linux, según los despliega PGP. Se puede apreciar como el sistema muestra el directorio donde están almacenados los anillos para cada Sistema Operativo.

```
C:\Windows\System32>gpg --list-keys
C:/Users/Daniel Alejandro/AppData/Roaming/gnupg/pubring.gpg
------------------------------------------------------------
pub   2048R/7615AB30 2015-07-29
uid    [  absoluta ] Nemo Nobody (Undomiel) <nemo@192.168.0.35>
sub   2048R/26E6B055 2015-07-29

pub   2048R/923C28E4 2015-07-29
uid    [desconocida] Karen Nobody (Undomiel) <karen@192.168.0.35>
sub   2048R/720AC8F2 2015-07-29

pub   2048R/B36D8C8F 2015-07-31
uid    [  absoluta ] Karen Nobody (Undomiel) <karen@192.168.0.35>
sub   2048R/6A55DF50 2015-07-31


C:\Windows\System32>gpg --list-secret-keys
C:/Users/Daniel Alejandro/AppData/Roaming/gnupg/secring.gpg
------------------------------------------------------------
sec   2048R/7615AB30 2015-07-29
uid                  Nemo Nobody (Undomiel) <nemo@192.168.0.35>
ssb   2048R/26E6B055 2015-07-29


C:\Windows\System32>
```

Figura 3.5 Anillos públicos y privados PGP en un ambiente Windows

```
clientone@ubuntu:~/Documents/PGP$ gpg --list-keys
/home/clientone/.gnupg/pubring.gpg
----------------------------------
pub   2048R/463C7058 2015-07-28
uid                  Karen Nobody <karen@192.168.0.35>
sub   2048R/4A5A89B8 2015-07-28

pub   2048R/D37BDE6A 2015-07-28
uid                  Karen Nobody (Undomiel) <karen@192.168.0.35>
sub   2048R/7CA27460 2015-07-28

clientone@ubuntu:~/Documents/PGP$
```

Figura 3.6 Anillo publico PGP en un ambiente Linux

De esta manera, el anillo de clave pública se completa y se configura para cada usuario localmente, complementando la información almacenada con la información de los anillos de clave pública que comparte con otros usuarios con los cuales se comunica.

3.2 OPENPGP Y WEB OF TRUST [1][15][16][17]

Al usar IDEA y RSA, protocolos patentados en Estados Unidos, PGP había tenido varios problemas legales de difusión, ya que la ley norteamericana concibe a la criptografía como un arma de guerra, por lo que su exportación estaba completamente prohibida. Esta situación hizo que Zimmermann haya tenido un juicio por no tener la autorización del departamento de Estado para la distribución del software. Eventualmente, y con la liberación al mercado de RSA, el juicio perdió base y fue retirado; sin embargo, los desarrolladores y profesionales de la seguridad buscaban un software que fuere libre de usar.

En 1998, PGP Corp, presentó una derivación de su código base sin protocolos criptográficos patentados a la IETF para que sea convertido en un formato interoperable de Internet, llamándolo OpenPGP. Fue publicado en noviembre de 1998 como el RFC 2440 y se ha mantenido como un estándar en evolución por parte del OpenPGP Working Group.

OpenPGP está actualmente especificado por el RFC 4880, publicado en noviembre de 2007, y complementado por los RFC 5581 publicado en junio de 2009 acerca del uso del protocolo Camellia[4] en PGP y el RFC 6637, publicado en junio de 2012 acerca del uso de criptografía de curvas elípticas en PGP. Esto adicionalmente a las variaciones de PGP para proteger MIME. La figura 3.5 muestra los protocolos con los que cuenta GnuPG, la versión operativa de OpenPGP

[4] Cifrado de bloque de 128 bits. Especificado en el RFC 3713 de abril de 2004.

```
C:\Windows\System32>gpg --version
gpg (GnuPG) 2.0.27 (Gpg4win 2.2.4)
libgcrypt 1.6.3
Copyright (C) 2015 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Home: C:/Users/Daniel Alejandro/AppData/Roaming/gnupg
Algoritmos disponibles:
Clave pública: RSA, RSA, RSA, ELG, DSA
Cifrado: IDEA, 3DES, CAST5, BLOWFISH, AES, AES192, AES256, TWOFISH,
         CAMELLIA128, CAMELLIA192, CAMELLIA256
Resumen: MD5, SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224
Compresión: Sin comprimir, ZIP, ZLIB, BZIP2

C:\Windows\System32>
```

Figura 3.7 Características de la versión 2.0.27 de GnuPG para Windows

La diferencia entre PGP comercial, propiedad de Symantec y OpenPGP, de distribución libre es también el concepto más revolucionario de esta tecnología: el concepto de la Red de Confianza, o Web of Trust.

3.2.1 RED DE CONFIANZA

Parte del éxito de PGP dentro de la red fue la introducción de una nueva forma de validar las claves pública y privada sin necesidad de depender de una entidad certificante. Para conseguirlo, dado que en PGP cada usuario puede generar su propio par de claves, el sistema hizo uso de certificaciones de terceros ‘de confianza’ que le permitan al remitente saber si el destinatario es quien afirma ser. Este sistema de verificación se conoce como Red de Confianza, (Web of Trust, en inglés) y surgió como un método paralelo al estándar X.509 para el manejo de certificados. Es decir, autentificar al dueño de un certificado sin necesidad de una entidad certificante centralizada.

Para el caso de PGP, el manejo de seguridad de los certificados se reduce a la necesidad de poder confiar en que el destinatario posea el par de claves que dice poseer, para que el remitente pueda agregarlas a su módulo PGP y pueda utilizarlas. Dicha confianza se traduce en tres escenarios:

- Los comunicantes se conocen y personalmente, o a través de un medio confiable, intercambian sus claves públicas.
- Los comunicantes conocen a un tercero confiable, que puede validar la autenticidad de las claves públicas y la identidad de sus propietarios.
- Los comunicantes obtienen las claves públicas de una entidad centralizada que maneja las claves públicas de muchos usuarios.

La figura 3.3 muestra la forma en la que PGP maneja su sistema de confianza para validar los certificados que asocian las claves públicas que posee con las identidades de los usuarios con los que quiere comunicarse. En el ejemplo de la figura, las flechas representan quien o quienes han firmado las claves públicas para ser consideradas válidas, diferenciadas entre las que ha firmado directamente el usuario y las que han sido firmadas por terceros de confianza. Además el usuario puede firmar claves que ya han sido previamente firmadas (Nodo E en la figura) haciendo que el nodo propietario de esa clave sea considerado más confiable. Esto demuestra que las consideraciones de confianza para las claves que almacenará el usuario son determinadas por la consideración de confianza del propio usuario, basándose en que nodos las certifican o de donde provienen.

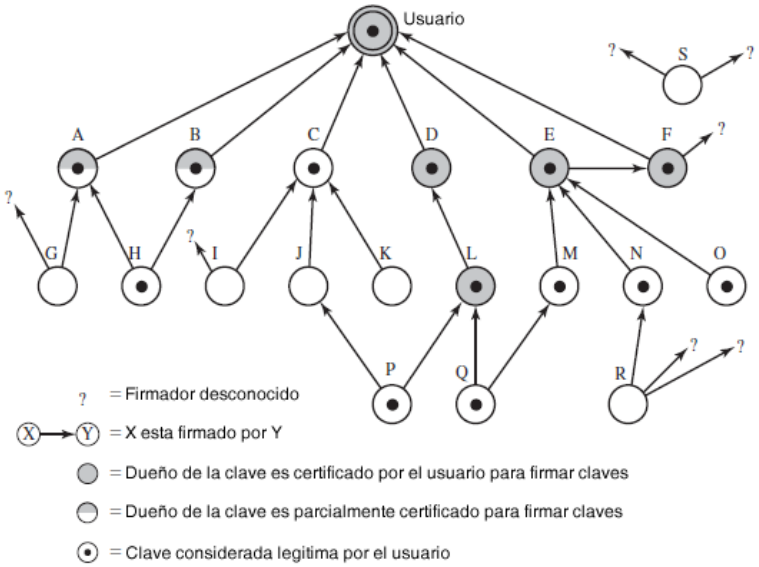


Figura 3.8 Descripción de establecimiento de confianza según Zimmermann [1]

Adicionalmente de las firmas de validación, la figura 3.3 muestra el concepto de firmador de confianza, esto se refiere a que el usuario asigna determinada confianza en un nodo para que este pueda firmar las claves públicas de otros nodos y poder considerarlas confiables o no. De esta manera, por ejemplo, el nodo C posee una clave legítima según el usuario, pero el nodo en sí mismo no es confiable para firmar otras claves públicas, por lo que los nodos I, J y K no son considerados confiables. Siguiendo esa línea, los nodos A y B son parcialmente confiables para firmar claves y los nodos D, E, F y L son confiables para firmar claves.

Pueden existir claves que provengan de entidades no confiables (desde el punto de vista del usuario) pero con las que se puede comunicar porque su identidad está certificada por algún otro nodo confiable para el usuario, como por ejemplo el nodo R, que se encuentra firmado por el nodo N, que el usuario considera de confianza. Adicionalmente existen nodos 'huérfanos', que no son certificados por nadie del círculo de confianza y que pueden ser adquiridos desde entidades certificantes o servidores de claves, en este caso representado por el nodo S. Estas entidades deben demostrar a PGP que son quién dicen ser, ya que el sistema no asume identificación solo por provenir de un servidor considerado seguro. Cabe señalar además que cada nodo puede poseer varias claves públicas asociadas a la misma identidad, lo que hace necesaria su identificación. La figura 3.9 muestra cómo se almacenan las claves públicas y sus firmas en la base de datos pública PGP del Instituto Tecnológico de Massachusetts (http://pgp.mit.edu)

```
Type bits/keyID     cr. time   exp time   key expir
------------------------------------------------------------------------------
pub  4096R/BEE5F60A 2015-02-05

uid Paul Zimmerman <zoomy942@gmail.com>
sig  sig3  BEE5F60A 2015-02-05 __________ __________ [selfsig]
sig  sig3  E09CAFAA 2015-02-05 2019-02-05 __________ Pete Zimmerman <peter.t.zimmerman@gmail.com>

sub  4096R/C90874A4 2015-02-05
sig sbind  BEE5F60A 2015-02-05 __________ __________ []

sub  4096R/19C6D59C 2015-02-05
sig sbind  BEE5F60A 2015-02-05 __________ __________ []
```

Figura 3.9 Ejemplo de claves firmadas almacenadas en los servidores del MIT

La figura 3.9 muestra el registro de la clave pública de Paul Zimmerman con su correo electrónico y su identificador de clave pública (BEE5F60A), y a continuación los identificadores de clave pública (también almacenados en la base de datos) de los usuarios que han firmado la clave de este usuario para asegurar su validez. Según la lógica de la Red de Confianza, mientras más firmas de certificación, más confianza existe sobre las claves al enviar correos seguros.

3.3 IMPLEMENTACIONES ACTUALES [13][14]

A través del estándar OpenPGP se han desarrollado varios aplicativos y plug-ins para clientes de correo que ofrecen todas o la mayoría de las funcionalidades del protocolo. Sin embargo, las que son respaldadas por la OpenPGP Alliance son el conjunto de aplicaciones GNU Project Guard o GPG y el servicio web Hushmail.

GPG fue desarrollado por la Free Software Foundation como un conjunto de instrucciones para línea de comandos que puede correr sobre Windows o sobre distribuciones Linux/Unix, y que utilizada como el estándar de firma y cifrado de Linux, en donde una versión viene instalada por defecto. GPG soporta una variedad de protocolos de cifrado, desde el 3DES hasta los nuevos sistemas de cifrado por curvas elípticas, y utiliza para la compresión del mensaje variaciones menos pesadas computacionalmente de Lempel-Ziv, como ZLIB o BZIP2; como se muestra en la figura 3.10.

Figura 3.10 Características y opciones de uso de GnuPG en su presentación para Windows

Por sí mismo, GPG solo permite cifrar y firmar mensajes o archivos, por lo que necesita de una interfaz que le permita operar con mensajes de correo electrónico. Para este cometido, en su versión para Windows trae un complemento que se integra a Outlook, mientras que en sistemas OpenSource utiliza un complemento conocido como Enigmail, el cual es compatible con los clientes de correo Thunderbird y Seamonkey. GPG no

está diseñado para interactuar con sistemas de correo electrónico sobre Web, como Gmail; sino únicamente a través de un cliente de correo electrónico.

