Tesis de Ingeniería en Informática

Análisis de vulnerabilidades esteganográficas en protocolos de comunicación IP y HTTP

Autor: Pablo Andrés Deymonnaz (81.755)
pdeymon@fi.uba.ar

Directora: Lic. Adriana Echeverría

Marzo 2012

Análisis de vulnerabilidades esteganográficas en protocolos de comunicación IP y HTTP

Resumen

La *esteganografía* es la técnica y arte de ocultar la comunicación de mensajes. Esto se logra a partir de *embeber* un mensaje que se desea comunicar dentro de otro mensaje, denominado portador, el cual simula ser la comunicación real.

El hecho de embeber información no debe afectar la interpretación que realiza el receptor sobre el portador. Es decir, la comunicación de los mensajes portadores debe desarrollarse como si no se embebieran los mensajes esteganográficos.

En el presente trabajo se analiza de manera pormenorizada la especificación de los protocolos de comunicación IP y HTTP para proponer la utilización de los mismos con fines esteganográficos. Se proponen formas de explotar mecanismos y estructuras de los mensajes de los protocolos que permitan comunicar información embebida de manera inadvertida (i.e. por fuera de lo establecido en el protocolo, y sin levantar sospecha para una tercera persona con acceso al canal de comunicación).

Asimismo, se presenta un artefacto de software desarrollado para mostrar la factibilidad, tanto de la comunicación esteganográfica cuanto del bloqueo a la misma.

1. Introducción

Se denomina *esteganografía* a la técnica de incorporar mensajes que se desea mantener secretos dentro de otros datos, llamados *portadores* (del inglés *carriers*)[1].

El objetivo de la esteganografía consiste en la comunicación de mensajes entre dos pares (en el sentido de iguales), a través de un canal de comunicación, de modo que el propio acto de la comunicación pase *inadvertido* por quienes puedan tener acceso a ese canal. Este concepto radica en la teoría de la *seguridad por oscuridad*[6], que supone que si nadie conoce que existe un mensaje oculto, nadie tratará de obtenerlo. El *canal* es simplemente el medio utilizado para transmitir el mensaje desde el emisor hacia el receptor.

Esta práctica es posible gracias a la existencia de *canales encubiertos*, este concepto fue introducido por primera vez en 1973 por Butler W. Lampson[4]. Se define *canal encubierto* a aquel que *no* está destinado a la transmisión de información, pero que de algún modo puede utilizarse para una comunicación. En oposición a los canales encubiertos, se encuentran los *canales legítimos* que comprenden a las vías de comunicación diseñadas a tal fin (e.g. la comunicación de información en el marco de lo establecido por el protocolo IP[2] o el

almacenamiento de información en un archivo en formato XML, conforme a su especificación[10]).

El *portador* es todo aquel conjunto de datos que sea susceptible de ser alterado para incorporarle un mensaje que se mantiene secreto, a partir de la aplicación de un algoritmo esteganográfico (denominado *estego-algoritmo*) que indica cómo realizar el procedimiento de incorporación. Este algoritmo puede estar parametrizado por una clave esteganográfica (denominada *estego-clave*) que define cómo aplicar el algoritmo (por ejemplo, la estego-clave podría representar el lugar dentro del portador a partir del cual se comienza a realizar la incorporación del mensaje). Tanto el estego-algoritmo cuanto la estego-clave deben estar previamente acordados entre el emisor y el receptor.

La capacidad de alterar el portador de manera imperceptible es posible a partir de la existencia de redundancia en el mismo. Las alteraciones pueden realizarse tanto en el contenido, cuanto en otros parámetros, como ser, el tiempo de respuesta en la emisión del portador.

La acción de ocultar el mensaje dentro del portador se denomina *embeber*, mientras que la acción de la recuperación posterior del mensaje oculto se denomina *extraer*[7].