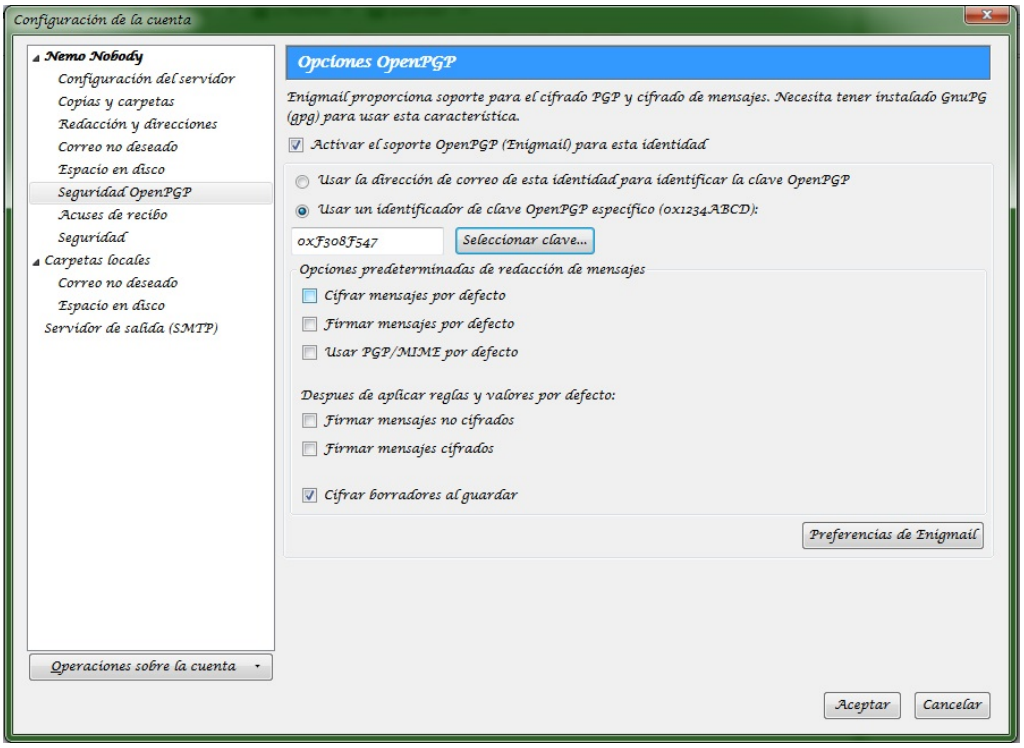


Figura 3.11 Integración de GPG con Enigmail en Thunderbird para Windows

A través de Enigmail es posible utilizar las claves generadas con GPG para el envío de mensajes sin importar el servicio de correo electrónico que se esté utilizando. Paralelamente Enigmail permite la comunicación con determinados servidores de claves públicas PGP para comprobar la veracidad de las claves de los usuarios que están comunicándose.

Adicionalmente Google a través de su comunidad de desarrolladores está diseñando y probando una nueva extensión para Google Chrome llamada ‘End-To-End’, con la cual poder brindar OpenPGP en su servicio de correo electrónico Gmail. Su funcionamiento está pensado para ser una variante de la propia Hushmail, en el que la base de este sistema es la de hacer que los usuarios puedan enviar correos con la seguridad propia de PGP, pero que no tengan que necesariamente preocuparse por el manejo de las claves, que se hará de forma centralizada y transparente por Google. De

esta manera Google pasaría a compartir con el usuario el acceso a la Red de confianza, de manera que el usuario pueda añadir nuevas claves públicas de otros usuarios o generar nuevos pares para sí mismo, pero sin almacenarlas localmente, sino en áreas seguras de los servidores de Gmail, a las cuales el usuario tendrá acceso mediante una contraseña, siempre que previamente se haya registrado como cliente de Gmail o posea algún tipo de certificado de identidad federado, como OpenID.

Este nuevo sistema apenas está en fase de pruebas y Google y su comunidad de desarrolladores aún no han dado una fecha tentativa para lanzar esta extensión a los consumidores.

3.4 PROBLEMAS DE IMPLEMENTACIÓN Y UTILIZACIÓN [6][19][20][21][22][23][24]

OpenPGP fue básicamente diseñado para ser una suerte de caja de herramientas, donde, según los avances y las necesidades propias de la implementación se puedan utilizar diferentes protocolos de seguridad, volviendo a OpenPGP una simple y elegante pero poderosa herramienta de cifrado y aseguramiento de correos electrónicos. Pero también le hizo blanco de varias vulnerabilidades propias de su utilización.

Los anillos de claves, como ya se explicó, se almacenan localmente en la terminal del usuario, y aunque las claves privadas del usuario están cifradas, un atacante podría ganar acceso a la terminal completa y hacerse con los anillos locales, lo que dejaría vulnerables a todas las claves públicas de la red de confianza del usuario, volviéndolo vulnerable a ataques de suplantación.

Para cifrar las claves privadas, PGP utiliza una contraseña escogida por el usuario, la cual es desde el inicio vulnerable a ataques de diccionario o estadísticos, ya que a pesar de la evolución de la seguridad en sistemas informáticos, muchos usuarios siguen utilizando como contraseñas su fecha

de nacimiento, identificador personal o el nombre de sus mascotas. Mientras más sencilla sea la contraseña escogida para proteger las claves privadas, más fácil es que puedan ser adquiridas por un atacante que gane acceso a los anillos de claves.

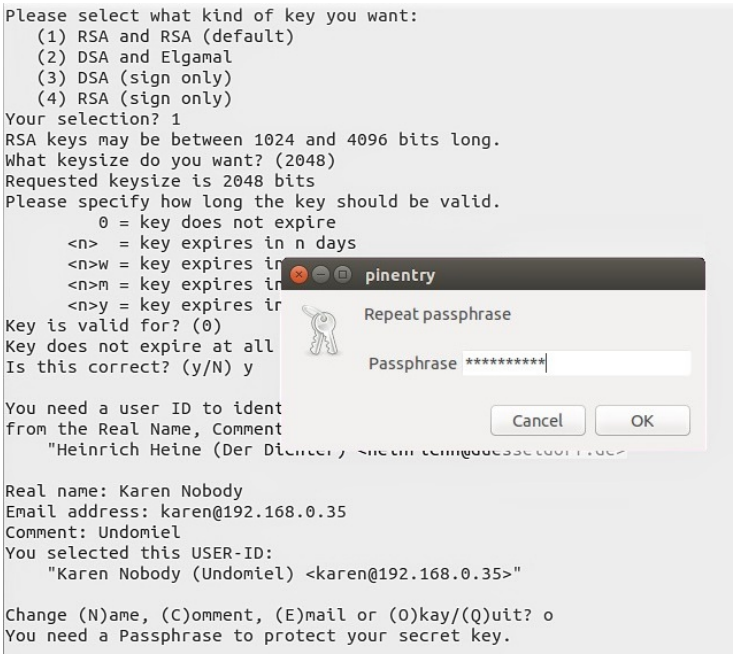


Figura 3.12 Solicitud de clave para cifrar la clave privada al momento de su almacenamiento en el anillo de claves privadas

Si bien el almacenamiento de las claves se protege por una contraseña, los archivos donde se almacenan son relativamente legibles y se almacenan sin ninguna clase de protección en lugares definidos del sistema operativos, como se muestra en las figuras 3.5 y 3.6. Cualquier atacante que gane acceso a la terminal del usuario sabe dónde tiene que buscar para obtener los anillos de claves.

El factor RSA de PGP es vulnerable si la longitud de la clave consiste en menos de 1000 bits. Actualmente las versiones más ligeras de RSA podrían verse comprometidas si no son usadas adecuadamente, por lo que PGP recomienda escoger claves RSA de 2048 bits, ya que esta longitud no afecta el rendimiento al solo cifrar la ‘clave de sesión’ y no un gran volumen de información. A pesar de esto, muchos usuarios escogen un valor menor de bits en RSA para sus contactos de mayor confianza. A partir de ciertas

versiones de GPG, la elección por defecto de una clave RSA es de 2048 bits, como se muestra en la figura 3.12, para evitar que los usuarios utilicen una menor longitud de clave por defecto.

Adicionalmente a estos problemas que podrían considerarse relativos al usuario, existen otros que son más propios del sistema en sí, y es que la implementación de una secuencia relativamente grande de protocolos de seguridad no siempre es sencilla y manejable. En 1999 un estudio conjunto de la Universidad de California y de la Universidad Carnegie Mellon [19] determinó que incluso para personas habituadas a manejar correo electrónico el uso de PGP era confuso y poco manejable.

Los problemas iban más allá del uso de una interfaz gráfica poco amigable, sino que fallaba en darle al usuario todas las opciones correspondientes para el envío de un mensaje. Dentro del universo del estudio, una cuarta parte de los involucrados expuso accidentalmente sus claves privadas, una cantidad similar no pudo acceder a su red de confianza y muchos ni siquiera pudieron ingresar a su propia base de claves públicas. Todo esto además de demostrar que para enviar un mensaje, un usuario relativamente habituado necesita en promedio 90 minutos.

Con OpenPGP estos problemas no se resolvieron, ya que el conjunto de herramientas se volvió un conjunto de instrucciones para ser manejadas a través de consola; y las interfaces graficas disponibles son muy limitadas y no logran una completa conexión con el sistema GPG, sin mencionar la cantidad de opciones configurables de los complementos de los clientes de correo de Linux y Windows. Todo este conjunto de características hace que el cifrado y el envío de mensajes no sean amigables para el usuario, sobre todo para el que no está habituado a usar un sistema computacional a nivel avanzado. Estas situaciones han resultado en brechas de seguridad que han hecho que los sistemas de cifrado no tengan cabida para los usuarios promedio. Glenn Greenwald, periodista del diario inglés The Guardian, que junto con Edward Snowden revelaron el programa PRISM de vigilancia de la

NSA ha dicho en varias entrevistas que le tomó tiempo aprender a utilizar el sistema de OpenPGP para poder comunicarse con Snowden, y que el sistema no poseía la usabilidad necesaria para la importancia que tenía en la comunicación. [30]

La utilización de claves en PGP y luego en OpenPGP siempre ha sido confusa y muchas veces inmanejable, ya que estas claves no fueron diseñadas para que los usuarios las manejaran, lo que hizo que fueran extremadamente largas, y muchas veces, ofrecieran cierta información adicional sobre el tipo de sistema que se estaba utilizando. Aun hoy, con las implementaciones de criptografía de curvas elípticas en OpenPGP, las claves siguen siendo demasiado largas y poco manejables, lo que hace que simplemente se vuelvan inseguras para su utilización por su complejidad, incluso si solo se toma en cuenta que el usuario apenas maneja los KeyID, que es un conjunto de 8 caracteres hexadecimales aleatorios que recibe de servidores de claves o de otros usuarios. Adicional al KeyID, para comprobar que la clave descargada pertenece al usuario en cuestión se utiliza el fingerprint de la clave pública, que son 40 caracteres hexadecimales en grupos de cuatro, con el que, supuestamente, se asegura la identificación de una clave.

```
C:\Windows\System32>gpg --fingerprint
C:/Users/Daniel Alejandro/AppData/Roaming/gnupg/pubring.gpg
------------------------------------------------------------
pub   2048R/7615AB30 2015-07-29
      Huella de clave = EDA9 4F0A 1626 6A9D 3BE3  E514 28CE D36B 7615 AB30
uid    [  absoluta ] Nemo Nobody (Undomiel) <nemo@192.168.0.35>
sub   2048R/26E6B055 2015-07-29

pub   2048R/B36D8C8F 2015-07-31
      Huella de clave = F1EB 6BB8 830A 211C C0B6  7FDC FC15 AE6E B36D 8C8F
uid    [  absoluta ] Karen Nobody (Undomiel) <karen@192.168.0.35>
sub   2048R/6A55DF50 2015-07-31

pub   2048R/5555515A 2015-07-28
      Huella de clave = 1B2E 7AD4 FC73 841C 57E4  B04A DDD6 CD40 5555 515A
uid    [desconocida] nemo uno (test) <nemo@192.168.0.35>

C:\Windows\System32>_
```

Figura 3.13 Ejemplo de los fingerprints de las claves almacenadas en el anillo de claves públicas de un usuario

Para cifrar los mensajes a enviar, es necesaria la clave pública del destinatario, que se puede enviar adjuntada a un mensaje firmado para que el otro usuario la almacene, o buscándola en servidores de almacenamiento

de claves en función de nombres o algún otro identificador que posea el usuario. Esto hace que la identidad del destinatario no pueda ser efectivamente comprobada, ya que cualquier persona puede utilizar cualquier nombre para registrar una clave pública. A partir de 2013, que el nombre de Edward Snowden se hizo conocido alrededor del mundo, muchos entusiastas de internet comenzaron a utilizar el nombre del famoso soplón para registrar sus claves públicas en internet. Esta característica dificulta aún más la obtención de una clave pública fidedigna, en un ambiente en el que ninguno de los usuarios puede certificar su identidad eficazmente.

(a)

(b)

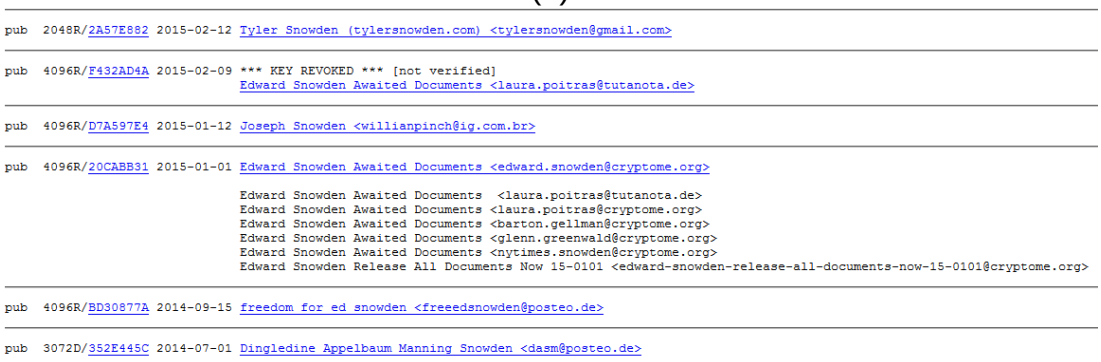


Figura 3.14 (a) y (b) Ejemplo de usuarios que se llaman a sí mismos Edward Snowden en el servidor de claves públicas del MIT

Este hecho sin mencionar que las transacciones de claves publicas entre los anillos, ya sea de carga o descarga se hace a través de un canal inseguro, el cual puede ser fácilmente intervenido durante su transmisión. Adicionalmente, no se realiza una comprobación de integridad entre lo que envió el usuario y lo que recibió el servidor, y los errores que se pudieran producir en la transmisión podrían impedir una adecuada comprobación de claves para su almacenamiento y firma, volviendo inútil una clave que podría

ser válida. En las figuras 3.15 y 3.16 se muestran ejemplos de carga y descarga de claves desde el servidor del MIT, donde se comprueba como viajan las claves a través de internet.

(a)

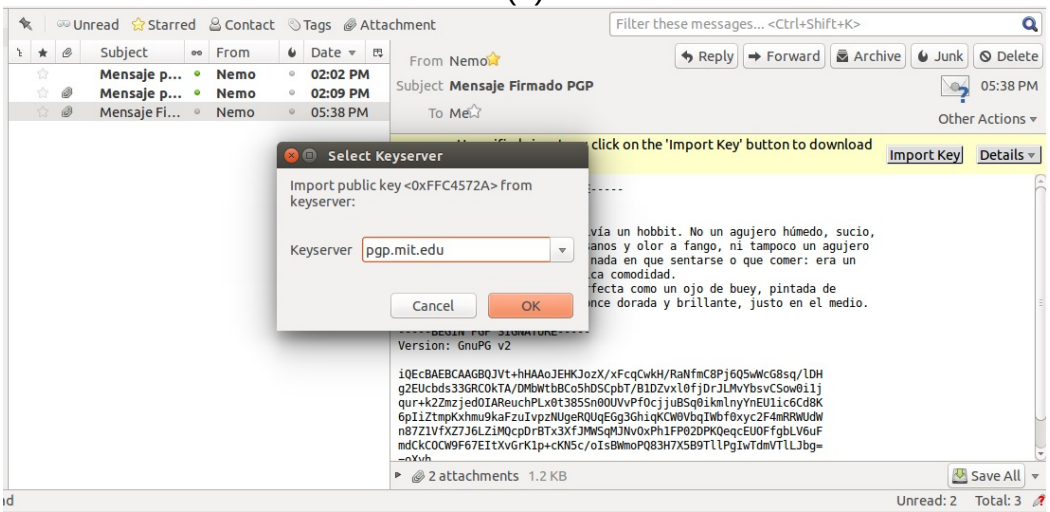
```
D:\Daniel Alejandro\Documents\pruebas>gpg --send-keys --keyserver pgp.mit.edu FFC4572A
gpg: enviando clave FFC4572A a hkp servidor pgp.mit.edu

D:\Daniel Alejandro\Documents\pruebas>_
```

(b)

```
POST /pks/add HTTP/1.1
Host: pgp.mit.edu:11371
Accept: */*
Pragma: no-cache
Cache-Control: no-cache
Content-Length: 1079
Content-Type: application/x-www-form-urlencoded
Expect: 100-continue

HTTP/1.1 100 Continue

keytext=-----BEGIN%20PGP%20PUBLIC%20KEY%20BLOCK-----%0AVersion%3A%20GnuPG%20v2%0A%0AmQENBFW3wgIBCAC385uh6I6M
%2Fog6r22Ou5URLnfnFWROCmmfgh7L%2BcuKZQKp6Z71%0A83PUDkZJHAH%2F3UQOOAMdHTmla16ZYIPmvSuJHN4pHEX3xxygN2BXt
%2BotZuLdO6eK%0Aww%2FabpnO%2B2u0%2FkqMXB25ZIn%2Bi1gmDKspHYbL84%2F%2FlydeCS72THNGQP1l%2Bw9BTeHf
%0AKE4xGmUF7q27JvPJP9tN6M4xo7ejBO9mRQbM6f04XGMIiHsOj%2Bluhl5%2BmTfc9iXF
%0Amup9DMa5LmM5ZUdvvlsJQlY9GPCo7Tt3Pz9K6z8zoilOrqvj9Mi7Rlb%2BZhtCEckM%0A0vMbkaDvTNyGEbDs
%2FMdj39PExSVNtTHgdtQxABEBAAGOkk5lbw8gTm9ib2R5IChv
%0AbmRvbWllbCkgPG5lbw9AMTkyLjE2OC4wLjM1PokBOQQTAQgAIwUCVbfCAgIbDwcL
%0ACQgHAwIBBhUIAgkKCwQWAgMBAh4BAheAAAoJEHKJozX%2FxFcqhIoH%2FjB9GpN5Rs%2BE
%0AgrnWYrkT6HlKv6uH046ZKCjg3zO6eKEfRIn4fyqmCrEI5p7CNRsPmkbvRKoXyUct%0AZBXTahJQk80dEfXbnwDJ9o31n5cIOoBYV5A
%2FNJ64k%2FbH5MDjBybj01eKKDua%2FOG2%0AzMuFZWybnVHgj3JqlT1m%2FaIRMhOjbRDBp5rTaQbXGTSy2sX%2FiB5HvtpLfjV8lG%2B
%2B%0ASy9ukDk8OeaKDDjvKJhNRHxX3Kgk4uLssmkhai7cL6QqjVS9b7vwn%2BexxHKKFEHM
%0ArN1hPRScHzRYSoz7vVPpXDTGGALby16pP6b2cGT4Ni6GPy47023jUOVj1ZMkyWre%0A71BWCDbJ5Tg%3D%0A%3D6FH5%0A-----END
%20PGP%20PUBLIC%20KEY%20BLOCK-----%0AHTTP/1.1 200 OK
Date: Tue, 28 Jul 2015 19:50:48 GMT
Server: sks_www/1.1.5
Cache-Control: no-cache
Pragma: no-cache
Expires: 0
Content-length: 129
X-HKP-Results-Count: 1
Content-type: text/html; charset=UTF-8
Access-Control-Allow-Origin: *
Via: 1.1 cryptonomicon.MIT.EDU:11371
Connection: close

<html><body>Key block added to key server database.  New public keys added: <br>1 key(s) added
successfully.<br></html></body>
```

Figura 3.15 (a) Comando de carga de la clave pública al servidor; y (b) captura de tráfico de Internet del envío de la clave pública

(a)



(b)

```
GET /pks/lookup?op=get&options=mr&search=0xFFC4572A HTTP/1.1
Host: pgp.mit.edu:11371
Accept: */*
Pragma: no-cache
Cache-Control: no-cache

HTTP/1.1 200 OK
Date: Tue, 28 Jul 2015 20:45:34 GMT
Server: sks_www/1.1.5
Cache-Control: no-cache
Pragma: no-cache
Expires: 0
Content-length: 986
X-HKP-Results-Count: 1
Content-type: application/pgp-keys; charset=UTF-8
Content-disposition: attachment; filename=gpgkey.asc
Access-Control-Allow-Origin: *
Via: 1.1 cryptonomicon.MIT.EDU:11371
Connection: close

-----BEGIN PGP PUBLIC KEY BLOCK-----Version: SKS 1.1.5Comment: Hostname: pgp.mit.edu
mQENBFW3wgIBCAC385uh6I6M/og6r22Ou5URLnfnFWR0CmmfgH7L+cuKZQKp6Z7183PUDkZJ
HAH/3UQO0AMdHTmla16ZYIPmvSuJHN4pHEX3xxygN2BXt+otZuLd06eKww/abpn0+2u0/kqMXB2SZIn
+i1gmDKspHYbL84//lydeCS72THNGQP1l+w9BTeHfKE4xGmUF7q27JvPJP9tN6M4xo7ejBO9mRQbM6fO4XGMIiHsOj
+luhl5+mTfc9ixFmup9DMa5LmM5ZUdvvlsJQlY9GPCo7Tt3Pz9K6z8zoilOrqvj9Mi7Rlb
+ZhtCEckM0vMbkaDvTNyGEbDs/Mdj39PExSVNtTHgdtQxABEB
AAG0Kk5lbW8gTm9ib2R5ICh0bmRvbWllbCkgPG5lbW9AMTkyLjE2OC4wLjM1PokBOQQTAQgA
IwUCVbfCAgIDDwcLCQgHAwIBBhUIAgkKCwQWAgMBAh4BAheAAAoJEHKJozX/xFcqhIoH/jB9GpN5Rs
+EgrnwYrkT6HlKv6UH046ZKCjg3zO6eKEfRIn4fyqmCrEI5p7CNRsPmkbvRKoXyUct
ZBXTahJQk80dEfXbnwDJ9o31n5cIOoBYVSA/NJ64k/bH5MDjBybj01eKKDua/OG2zMuFZWyb
nVHgj3JqlT1m/aIRMhOjbRDBp5rTaQbXGTSy2sX/iB5HvtpLfjv8lG++Sy9ukDk8OeaKDDjv
KJhNRHxX3Kgk4uLssmkhai7cL6QqjVS9b7vwn+exxHKKFEHMrN1hPRScHzRYSoz7vVPpXDTG
GALbyl6pP6b2cGT4Ni6GPy47023jUOVj1ZMkyWre71BWCDbJ5Tg==6FH5-----END PGP PUBLIC KEY BLOCK-----
```

Figura 3.16 (a) Método de descarga de clave pública para validación de un correo firmado en Enigmail; y (b) captura de tráfico de Internet de la descarga de la clave pública

Algunos consultores de seguridad consideran que para resolver los problemas que genera el manejo de claves en PGP, además de los inconvenientes de seguridad de la Red de Confianza, es la utilización de otro tipo de entidad certificante para las claves de PGP, tal como lo propone Google para su nueva extensión de seguridad. Sin embargo, este tipo de centralización abre una brecha de seguridad ya que esta entidad centralizada poseería todas las claves de los usuarios, lo que le permitiría conocer el contenido de los mismos y poder prestarlos a entidades gubernamentales si el caso lo requiriera, como ya ha ocurrido en anteriores ocasiones. Además, que la entidad centralizada conozca las claves de los usuarios le permitiría acceder a los mensajes antiguos ya enviados o recibidos que estén almacenados en sus servidores; vulnerabilidad que ya poseía PGP y OpenPGP dado que si un atacante lograba hacerse con la clave privada del usuario, inmediatamente tenía acceso a todo el correo almacenado en su terminal y a todos los archivos que hubieran sido manipulados por estos, es decir, PGP y sus derivados nunca contemplaron el secreto a futuro de la información que se transportaba con el sistema.