Como convención, se emplea el término *mensaje legítimo* para referir a la información o al mensaje transportado por el portador, y el término *mensaje esteganográfico* para el que se embebe en el portador a partir de técnicas esteganográficas. En este contexto, la palabra *legítimo* se utiliza para denotar que la comunicación que se efectúa para transmitir ese mensaje es conocida, es decir, alguien con acceso al canal de comunicación puede tomar conocimiento de la realización del acto de comunicación, a diferencia de lo que ocurre con el mensaje esteganográfico.

Se denomina *protocolo de comunicación* al conjunto de reglas o convenciones que rige el intercambio de bloques de datos entre dos pares[12]. Se denomina *mensaje de protocolo* a cada una de las unidades que se transmite independientemente a través de un canal de comunicación y que está construida sobre la base de las reglas del protocolo de comunicación. Se denomina *esteganografía de protocolo* a la aplicación de técnicas esteganográficas sobre mensajes de protocolos de comunicación utilizados como portadores[5].

El objetivo principal del trabajo consiste en el estudio pormenorizado de los protocolos de comunicación IP y HTTP para encontrar aplicaciones de técnicas esteganográficas sobre los mismos. Se entiende por *técnica esteganográfica sobre un protocolo* a cada una de las interpretaciones o modificaciones que se aplica a un mensaje del protocolo que permita comunicar un mensaje de manera inadvertida, fuera de las reglas del protocolo. A la posibilidad de aplicación de esas técnicas sobre los protocolos se las denomina "*vulnerabilidades esteganográficas*" de los protocolos.

Los protocolos estudiados permiten el reenvío de sus mensajes a través de *intermediarios* que se encuentran en el recorrido entre el emisor y el receptor. El objetivo de encontrar técnicas esteganográficas en los protocolos sustenta el hecho de que un intermediario actúe como emisor esteganográfico a partir de *embeber* el mensaje esteganográfico dentro del mensaje de protocolo que debe retransmitir, para que otro intermediario del protocolo, ubicado más adelante

en la ruta del mensaje, realice la *extracción* del mensaje esteganográfico. Esta extracción puede modificar o no el mensaje portador. En ambos casos, esto se realiza sin alterar la semántica del mensaje del protocolo (i.e. de la comunicación legítima). Estos intermediarios están gobernados por personas que desean comunicarse de manera inadvertida. El interés de realizar este tipo de comunicación consiste en no despertar sospecha a terceras personas acerca de que se está realizando un acto de comunicación. Es decir, no sólo se disimula el mensaje esteganográfico sino también el propio acto de comunicación de ese mensaje.

El resto del trabajo continúa de la siguiente forma: en la sección 2 se presentan las técnicas esteganográficas propuestas sobre el protocolo IP, en la sección 3 sobre el protocolo HTTP. En la sección 4 se presentan las conclusiones del trabajo y las futuras líneas de investigación. Por último, en el Apéndice se presentan los lineamientos de diseño de un artefacto de software construido para mostrar la factibilidad de aplicación de las técnicas propuestas.

2. Análisis esteganográfico del Protocolo IP

En la presente sección se realiza el análisis de las estructuras (i.e. campos de datos) que define la especificación del protocolo IP para la transmisión de mensajes, con el objetivo de encontrar canales encubiertos que puedan ser utilizados para una comunicación esteganográfica. Se analiza el protocolo IP versión 4 a partir de su especificación, la RFC 791[2].

2.1. Generalidades del protocolo IP

IP es un protocolo de capa de red diseñado para ser utilizado en redes interconectadas, su objetivo es la transmisión de bloques de bits, denominados *datagramas*. A los fines del presente trabajo, se utiliza la palabra *subred* para denotar el conjunto de equipos informáticos y software conectados entre sí por medio de dispositivos físicos, que pueden comunicarse datagramas sin necesidad de intermediarios (más allá de los protocolos de capas inferiores). Se utiliza la palabra *red* para denotar de forma general a un conjunto de subredes interconectadas entre sí a través de equipos informáticos denominados *gateways*.