Además de estos inconvenientes, se suma uno que es inherente a la construcción propia del sistema y que actualmente es una de sus más

fuertes vulnerabilidades. Como se explicó anteriormente, PGP firma y cifra el mensaje y luego lo 'traduce' a base64 para que pueda ser enviado como un mensaje de correo electrónico cualquiera, en 1991, bajo el RFC 822. Este tipo de configuración deja a los encabezados y otro tipo de metadatos libres de cifrado, lo que hace que viajen a través de internet de manera insegura, como se muestra en la figura 3.17, en la que si bien el mensaje va cifrado por PGP, las direcciones de envío y recepción así como las características del servidor y del cliente que envió el mensaje están en texto legible. En 1991 este tipo de información eran rezagos y no servían más que para hacer direccionamiento, por lo que su valor intrínseco era nulo. Hoy los metadatos del correo electrónico son una fuente valiosa de información para quienes quieran espiar el correo electrónico; y sirven, sobre todo a las agencias gubernamentales, para realizar rastreos, estadísticas y otro tipo de procesamiento de la información, convirtiendo a los metadatos en información casi tan valiosa como el mensaje en sí. Esta situación hace que los comunicantes estén igual de inseguros tanto si envían un mensaje cifrado, como si no, porque los sistemas de vigilancia conocen a quien se envían mensajes, de donde provienen y con qué frecuencia.

```
To: Karen Nobody <karen@192.168.0.35>
From: Nemo <nemo@192.168.0.35>
Subject: Mensaje Cifrado PGP
Message-ID: <55BADA4C.4030001@192.168.0.35>
Date: Thu, 30 Jul 2015 23:15:40 -0300
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:38.0) Gecko/20100101
 Thunderbird/38.1.0
MIME-Version: 1.0
Content-Type: multipart/mixed;
 boundary="------------070204020909000406070704"

This is a multi-part message in MIME format.
--------------070204020909000406070704
Content-Type: text/plain; charset=utf-8
Content-Transfer-Encoding: 8bit

-----BEGIN PGP MESSAGE-----
Charset: utf-8
Version: GnuPG v2

hQEMA3bfJpNqVd9QAQf+KTCyMbOLTkiYXjHl4HshoAg0xMmnHpIPObQ2ZhC6+iyo
FWZMOy+nHlHCLfTOymgaw9JLLHf1sGSn9og0VYDzwHUp1eo39y8W5DnA6iReOjNN
IoSszfmaHlgeU7rN8XNZBLmLFm9OZAOVOMjw97JSu588IFr68+MhHrgbjfRGCwA+
ZS0AdlAuCrQus53uTBarbYqHyx8OfSiVilBxonjTP07KggrJzRTLIr2ZWI3yT0ZE
SXRDZrFzuQAuhGXx9wWTEOYIMFimqMVI7jdnchDMoCPmEamp+vA1/zTS87AuqUVi
BWzCewfuVfMAYkBJ4OhlFixHyGkSkx5DF5r/I7inGYUBDAMht5pfJuawVQEIANqy
e8bsDD+MNA5W7KY5lGjTsGXOqTL/Zock2q6fPOpUjjWXBFXvG4Wws1PqUD39sipG
mlIgehHFinMtoFygaCOpAZcM3YU33pOVT9EStmqpjwgDA3jmPxPF9KZl7/ZAwc2k
N+LXNJeHG3W2WDEfWZvCOoftuII7PvYdW4OdfcXw6jAWBkmzcI3cs/RfaqMGLXCX
Vo1H66tj85d2hEgkM+Lsgj6UpFuwAy+XwxA3YtnQbVbl5SNlv8hpnSwLU3FEoi37
YSvLvoJEw5XmWzK9AVvOUCpwDkOPyxJ26lgfk7fvfq8lt4G3C4MH7/zThzIgUUnz
CYjOUS4wf6UlEUa9TQnSwJABeUBVpz6etpjO3pcQE9TCnuB1lJGluvXlWNt7LwgG
4W0YJF6VhrEUsTDi54k0kbVzvaUYKX4fnbDQRsDpOvEXu39oN6wS7AKuQJeUIcSf
nEecuwSSJ4iPlvmwtQL2LCvGvd9WSQLTa7r9yEqDJ78d/ePaRPRPpaMofLNlieLa
NF6jqF8e0mH7Zh+GBNYoqigxLjE9rZEqbvqRvorA/SHQBB+uh1NaaW0gO20GeH2R
Hre+GPaOfqNVP2bbwge3QkjE1Wa3BPdnZJEEG5P+nk3vXM87di+H5PkaUxz4aX8B
9YdJZM+TOD/vUIx+DM8DrEsJ8TeH3UdLbuGWQONsJNkDO6K8dwdWeSID0Y9yX7ng
JdqqTmMSDMZnDCGVnhHLUjTwj/lw66xZNfwJk5IVpP0WTbqssWhJbYl1kJRrkSeB
bePbo6EqlhUWJJ2bwwBpHJo=
=tyUk
-----END PGP MESSAGE-----
```

Figura 3.17 Mensaje cifrado con PGP con la cabecera en texto plano

Otro eslabón débil dentro de la ya debilitada cadena de OpenPGP, es el uso de la Web of Trust, en apariencia una de las partes más seguras, por

la certificaciones de cada clave pública a ser utilizada. Sin embargo, encierra muchos problemas, ya que su diseño nunca fue pensado para ser escalable o manejable en casos de almacenamiento, por la cantidad de claves existentes. Adicionalmente y como el envío de correos, la Web of Trust no protege adecuadamente los metadatos de las claves que almacena, y realiza las consultas de claves y firmas sin ningún tipo de capa de seguridad, por lo que la página entera puede ser suplantada sin inconvenientes ya que se envía prácticamente en texto plano, como se muestra en la figura 3.18. Pero el principal problema es que las certificaciones de confianza no permiten validar la entera Web of Trust, y dado que el algoritmo con el que funciona pondera el camino más corto para obtener una clave certificada, hace posible que una clave falsificada admitida entre las claves de más confianza permitiría que los grupos de claves de menos confianza, recién ingresadas, que estén certificadas por la clave falsificada se conviertan todas claves destinadas para ataques del tipo MITM[5], orquestados por el creador de la clave falsificada. Esto sobre todo, porque según la configuración de la red, cada una de las claves consideradas de alta confianza funcionan como pequeñas Entidades Certificantes para las claves de menor rango. Esto hace que la Web of Trust pierda completamente su función como entidad independiente de certificación.

(a)

Search results for 'nemo'

Type	bits/keyID	Date	User ID
pub	2048R/7615AB30	2015-07-29	Nemo Nobody (Undomiel) <nemo@192.168.0.35>
pub	2048R/FFC4572A	2015-07-28	Nemo Nobody (Undomiel) <nemo@192.168.0.35>
pub	2048R/5555515A	2015-07-28	nemo uno (test) <nemo@192.168.0.35>
pub	2048R/F308F547	2015-07-25	Nemo Nobody (test) <nemo@undomiel.com>
pub	2048R/85FCB6F6	2015-07-25	Nemo Nobody (test) <nemo@undomiel.com>
pub	4096R/3B9C5ED7	2015-07-24	Nemo <ahmed4055haggag@yahoo.com>
pub	4096R/9A31847D	2015-05-15	Nemo Excellence <reny72@ya.ru> Nemo Excellence <reng72@gmail.com>
pub	2048R/DAC4BFE2	2015-03-24	Sea Cay Nemo <KMKAdmin@TheCollegium.Net>
pub	4096R/B3EA0FF3	2015-03-23	Nemo-blah <nemo@blah.gq>
pub	4096R/F9B0A62B	2015-03-10	Rainer Wahls <o.nemo@posteo.net>
pub	4096R/91DE34BC	2015-02-15	Nemo (lardo) <newark@logitai.com>
pub	4096R/664966E0	2014-12-16	Junghan Kim <cosmos.nemo@gmail.com>

[5] Man in the Middle, u Hombre en el Medio. Famoso ataque de suplantación.

(b)

```
GET /pks/lookup?search=nemo&op=index HTTP/1.1
Host: 18.9.60.141
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:39.0) Gecko/20100101 Firefox/39.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: es,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
DNT: 1
Referer: http://18.9.60.141/
Connection: keep-alive

HTTP/1.1 200 OK
Date: Sun, 26 Jul 2015 04:18:25 GMT
Server: sks_www/1.1.5
Cache-Control: no-cache
Pragma: no-cache
Expires: 0
Content-length: 60694
X-HKP-Results-Count: 232
Content-type: text/html; charset=UTF-8
Access-Control-Allow-Origin: *
Via: 1.1 pgp.mit.edu
Connection: close

<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"
>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Search results for 'nemo'</title>
<meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
<style type="text/css">
/*<![CDATA[*/
 .uid { color: green; text-decoration: underline; }
 .warn { color: red; font-weight: bold; }
/*]]>*/
</style></head><body><h1>Search results for 'nemo'</h1><pre>Type bits/keyID     Date       User ID
</pre><hr /><pre>
pub  2048R/<a href="/pks/lookup?op=get&amp;search=0x50A2C33EF308F547">F308F547</a> 2015-07-25 <a
href="/pks/lookup?op=vindex&amp;search=0x50A2C33EF308F547">Nemo Nobody (test) &lt;nemo@undomiel.com&gt;</a>
</pre><hr /><pre>
pub  2048R/<a href="/pks/lookup?op=get&amp;search=0x696A411C85FCB6F6">85FCB6F6</a> 2015-07-25 <a
href="/pks/lookup?op=vindex&amp;search=0x696A411C85FCB6F6">Nemo Nobody (test) &lt;nemo@undomiel.com&gt;</a>
</pre><hr /><pre>
pub  4096R/<a href="/pks/lookup?op=get&amp;search=0xC4DAA2C13B9C5ED7">3B9C5ED7</a> 2015-07-24 <a
href="/pks/lookup?op=vindex&amp;search=0xC4DAA2C13B9C5ED7">Nemo &lt;ahmed4055haggag@yahoo.com&gt;</a>
</pre><hr /><pre>
pub  4096R/<a href="/pks/lookup?op=get&amp;search=0xF96C6FBE9A31847D">9A31847D</a> 2015-05-15 <a
href="/pks/lookup?op=vindex&amp;search=0xF96C6FBE9A31847D">Nemo Excellence &lt;reny72@ya.ru&gt;</a>
                              Nemo Excellence &lt;reng72@gmail.com&gt;
(...)
pub  4096R/<a href="/pks/lookup?op=get&amp;search=0xC786C86943E18598">43E18598</a> 2012-08-09 <a
href="/pks/lookup?op=vindex&amp;search=0xC786C86943E18598">Matthew Bentley &lt;matthew@mtbentley.us&gt;</a>
                              Matthew Bentley &lt;matt@koalah.co&gt;
                              Matthew Bentley &lt;bentley@riseup.net&gt;
                              Matthew Bentley &lt;matthew@riseup.net&gt;
                              Matthew Bentley &lt;mbentley@lavabit.com&gt;
                              Matthew Bentley &lt;bentley.m@husky.neu.edu&gt;
                              Matthew Bentley &lt;bentley@seattlemesh.net&gt;
                              Matthew Bentley &lt;matthew@seattlemesh.net&gt;
                              Matthew Bentley &lt;matthew@ballardbentleys.com&gt;
                              Matthew Bentley (Ad eundum quo nemo ante iit)
&lt;matthew.t.bentley@gmail.com&gt;
(...)
pub  1024R/<a href="/pks/lookup?op=get&amp;search=0xCCA2F68885907FFD">85907FFD</a> 1995-04-09 <a
href="/pks/lookup?op=vindex&amp;search=0xCCA2F68885907FFD">John Nemo (Nautilus)
&lt;100325.1221@compuserve.com&gt;</a>
</pre><hr /><pre>
pub  1024R/<a href="/pks/lookup?op=get&amp;search=0xAAA322B67239A7F9">7239A7F9</a> 1995-02-09 <a
href="/pks/lookup?op=vindex&amp;search=0xAAA322B67239A7F9">Greg Palmer!
&lt;100325.1221@compuserve.com&gt;</a>
</pre></body></html>
```

Figura 3.18 (a) Resultados de la búsqueda de la cadena de caracteres ‘nemo’ en el servidor del MIT; y (b) captura de tráfico de Internet de la búsqueda mencionada

Aun y a pesar de todas estas fallas, el componente criptográfico que utiliza OpenPGP es lo suficientemente robusto como para que ni siquiera la NSA haya logrado vulnerarlo. Es decir, OpenPGP y sus derivados son seguros por los protocolos que utilizan, pero la filosofía con la que el sistema fue desarrollado ha demostrado su obsolescencia y ha perdido su validez, sobretodo en un mundo donde las condiciones de las comunicaciones requieren de un nuevo enfoque de manejo de la seguridad, con las mismas herramientas, pero utilizadas de forma diferente.

CAPÍTULO 4
CARACTERISTICAS DE UN NUEVO SISTEMA DE CORREO ELECTRONICO SEGURO

Si bien OpenPGP y sus derivados son aun herramientas válidas para asegurar las comunicaciones en internet, sus implementaciones siguen los mismos parámetros con los que fueron diseñados, a principios de 1990. Desde aquel tiempo muchas tecnologías han hecho su aparición y han permitido descubrir las vulnerabilidades propias de PGP, porque en los últimos 20 años la forma en la que utilizamos y convivimos con la red insegura ha cambiado drásticamente.

Muchas de las formas en las que se ha pensado atenuar las vulnerabilidades mantienen los mismos procedimientos de funcionamiento con el que fueron diseñados, por lo que las soluciones no se ajustan más que temporariamente al avance del funcionamiento de las redes y los nuevos esquemas de seguridad. A continuación se describirán las características que debería implementar un protocolo diferente de usabilidad que plantea otro tipo de soluciones para disminuir la complejidad del uso de OpenPGP y su relación con el correo electrónico inseguro.

4.1 WEB OF TRUST

Las vulnerabilidades de la Red de Confianza, explicadas como parte del apartado 3.4, hacen que sea necesario pensar en un nuevo concepto para el manejo de las claves públicas, de tal manera que pueda evitarse que un usuario malintencionado alcance a controlar un grupo de claves públicas.

La forma más sencilla de evitar la suplantación sería renunciar a la independencia de la red basada en la confianza de los usuarios y utilizar una autoridad que certifique la identidad de todos los poseedores de claves, es decir, volver a utilizar Entidades Certificantes, pero la gestión propia de

claves de PGP fue la que la convirtió en la mejor forma de compartir información, sobretodo en lugares donde adquirir una clave puede ser no solo peligroso, sino imposible.

Considerar la necesidad de que toda la Red de Confianza deba ser de alguna manera certificada, implica que toda la red de confianza pueda ser consolidada en un ente distribuido único para su acceso y consulta, o al menos, que existan pocas redes de confianza certificadas que puedan inter-operar entre sí y además considerar el almacenamiento de las claves públicas dentro de los servidores de claves. Actualmente los servidores de claves son solo repositorios de acceso inseguro y no brindan ningún tipo de confianza [20] [23].

Tomando como base el manejo de claves de DKIM, explicado en el apartado 2.2.2, se puede convertir a la red de confianza en un sistema distribuido que sea asegurado por un conjunto de pares de claves propio del sistema y posea características que se integren con los clientes haciendo de OpenPGP un ente hibrido interconectado.

Para este cometido, la nueva red de confianza debería tener la estructura de una base de datos distribuida e interactiva en la que los usuarios del sistema no puedan eliminar o alterar ninguno de los valores que se almacenan en ella, sino solo con la capacidad de agregar nuevos campos, que serán las claves públicas agregadas por los usuarios del sistema. Cada una de las claves públicas almacenadas, por defecto, estará firmada por la clave privada del usuario que la creó y por la clave privada del servidor donde es almacenada, que es el servidor en el que se creó la cuenta de correo electrónico; y según sus relaciones de confianza con otros usuarios, se irán añadiendo firmas de certificación como una característica adicional al sistema. Cada servidor contará con un certificado proporcionado por una Autoridad Certificante (CA), como se explica en el apartado 4.4.

Los parámetros que deberá incluir esta base de datos para el almacenamiento de las claves públicas son:

- Key ID, un identificador unívoco de clave.
- Propietario; correo electrónico del dueño de la clave pública.
- Alias; identificador del propietario de la clave.
- La clave pública, almacenando la clave pública que fue enviada.
- Fingerprint; extracto de la clave pública, derivado de su componente independiente, explicada en el apartado 4.2, almacenado codificado en base64.
- Estatus; indica si la clave está en funcionamiento o ha sido dada de baja
- Timestamp; marca de tiempo en el que fue agregada la clave, sirve para comprobación de entrada única.

Key ID	Propietario	Alias	Clave Publica	Fingerprint	Estatus	Timestamp
Código de identificación único definido por el sistema	Dirección de Correo Electrónico del propietario	Nombre que relaciona al propietario y a la clave	La clave pública almacenada	Extracto de la clave pública, codificado	Habilitado/ Deshabilitado	Fecha/Hora a la que se ingresó el registro

Figura 4.1 Descripción del modelo de almacenamiento de claves públicas.

Adicionalmente a esta tabla del sistema a donde los usuarios no tienen capacidad de modificación, se añade otra en donde se almacenará el grado de confianza que tiene determinada clave pública dentro de la red. Los parámetros de esta tabla son:

- Key ID, el identificador de la clave en cuestión.
- Firmas de confianza, el número de firmas con las que se ha validado la clave en cuestión.
- Timestamp; marca de tiempo del ingreso de la última firma de certificación.

Key ID	Valor de confianza	Timestamp
Código de Identificación de la clave pública.	Valor cardinal de cuantas firmas posee la clave pública, por defecto en 2	Tiempo en que se modificó el campo.

Figura 4.2 Descripción del almacenamiento de la información de confianza.

El algoritmo del sistema permitirá mostrar las claves públicas que posean más firmas de certificación, pero esas firmas de 'más confianza' no volverán a funcionar como CA's intermedias, ya que toda la red estará certificada por sí misma. Desde el punto de vista del sistema, todas las claves ingresadas son iguales; pero las claves con más firmas de certificación son más confiables desde el punto de vista de los usuarios.

En la figura 4.3 se especifica el procedimiento para ingresar una certificación a una clave existente, donde el usuario firmante solicita una validación de la clave, tras lo cual procede a firmarla con su clave privada y enviarla al servidor; todo esto a través de un proceso automatizado. El sistema construye un paquete para el envío, que contiene la clave pública del usuario firmante, la clave firmada, la firma, el alias asociado a esta firma y el alias del firmante; datos que el sistema necesita para validar la certificación. Este paquete es enviado el servidor que contiene la cuenta del usuario firmante y en este servidor se valida que las claves se correspondan a las identidades almacenadas. Si la clave firmada no está almacenada en ese servidor, el sistema hará una búsqueda recursiva hasta ubicar el servidor en el que se encuentra la clave a firmar y una vez validados los datos enviará la información necesaria para almacenar la firma. Cuando todas las claves hayan sido validadas, el sistema comprobará que el firmante no haya firmado ya previamente la clave a firmar, cotejando el archivo de firmadores; en el que se almacenan en un vector los alias y los Key ID's de los firmantes. Si no existiese ningún problema se añade al archivo el alias del firmante con la Key ID de su clave y se modifica el valor cardinal de confianza. Si algún de estos procesos encontrara algún error, el sistema devolvería el mismo al usuario firmante y detendría el proceso.

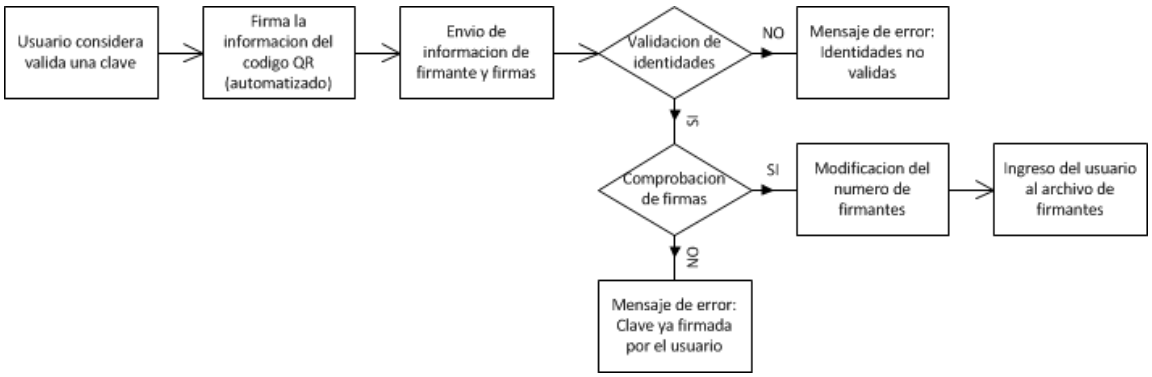


Figura 4.3 Descripción del proceso de firma de confianza para una clave pública.

Desde el punto de vista de la arquitectura, establecer un solo servidor donde se almacenen las claves públicas limitaría el crecimiento de la red, sin importar con cuantos servidores redundantes se pueda conseguir. La mejor forma de administrar este almacenamiento es emular el funcionamiento de DNS, estableciendo un sistema segmentado de arquitectura distribuida, de tal manera que todos los servidores destinados a almacenar las cuentas de correo y las claves publicas asociadas contengan solamente una parte de la totalidad de la red. Cada servidor, por su propia cuenta, deberá tener las necesarias replicas sincronizadas del mismo, para asegurar la completa disponibilidad del sistema.

Se deben establecer servidores de acuerdo al número de usuarios por región geográfica y las cuentas se crearan en función de la cercanía del usuario con un servidor. Estos servidores serán clasificados para su identificación, lo cual permite establecer un índice de los mismos. Adicionalmente cada servidor deberá contar con un sistema de caché de búsqueda de usuarios, para agilizar la búsqueda de claves que no estén contenidas en el servidor donde el usuario creó su cuenta.

La indización de servidores permite limitar y facilitar la búsqueda de claves públicas de usuario. Cuando un usuario necesita validar una clave, el sistema determinará, según el valor del fingerprint, explicado en el apartado 4.2, en que servidor debe estar almacenada la clave pública en cuestión. Por la configuración de los servidores, si una clave no es encontrada en el

servidor que señala el fingerprint, devolverá un mensaje de error frente a la no existencia de la clave.

Para dar de baja una clave pública se realiza un proceso análogo al del ingreso de una nueva clave a la red. Cuando el usuario quiere dar de baja una clave, el cliente previamente hace una comparación en la que comprueba si la clave que se quiere deshabilitar se corresponde con una de las claves privadas almacenadas para el usuario propietario y según su correspondencia, se permite continuar con el proceso. Este proceso se realiza calculando la clave pública a través de la correspondiente privada almacenada. Si la clave calculada se corresponde con la que se pretende eliminar continuará con el proceso, caso contrario devolverá un mensaje de error, como se muestra en la figura 4.4. De esta manera se evita que cualquier usuario pueda deshabilitar claves públicas que no le pertenezcan.

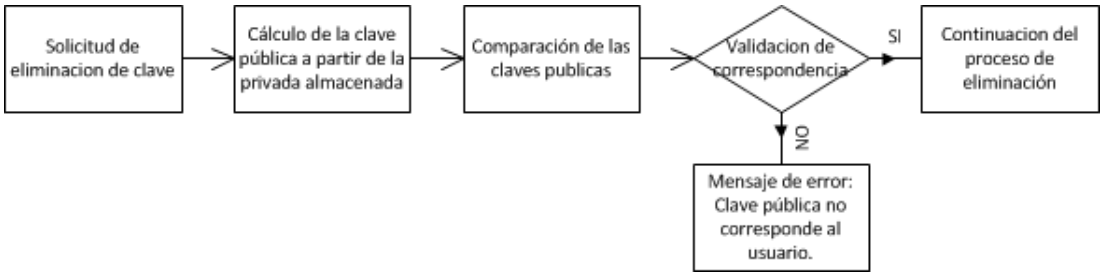


Figura 4.4 Descripción del proceso de validación de correspondencia de clave pública para la eliminación.

Adicionalmente, el cliente almacena junto a las claves privadas las Key ID asignadas a cada clave pública, para complementar la información de eliminación. Cabe aclarar que ningún usuario en ningún momento tiene acceso a las Key ID, que son manejadas automáticamente entre el servidor de claves y el cliente de correo.

Cuando se ha superado esta comprobación, se envían al servidor la Key ID y el alias de usuario, además de una notificación de eliminación, para que la red pueda reingresar estos valores a la base de datos. Cuando el servidor comprueba, mediante la Key ID que la clave existe, elimina del sistema todos los valores asignados a la mencionada Key ID. Si el sistema

no ha detectado ningún error durante la eliminación de la clave pública, automáticamente también eliminará todos los valores de confianza almacenados para esa clave en particular.

Cuando un usuario realice una consulta sobre una clave pública, el sistema siempre le devolverá el registro más nuevo en la red para saber si una clave está habilitada o no. El objetivo de este mecanismo es evitar que una clave pública fuera de servicio pueda seguir siendo utilizada. La eliminación, mostrada en la figura 4.5, será transparente para el usuario, ya que será un proceso automatizado por el cliente instalado en su ordenador.

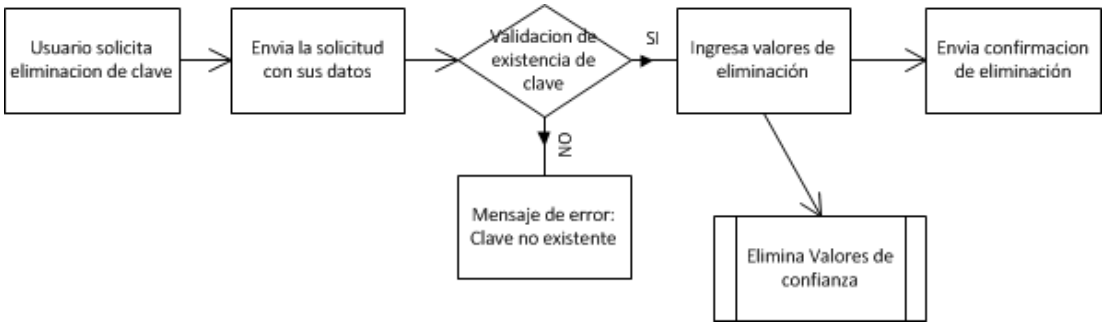


Figura 4.5 Descripción del proceso de eliminación de una clave pública.

4.2 MANEJO E INTERCAMBIO DE CLAVES

Uno de los principales inconvenientes de las implementaciones actuales es que las claves públicas son inmanejables para los usuarios, por la forma en la que PGP las presenta. Sobre todo en los sistemas PGP que no tienen interfaz gráfica, en donde el usuario tiene que ingresar manualmente el *fingerprint* de la clave pública a su anillo privado antes de utilizarlo. Todo esto sin mencionar el problema de asegurar mediante contraseña el anillo de claves localmente, que según la fortaleza de la contraseña elegida por el usuario puede ser más o menos vulnerable a un ataque de acceso y/o suplantación.

Para OpenPGP se diseñó un sistema basado en curvas elípticas, que permite reducir el tamaño de las claves y dejar de utilizar claves tipo RSA, las cuales incluso su *fingerprint* es complicado. A pesar de ser más cortas,

actualmente los usuarios siguen teniendo problemas para manejar una clave que, claramente, no fue diseñada para ser manejada por personas.

Para este nuevo protocolo de usabilidad, se quiere aprovechar las características matemáticas de las curvas elípticas de módulo p [2], que muestra como la clave pública generada está compuesta de dos componentes cartesianas modulo p.

La figura 4.6 muestra de una manera global el proceso de creación y almacenamiento de claves. Cuando el usuario genere un nuevo par de claves, el cliente automáticamente almacenará la clave privada dentro de un espacio de la bóveda segura generada para el almacenamiento de correos electrónicos, explicada en el apartado 4.5, de tal manera que no puedan obtenerse las claves almacenadas si el equipo del usuario fuera vulnerado.

La clave pública generada será firmada con su par privado y se enviará mediante un canal seguro al servidor en donde se ha creado la cuenta de correo del usuario. Una vez en el servidor, la clave será firmada por la clave privada del servidor y con esta firma, se creará el archivo donde se almacenarán los usuarios firmantes de confianza; en principio solo el servidor y el propio usuario. Una vez que la clave pública ha sido almacenada, el servidor debe generar el *fingerprint* que será devuelto al usuario para que pueda compartirlo con sus pares y pueda comunicarse.

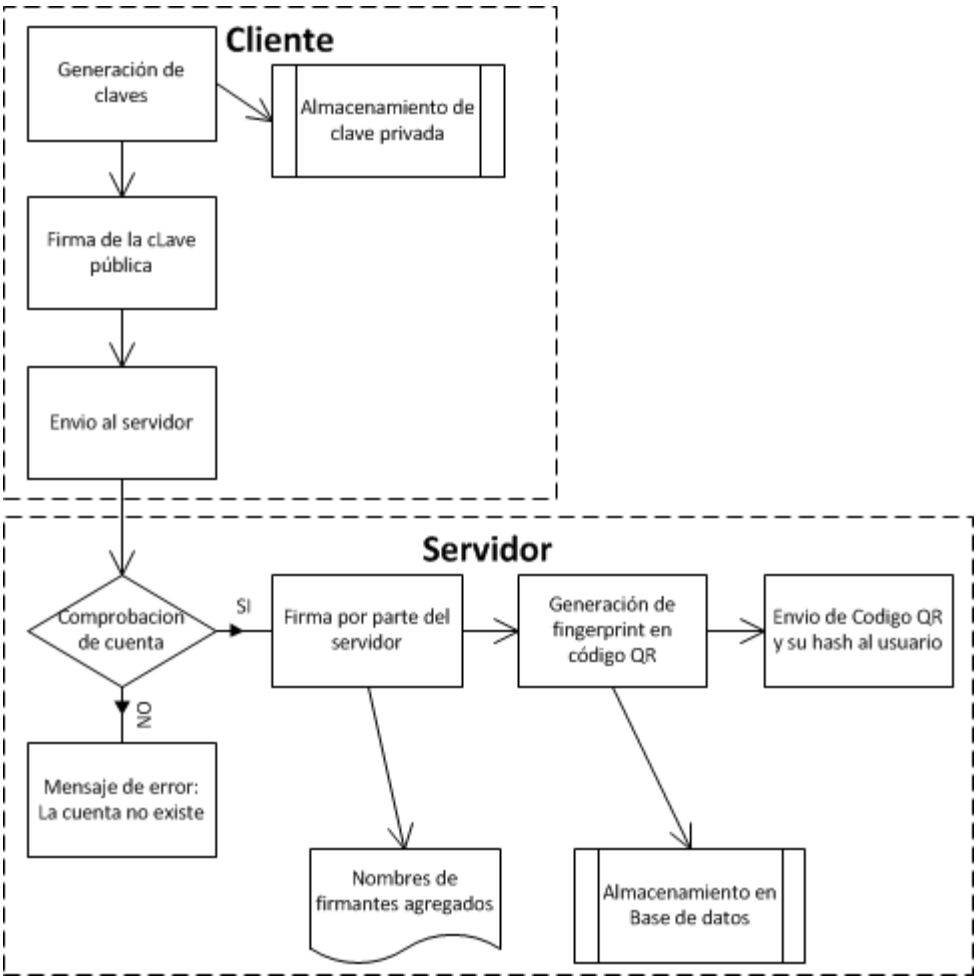


Figura 4.6 Descripción del proceso de creación y almacenamiento de una nueva clave pública.

Para calcular el *fingerprint* de la clave pública, el sistema hará uso de las características matemáticas que posee una clave pública generada por curvas elípticas. Como cada clave pública es una multiplicación modular de un punto generador que pertenece a la curva y un entero grande, en este caso la clave privada, la clave pública es también un punto de la curva elíptica y está formado por dos componentes cartesianas. Conociendo la fórmula precisa de la curva que está siendo utilizada, y la componente independiente de la misma, es posible generar, sin demasiado trabajo de procesamiento, la componente dependiente de la clave pública. Esta característica permite que se pueda utilizar la variable independiente codificada en base64 como el fingerprint de la clave pública a ser compartido. Este proceso de separación de componentes es análogo al

usado por sistemas Bitcoin para establecer las direcciones de las billeteras electrónicas [29].

Sin embargo este fingerprint sigue siendo un conjunto ininteligible de caracteres para las personas, por lo que se lo debe codificar en forma de un código QR. Antes de realizar la transformación a QR, el servidor añadirá dos caracteres hexadecimales al fingerprint en la ubicación menos significativa del mismo, de tal manera que estos caracteres sirvan para identificar que servidor es el que generó el fingerprint y almacena la referida clave pública. La razón de utilizar un código QR radica en que al ser un archivo de imagen, es fácilmente portable por el usuario para su compartición, además que el ser un código QR lo hace ubicuo para la interpretación de la información que representa. Este proceso se muestra en la figura 4.7.

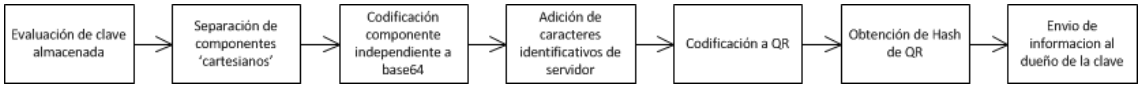


Figura 4.7 Descripción del proceso de codificación de fingerprint.

Esta configuración ayuda a la transmisión de la clave pública a través de internet. En este caso, los clientes de correo sean los que deban calcular la componente faltante para obtener la clave pública completa para enviar mensajes seguros.

Para esto, además, cada cliente vendrá equipado con un lector de códigos QR, con el cual obtener el fingerprint alfanumérico y realizar todas las validaciones y cálculos necesarios para establecer una comunicación.

Para OpenPGP, según la NIST[6], la forma genérica de la curva elíptica utilizada es

$$y^2 mod(p) = x^3 - 3x + b\ mod(p)$$

[6] National Institute of Standards and Technology o Instituto Nacional de Estándares y Tecnología.

Siendo las curvas específicas P-256, P-384 y P-521, cada una con primitivas diferentes para los cálculos respectivos.

Cuando un usuario quiera enviar un correo electrónico, el cliente le pedirá al remitente el código QR que representa a la clave pública del destinatario para su comprobación. El remitente obtendrá el código QR directamente del destinatario, en la forma que este haya escogido para publicar su código QR. El cliente construirá un paquete en el cual envíe la solicitud de comprobación de la clave pública del destinatario. Este paquete contendrá, además de la clave pública y el alias del solicitante, el *fingerprint* codificado en base64 del destinatario, obtenido a partir de la decodificación del código QR, tal como está almacenado en el servidor, el alias del destinatario y su correo electrónico, para que sean verificados por el servidor. Si el servidor verifica que la clave pública existe y se corresponde, envía un mensaje de validación, tras lo cual el cliente decodificará el *fingerprint* hasta obtener la componente de la clave pública y calculará la clave completa del destinatario para establecer la comunicación y enviar los correos electrónicos, tal como se muestra en la figura 4.8.

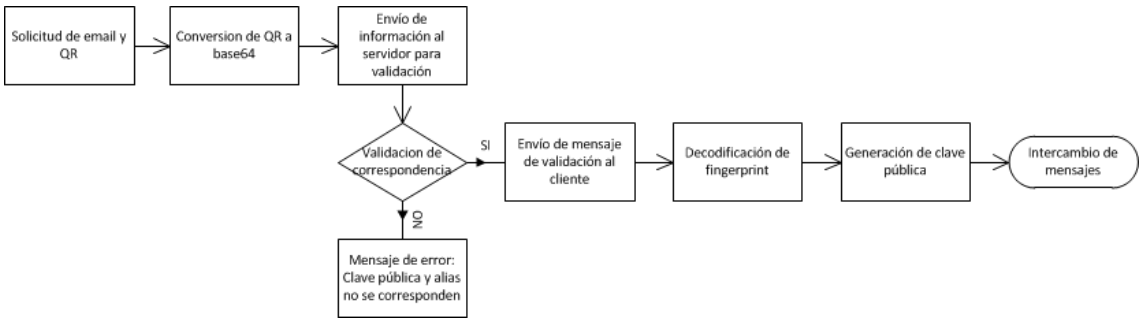


Figura 4.8 Descripción del proceso de envío de correos electrónicos.

De esta manera, y a través de los códigos QR se elimina el intermediario en la adquisición de claves públicas, que consistía en repositorios de claves que no eran necesariamente seguros ni autenticados, y deja al usuario el manejo de sus claves públicas como él lo considere seguro y necesario para sus comunicaciones.

Una de las ventajas importantes de utilizar criptografía de curvas elípticas está en que su tiempo de cálculo no consume ingentes cantidades de procesamiento, por lo que los clientes podrían hacer el cálculo de la clave pública por cada correo que sea necesario de enviar, haciendo posible que el usuario no necesite tener un sistema de almacenamiento adicional de claves públicas de las personas con las que se quiere comunicar. El objetivo de utilizar códigos QR es el de hacer las claves públicas lo más públicas y manejables posibles, de tal manera que su utilización no reporte un problema al usuario, complicando los niveles de seguridad. De la misma manera, el objetivo de almacenar las claves privadas generadas dentro de una bóveda cifrada, hace que estas permanezcan lo más privadas posibles, eliminando también la necesidad de un anillo de claves privadas que contenga todos los pares del usuario.

4.3 SISTEMA DE DIRECTORIO

El hacer que las claves públicas sean lo más públicas posibles, hace que el modelo presentado pueda ser vulnerable al spam. Para solventar este inconveniente, el modelo presentado tendría un sistema de directorio propio que impediría a un *spammer* obtener las direcciones de los usuarios del sistema de manera automática.

Siendo que un usuario puede tener varias claves públicas asociadas a su cuenta de correo electrónico, el sistema de directorio funcionará buscando usuarios según sus alias. Esta configuración funciona además porque existen, entre otros, dos tipos de usuarios de correo seguro: los usuarios que no tienen problema en mostrar su identidad a sus interlocutores, y los usuarios que buscan ocultar su identidad a toda costa. De esta forma se evita obligar al usuario que busque anonimato a revelar su identidad o datos personales, datos que podrían ponerlo en peligro.

Cuando alguien quiera comunicarse con un usuario de este modelo, además de obtener el código QR que identifique la clave pública, deberá

acceder al directorio y buscar al usuario por su alias asociado a la clave pública obtenida. Cuando el cliente decodifique el código QR obtendrá además del fingerprint, un identificador hexadecimal que indica al cliente con que servidor debe comunicarse para buscar la cuenta del destinatario y comprobar la validez de las claves públicas que el destinatario dice poseer. Este proceso de identificación se hace internamente como un proceso del cliente, por lo que para el usuario es transparente.

Para poder acceder al correo electrónico, el interlocutor deberá responder a un desafío para comprobar que es un ser humano, esto es, deberá resolver un CAPTCHA diseñado para el sistema a fin de poder conocer el correo electrónico con el cual quiere comunicarse, como se muestra en la figura 4.9. El sistema almacenará, en una memoria caché, que el usuario ha solicitado la dirección de correo electrónico y validado el CAPTCHA.

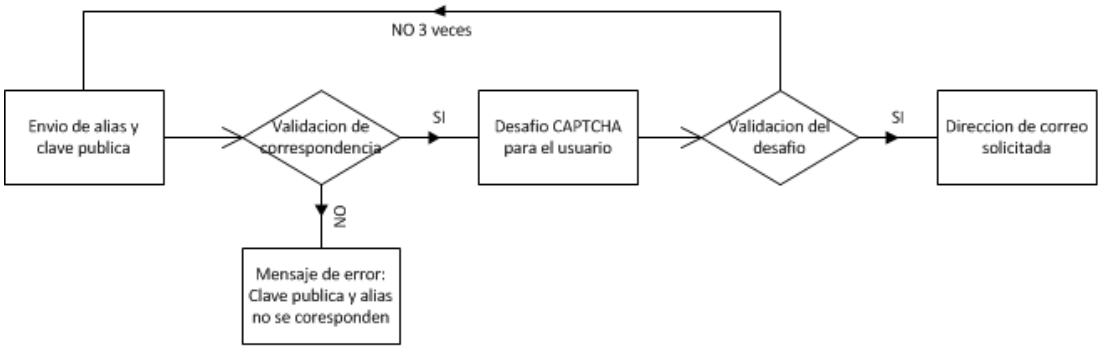


Figura 4.9 Descripción del proceso de obtención de dirección de correo electrónico mediante CAPTCHA.

Si un usuario considerara que un CAPTCHA no es suficientemente seguro, se habilitaría la opción en la cual cuando un interlocutor quiera comunicarse con un usuario del sistema, deberá hacer una solicitud de comunicación, la cual consistirá en ingresar su dirección de correo electrónico, para que le sistema le envíe un correo seguro al usuario informándole que existe alguien que quiere comunicarse con él y que posee una determinada dirección de correo electrónico. De esta manera, si el usuario acepta el pedido de comunicación, podrá ponerse en contacto con el

solicitante para entablar una comunicación segura. Si el usuario considera que el solicitante no es de confianza podrá el mismo descartar la solicitud de comunicación enviada por el sistema. Este tipo de comunicación está especificado en la figura 4.10.

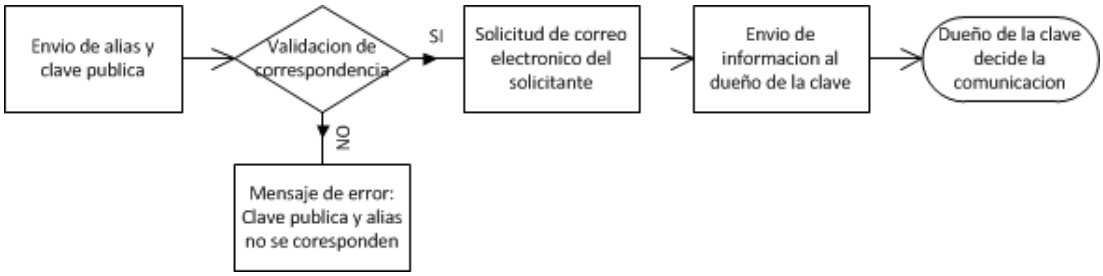


Figura 4.10 Descripción del proceso de obtención de dirección de correo electrónico mediante comunicación con el usuario.

Cuando se ha concretado la comunicación, es decir, se ha enviado un mensaje y este ha sido respondido, el alias y el correo electrónico de los comunicantes se almacenaran en un archivo de contactos, adicionalmente de los datos que los usuarios han proporcionado a la red; es decir, no se podrá modificar nombres o añadir información localmente. Este archivo, almacenado en la bóveda cifrada del sistema, facilitará la comunicación entre usuarios frecuentes, además que evitará la consulta de correos electrónicos recurrentemente, sobre todo si ya se ha establecido la comunicación. Este sistema de almacenamiento es manejado por el cliente automáticamente sin injerencia del usuario. Si en algún momento en el envío de un mensaje, las claves no superan la validación definida en la figura 4.8, el cliente procede a borrar los datos del comunicante del archivo de contactos local.

4.4 CIFRADO DE CANAL

Uno de los valores más importantes para un sistema de correo electrónico seguro es poder contar un una transmisión cifrada de los contenidos de los mensajes y de los metadatos que se generan en cada mensaje. Para asegurar una comunicación cifrada, los servidores en los que se encuentren definidas las cuentas de usuario y almacenadas sus claves

públicas deberán tener obligatoriamente un certificado digital proporcionado por una entidad certificante centralizada, ya que esta comunicación no puede depender de un sistema de confianza, sino que debe asegurar completamente las comunicaciones.

Los clientes de correo vendrán con los certificados adecuados preinstalados para establecer comunicaciones seguras desde que el usuario genera y envía para su almacenamiento su primera clave pública. De igual manera para devolverle al usuario su código QR de una manera segura.

4.5 TRANSMISIÓN Y ALMACENAMIENTO DE CORREOS ELECTRONICOS

Uno de los problemas que enfrentan los sistemas de correo seguro es la intromisión de los organismos gubernamentales de seguridad de los gobiernos a los servidores donde se encuentran las casillas de los correos electrónicos de los usuarios. Toda comunicación que no sea controlada completamente por el usuario es susceptible de ser adquirida por cualquier ente de control.

La mejor forma para evitar este y otros ataques, como el robo de contraseñas de cuentas de correo, es que el propio usuario almacene los correos recibidos de manera segura en su propia terminal, sin depender de ninguna conexión para poder revisar la información que le ha sido enviada. De esta manera este protocolo propuesto utilizará Internet solamente como medio de transporte, y no como un servicio de almacenamiento que puede ser vulnerado.

Para el almacenamiento de correos seguros dentro del terminal del usuario, el programa cliente establecerá una bóveda administrada por el sistema localmente, utilizando para el cifrado del archivo de correos electrónicos una clave IDEA de 128 bits. Para generar esta clave, IDEA utilizará como semilla el tiempo que le toma al cliente instalarse,

conjuntamente con el hash de una contraseña escogida por el usuario, para obtener una clave que establecerá el lugar seguro de los correos dentro del sistema de archivos.

El tiempo de instalación será almacenado en los archivos de configuración finales de la aplicación, de modo que ni el usuario ni ningún atacante puedan acceder a él o modificarlo. El hash adicional se calculara con cada activación de la aplicación cliente, cuando será solicitada la clave proporcionada por el usuario durante la instalación. Si el usuario perdiera u olvidara su contraseña, perdería acceso a su sistema de correos, contactos y claves privadas personales.

Ya que el correo electrónico está basado en un modelo no orientado a conexión, el envío de correos a un usuario desconectado necesita de un espacio de almacenamiento temporal seguro hasta que el destinatario sea capaz de descargar sus mensajes a su terminal. Para este motivo, el servidor en el que se encuentra registrada la cuenta del usuario y almacenadas sus claves públicas deberá contar con un sistema de almacenamiento temporal cifrado por el servidor y sus certificados, además de las seguridades ya añadidas por OpenPGP. Es decir, adicionalmente al procedimiento de cifrado y compresión proporcionado por OpenPGP, cuando se almacene temporalmente en el servidor, también estará cifrada y protegida por la clave pública del servidor en cuestión.

Este almacenamiento temporal de mensajes debe tener, necesariamente, un tiempo de caducidad, tras el cual los mensajes en el servidor deben ser eliminados. El tiempo de caducidad del almacenamiento seguro será establecido por el dueño de la casilla correspondiente según las posibilidades que ofrezca el servidor. El objetivo es siempre evitar a toda costa que el servidor contenga información sensible de sus usuarios en todo momento, asegurando su privacidad, incluso de los propios administradores del servicio.

Cuando el cliente, a través de POP3 Seguro[7] solicita descargar los mensajes a su terminal, estos previamente liberados del cifrado del servidor, se envían a través del canal seguro con el cifrado de OpenPGP para ser descifrados localmente. Si el tiempo de espera en el servidor se ha cumplido, el servidor procederá a borrar los mensajes que no hayan sido descargados, y enviará una notificación al remitente que sus mensajes no pudieron ser leídos.

4.6 RECOMENDACIONES DE IMPLEMENTACIÓN

La necesidad de la existencia de un sistema de seguridad nativo, que no dependa de aplicaciones o complementos para proteger los mensajes que se ha explicado previamente mediante casos de uso, indica como el sistema debería funcionar para proporcionar la seguridad que los usuarios necesitan en Internet.

Con las premisas previamente señaladas, la codificación e implementación de las características y recomendaciones descritas debe realizarse considerando la forma más versátil de presentación para el usuario además de ser confiable en cuestiones programáticas. Es por eso que al momento de plantear el nuevo protocolo de usabilidad no se lo ata a ningún tipo de sistema informático o lenguaje de programación, para que los implementadores puedan tener la libertad de utilizar el que consideren necesario, pero manteniendo las recomendaciones y los casos de uso planteados.

Para la implementación de los casos de uso expuestos, los desarrolladores deberían definir métricas y establecer políticas de seguridad durante el desarrollo y las pruebas del sistema, para de esta manera evitar vulnerabilidades en la aplicación a nivel de código que no sean detectadas a nivel de pruebas. La misión de las métricas de programación es evitar que

[7] Uso de TLS para protocolos POP3 e IMAP, definido en el RFC 2595, de 1999.

puedan quedar puertas traseras o espacios vulnerables a través de los cuales se puedan obtener las claves de los usuarios o la información que se almacena cifrada. La idea de un sistema de seguridad como el aquí planteado es el de brindar confianza a los usuarios para sus comunicaciones a todo nivel, desde la instalación y uso, pasando por las actualizaciones propias del sistema.

Uno de los errores más frecuentes en el uso de cualquier sistema OpenPGP está en que la interfaz de usuario es compleja y no amigable, dando la apariencia de ser apta solamente para usuarios entrenados en programación y codificación; lo que ha hecho que este tipo de sistemas sean vistos con desconfianza por los usuarios normales. Adicionalmente a las recomendaciones expuestas, es necesario que los desarrolladores las presenten de manera amigable a los usuarios para su utilización, de modo que no solo tecnológicamente sea más fácil enviar un correo electrónico seguro, sino también versátil y manejable, sin que opciones confusas dentro de las configuraciones puedan crear brechas de seguridad desde el lado del usuario. No hay que olvidar que mientras un sistema seguro es más fácil de utilizar, hace que sea utilizado correctamente.

Es labor de los implementadores, adicionalmente, utilizar las recomendaciones de esta investigación de tal manera que puedan ser descubiertas y corregidas las posibles fallas que surjan durante el implementación y prueba del nuevo protocolo, para así construir un sistema seguro, como el que los usuarios realmente necesitan, lo que fue el motivo de la realización de este trabajo.

A través de estas recomendaciones se espera constituir un nuevo protocolo que permita que los usuarios puedan integrarse más fácilmente a los sistemas de correo seguros, además de brindarles las facilidades comunicacionales por las que el correo electrónico es importante dentro de un ambiente seguro nativo e intuitivo, permitiendo convertir a la seguridad de una complicación a un modelo de confianza.

CONCLUSIONES

Casi la totalidad de los servicios que funcionan en internet están basados en protocolos que no fueron diseñados para ser universalizados o para proteger la información que estaban transportando. Esta es la razón por la que varios especialistas siempre consideraron Internet como un espacio sin ley, donde cualquiera, sobretodo gobiernos y entidades de control, podían acceder a la información que quisieran sin que existan consecuencias para sus acciones, dejando a los usuarios, junto con su información, indefensos. Aumentar la seguridad de las comunicaciones ha sido un desafío con el que se han enfrentado todos los desarrolladores de aplicaciones y casi todos los usuarios, en mayor o menor medida.

Para la realización de este trabajo, se planteó un análisis del funcionamiento de correo electrónico y sus vulnerabilidades inherentes; además de las características actuales de la seguridad de correo electrónico, sus protocolos y configuraciones. A partir de estos datos, se pretende definir un nuevo método con el cual el correo electrónico seguro pueda funcionar cumpliendo características criptográficas desde su constitución, permitiendo una comunicación más fluida y segura.

Para iniciar el análisis, en el apartado 2.1 se especificaron las características de seguridad que debe cumplir el correo electrónico, como autenticación, privacidad y anonimato. La explicación de cada una de éstas características permite entender por qué es necesario que las mismas se apliquen en el funcionamiento del correo electrónico en Internet y de cómo los algoritmos, explicados en el capítulo 2 y sobretodo en el capítulo 3, muestran los intentos de aplicar las características de seguridad a un entorno claramente inseguro, y que, por defecto, necesita mantener compatibilidad con modelos de correo electrónico antiguos o desfasados, que no están necesariamente asegurados.

Con este cometido, esta investigación se planteó como hipótesis establecer un nuevo sistema de correo electrónico con seguridad nativa a través de sistemas de encriptación y seguridad como PGP permitirá que las comunicaciones puedan satisfacer los parámetros de seguridad que aplica la criptografía moderna en las comunicaciones brindando seguridad a los usuarios y a sus datos. Esta hipótesis se demuestra mediante el desarrollo de un nuevo protocolo de usabilidad para el correo electrónico, que aproveche las fortalezas de OpenPGP pero que permita eliminar las complejidades propias de su utilización.

Este nuevo protocolo de usabilidad viene definido en el capítulo 4, donde se ejemplifica, mediante casos de uso, la forma en la que debería comportarse un sistema de correo seguro siguiendo las directrices de seguridad establecidas por la criptografía. Las características recomendadas para este nuevo protocolo definen, implícitamente, el acercamiento del protocolo completo a una seguridad nativa, de la que el usuario puede gozar desde el momento de la creación de su cuenta de correo, y no a través de extensiones u otro tipo de añadidos al sistema de correo, que sin importar cuánta seguridad aseguren brindar, la complejidad con la que funcionan no es más que una traba. Estas características, adicionalmente, incluyen la posibilidad del usuario de escoger entre la completa autenticación y el completo anonimato, sin perder en ningún momento la integridad de los mensajes enviados, dándole al usuario de este nuevo sistema completo control de sus datos y comunicaciones.

Sin importar la forma de implementación de este nuevo protocolo, aun y con los canales cifrados que se proponen, envía los mensajes por el protocolo SMTP. No se puede olvidar que este nuevo protocolo planteado es parte de un universo más grande, Internet, y que como tal tiene que convivir con sistemas antiguos y poco seguros. No se debe olvidar que todo sistema de seguridad es tan seguro como su componente más insegura, y por lo tanto se debe tener en consideración otros métodos de aseguramiento de información.

Una implementación de seguridad completa necesariamente debería ser planteada sobre el protocolo base SMTP; sin embargo SMTP, por sus propias características y masificación, no puede ser actualizado sin considerar una evolución completa de la red y de los protocolos sobre los cuales está basada. Aun para basar SMTP directamente en PGP o en cualquier otro algoritmo de seguridad, es necesario pensar más allá y evaluar cómo esta actualización afectará el funcionamiento de toda la red. Encontrar la mejor manera de hacer evolucionar los protocolos de Internet es tal vez la tarea más difícil en este momento, por sus implicaciones, pero una tarea obligatoria si se quiere que la compatibilidad de los protocolos base con sistemas como el de este trabajo no presenten las actuales brechas de seguridad.

Antes que en Internet terminen de concretarse ideas como el ‘Big Data’ y el ‘Internet de las cosas’ es necesario comenzar a replantearse los pilares del uso de la red y cómo esta evolución podría afectar nuestra forma de comunicarnos a mediano y largo plazo. No hay que dejar de considerar que la usabilidad de Internet ha cambiado drásticamente varias veces desde su primera masificación hasta su actual modelo de usabilidad y seguirá cambiando según las necesidades comunicacionales de los internautas. Es por eso que desarrolladores y usuarios en conjunto deben establecer parámetros de funcionamiento de Internet en general. Todos los sistemas que dependan de la red van necesitar ser replanteados en algún momento para que puedan evolucionar, y se vuelve imperioso pensar en su evolución antes que Internet, en general, no pueda manejar nuestro mundo virtual.

BIBLIOGRAFÍA ESPECÍFICA

[1] STALLINGS, William. CRYPTOGRAPHY AND NETWORK SECURITY PRINCIPLES AND PRACTICE. Pearson Education Inc. as Prentice Hall. 2011.

[2] STALLINGS, William. CRYPTOGRAPHY AND NETWORK SECURITY PRINCIPLES AND PRACTICE. Pearson Education Inc. 2014.

[3] TANENBAUM, Andrew S.; WETHERHALL, David J. COMPUTER NETWORKS. Pearson Education Inc. as Prentice Hall. 2011.

[4] KUROSE, James F.; ROSS, Keith W. COMPUTER NETWORKING: A TOP DOWN APPROACH. Pearson Education Inc. as Addison-Wesley. 2013.

[5] KAUFMAN, Charlie; PERLMAN, Radia; SPECINER, Mike. NETWORK SECURITY: PRIVATE COMMUNICATION IN A PUBLIC WORLD. Prentice Hall. 2002.

[6] BAUER, Craig P. SECRET HISTORY: THE STORY OF CRYPTOLOGY. CRC Press. 2013.

[7] Apuntes de clase y Material teórico de Criptografía I. Dr. Hugo Scolnik. Maestría de Seguridad Informática. Facultad de Ciencias Exactas. Universidad de Buenos Aires.

[8] DAVENPORT, David. ANONYMITY ON THE INTERNET: WHY THE PRICE MAY BE TOO HIGH. Communications of the ACM. 2002. Vol. 45 No. 4.

[9] CLAESSENS, Joris; PRENEEL, Bart; VANDEWALLE, Joos. SOLUTIONS FOR ANONYMOUS COMMUNICATION ON THE INTERNET. IEEE.1999.

[10] GABBER, Eran; GIBBONS, Phillip B.; MORRIS KRISTOL, David; POTOMAC, Matias, MD; MAYER, Alain J. SYSTEM AND METHOD FOR PROVIDING ANONYMOUS REMAILING AND FILTERING. Patent No.: US 6,591,291 B1. Julio de 2003.

[11] The TOR Project; https://www.torproject.org/about/overview.html. (Consultado el 21 de mayo de 2015)

[12] GOLDSCHLAG, David; REED, Michael; SYVERSON, Paul. ONION ROUTING FOR ANONYMOUS AND PRIVATE INTERNET CONNECTIONS. Communications of the ACM. 1999. Vol. 42, No. 2.

[13] The Free Software Foundation. GUIA DE “GNU PRIVACY GUARD”.1999

[14] Google’s ‘end-to-end’ extension. (Consultadas el 21 de mayo de 2015)

- https://github.com/google/end-to-end/wiki/Key-Distribution
- https://github.com/google/end-to-end/wiki
- https://github.com/google/end-to-end/wiki/Keyring

[15] LUCAS, Michael W. PGP & GPG: EMAIL FOR THE PRACTICAL PARANOID. No Starch Press Inc. 2006.

[16] OpenPGP for Complete Beginners. Zachary Voase. http://zacharyvoase.com/2009/08/20/openpgp/. (Consultado el 21 de mayo de 2015)

[17] The OpenPGP Alliance. http://www.openpgp.org/about_openpgp/. (Consultado el 21 de mayo de 2015)

[18] RFC 4880. OpenPGP Message Format. Internet Engineering Task Force. 2007.

[19] Whitten, Alma; Tygar, J. D. WHY JOHNNY CAN’T ENCRYPT: A USABILITY EVALUATION OF PGP 5.0. School of Computer Science. Carnegie Mellon University. 1999.

[20] ¿What's the matter with PGP?. Mathew Green. http://blog.cryptographyengineering.com/2014/08/whats-matter-with-pgp.html. (Consultado el 21 de mayo de 2015)

[21] GPG And Me. Moxie Marlinspike. http://www.thoughtcrime.org/blog/gpg-and-me/. (Consultado el 21 de mayo de 2015)

[22] The New Yorker: The Daunting Challenge of Secure E-mail. Mathew Green. http://www.newyorker.com/tech/elements/the-daunting-challenge-of-secure-e-mail. (Consultado el 21 de mayo de 2015)

[23] Why the Web of Trust Sucks. Mike Perry. https://lists.torproject.org/pipermail/tor-talk/2013-September/030235.html. (Consultado el 21 de mayo de 2015)

[24] Cryptography Expert Says, 'PGP Encryption is Fundamentally Broken, Time for PGP to Die'. Wang Wei. http://thehackernews.com/2014/08/cryptography-expert-pgp-encryption-is_19.html. (Consultado el 21 de mayo de 2015)

[25] RFC 6637. ELLIPTIC CURVE CRYPTOGRAPHY (ECC) IN OPENPGP. Internet Engineering Task Force. 2012.

[26] FEDERAL INFORMATION PROCESSING STANDARDS. Digital Signature Standard (DSS). Information Technology Laboratory. National Institute of Standards and Technology. 2013.

[27] National Institute of Standards and Technology. MATHEMATICAL ROUTINES FOR THE NIST PRIME ELLIPTIC CURVES. 2010.

[28] American National Standards Institute. ANSI X9.62:2005 - THE ELLIPTIC CURVE DIGITAL SIGNATURE ALGORITHM (ECDSA). 2005.

[29] ANTONOPOLUS, Andreas M. MASTERING BITCOIN. O'Reilly Media Inc. 2010.

[30] NSA-proof encryption exists. Why doesn't anyone use it?. Timothy B. Lee. http://www.washingtonpost.com/news/wonkblog/wp/2013/06/14/nsa-proof-encryption-exists-why-doesnt-anyone-use-it/ (Consultado el 11 de agosto de 2015)

BIBLIOGRAFÍA ADICIONAL

- Institute of Electrical and Electronic Engineers. IEEE P1363a/D4 - STANDARD SPECIFICATIONS FOR PUBLIC KEY CRYPTOGRAPHY: ADDITIONAL TECHNIQUES. 2000
- KOBLITZ, Neal. A COURSE IN NUMBER THEORY AND CRYPROGRAPHY. Springer-Verlag. Second Edition. 1994. ISBN 3-540-94293-9.
- Wikipedia: Secure Hash Algorithm. http://es.wikipedia.org/wiki/Secure_Hash_Algorithm. (Consultado el 21 de mayo de 2015)
- Wikipedia: CAST-128. http://es.wikipedia.org/wiki/CAST-128. (Consultado el 21 de mayo de 2015)
- Wikipedia: GPG. http://es.wikipedia.org/wiki/GNU_Privacy_Guard. (Consultado el 21 de mayo de 2015)
- Wikipedia: PGP. http://es.wikipedia.org/wiki/Pretty_Good_Privacy. (Consultado el 21 de mayo de 2015)
- Wikipedia: Key Management. http://en.wikipedia.org/wiki/Key_management. (Consultado el 21 de mayo de 2015)
- Wikipedia: ElGamal Signature Scheme. http://en.wikipedia.org/wiki/ElGamal_signature_scheme. (Consultado el 21 de mayo de 2015)
- Wikipedia: ElGamal encryption. http://en.wikipedia.org/wiki/ElGamal_encryption. (Consultado el 21 de mayo de 2015)
- Wikipedia: Forward Secrecy. http://en.wikipedia.org/wiki/Forward_secrecy. (Consultado el 21 de mayo de 2015)
- Wikipedia: Secure SMTP. http://en.wikipedia.org/wiki/SMTPS. (Consultado el 21 de mayo de 2015)
- Wikipedia: StartTLS. http://en.wikipedia.org/wiki/STARTTLS. (Consultado el 21 de mayo de 2015)

- Wikipedia: TLS. http://en.wikipedia.org/wiki/Transport_Layer_Security. (Consultado el 21 de mayo de 2015)
- Wikipedia: Electronic Mail. https://en.wikipedia.org/wiki/Email. (Consultado el 21 de mayo de 2015)
- Wikipedia: Email Encryprion. http://en.wikipedia.org/wiki/Email_encryption. (Consultado el 21 de mayo de 2015)
- Wikipedia: Criptografía de Curvas Elipticas. http://es.wikipedia.org/wiki/Criptograf%C3%ADa_de_curva_el%C3%ADptica. (Consultado el 21 de mayo de 2015)
- Wikipedia: DNS. http://es.wikipedia.org/wiki/Domain_Name_System. (Consultado el 21 de mayo de 2015)
- Email Self Defense; The Free Software Foundation. https://emailselfdefense.fsf.org/en/. (Consultado el 21 de mayo de 2015)
- The GNU Privacy Guard. https://www.gnupg.org/index.html. (Consultado el 21 de mayo de 2015)
- ¿WHY DO YOU NEED PGP? by Phil Zimmermann. http://www.spectacle.org/795/byzim.html. (Consultado el 21 de mayo de 2015)
- A Critique Of Lavabit. Moxie Marlinspike. http://www.thoughtcrime.org/blog/lavabit-critique/. (Consultado el 21 de mayo de 2015)
- PGP 6.5.8. http://www.pitt.edu/~poole/PGP.htm. (Consultado el 21 de mayo de 2015)
- Glenn Greenwald: How I Met Edward Snowden. http://www.motherjones.com/politics/2014/05/glenn-greenwald-no-place-to-hide-book-excerpt (Consultado el 10 de agosto de 2015)
- Postfix Configuration Parameters. http://www.postfix.org/postconf.5.html#resolve_numeric_domain (Consultado el 10 de agosto de 2015)
- Setup mail server on Ubuntu 14.04 (Postfix – dovecot). http://www.krizna.com/ubuntu/setup-mail-server-ubuntu-14-04/ (Consultado el 10 de agosto de 2015)

- Enigmail Documentation. https://www.enigmail.net/documentation/index.php (Consultado el 10 de agosto de 2015)
- Hushmail Documentation. https://www.hushmail.com/about/technology/ (Consultado el 10 de agosto de 2015)
- Quick And Easy Setup For DomainKeys Using Ubuntu, Postfix And Dkim-Filter. https://www.howtoforge.com/quick-and-easy-setup-for-domainkeys-using-ubuntu-postfix-and-dkim-filter (Consultado el 10 de agosto de 2015)
- How to Setup DKIM (DomainKeys) with Postfix on Ubuntu & Debian. http://tecadmin.net/setup-dkim-with-postfix-on-ubuntu-debian/ (Consultado el 10 de agosto de 2015)
- Setup DKIM (DomainKeys) for Ubuntu, Postfix and Mailman. http://askubuntu.com/questions/134725/setup-dkim-domainkeys-for-ubuntu-postfix-and-mailman (Consultado el 10 de agosto de 2015)
- How to configure DNS server in Ubuntu 14.04. www.krizna.com/ubuntu/configure-dns-server-ubuntu-14-04/ (Consultado el 10 de agosto de 2015)
- Configuración de servicio de nombres de Dominio. https://help.ubuntu.com/14.04/serverguide/dns-configuration.html (Consultado el 10 de agosto de 2015)
- Firmado y cifrado de e-mail con S/MIME y PGP. http://blog.s21sec.com/2009/06/firmado-y-cifrado-de-e-mail-con-smime-y.html (Consultado el 10 de agosto de 2015)
- GnuPG Howto. https://help.ubuntu.com/community/GnuPrivacyGuardHowto (Consultado el 10 de agosto de 2015)
- Por qué necesitas TOR. http://spn314.blogspot.com.ar/2012/08/por-que-necesitas-tor.html (Consultado el 10 de agosto de 2015)
- Emkei’s Fake Mailer. https://emkei.cz/ (Consultado el 10 de agosto de 2015)