IP provee también la función de fragmentación y posterior reensamblado en destino de los datagramas, cuando éstos deben atravesar subredes que permiten el envío de paquetes de datos más pequeños que la longitud del datagrama.

El protocolo IP *no* provee confiabilidad (i.e. no se asegura la recepción de los mensajes enviados, no se envía confirmación de recepción del destino ni de los enrutadores intermedios), control del flujo de mensajes, retransmisiones, ni secuenciamiento (no se asegura la entrega ordenada de los datagramas en destino, respecto del orden en el cual fueron entregados a la capa inferior en la computadora de origen). No se provee un mecanismo de control de errores sobre los bits transportados, sólo hay una suma de verificación (en inglés *checksum*) sobre los campos del encabezado del datagrama IP.

Cada datagrama es tratado como una unidad *independiente*, es decir, no se almacena información de estado entre diferentes datagramas. En consecuencia,

la comunicación a través del protocolo IP no requiere de fases de conexión ni liberación.

En la figura 1 se muestra el esquema del encabezado del mensaje del datagrama IP sobre la base de lo establecido en la especificación.

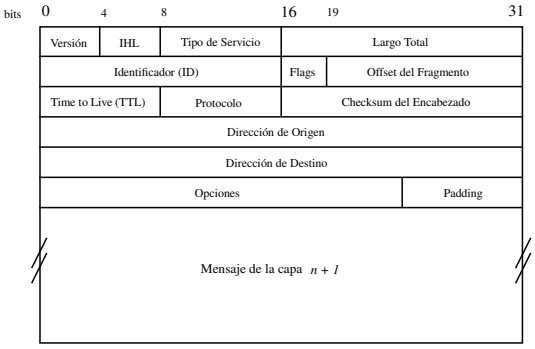

Figura 1: Esquema de encabezado de datagrama IP y el mensaje de la capa superior

2.2. Aplicaciones esteganográficas sobre IP

A continuación se describen sintéticamente los campos que son susceptibles a ser usados en el marco de una comunicación esteganográfica, sin malograr el objetivo de la comunicación real.

Tipo de Servicio Compuesto por 8 bits. Provee una indicación de los parámetros de calidad del servicio deseado. Esos parámetros se utilizan como guía para determinar qué tipo de servicio utilizar para transmitir un datagrama a través de una subred particular.

Este campo está compuesto por los bits con la siguiente semántica:

- Bits 0 a 2: precedencia del datagrama.
- Bit 3: 0 = Demora normal (*delay*, en inglés), 1 = baja demora.
- Bit 4: 0 = Tasa de transferencia normal, 1 = alta tasa de transferencia.
- Bit 5: 0 = Confiabilidad normal, 1 = confiabilidad alta.
- Bits 6 y 7: reservados para uso futuro.

Se observa que los bits reservados pueden ser modificados y utilizados para embeber bits del mensaje esteganográfico. Por otra parte, la alteración de los correspondientes a la indicación de demora, tasa de transferencia y confiabilidad implicarían una variación del costo del servicio, aunque bien podrían ser utilizados para la transmisión de mensajes esteganográficos, con la consideración de que se modificaría, en consecuencia, parte de la semántica del datagrama construido, pero sin alterar el objetivo de la comunicación (i.e. el datagrama llega a destino correctamente). Lo mismo puede realizarse con la modificación de la sección de precedencia (se debe tener en cuenta que, por ejemplo, la modificación de la precedencia de un datagrama con valor crítico, podría traer como consecuencia la demora de una comunicación importante para una emergencia).

Identificador (ID) Compuesto por 16 bits. Contiene un valor asignado por el emisor para permitir la fragmentación y posterior reensamblado de los fragmentos del datagrama. Este campo permite 65,536 valores diferentes (2^{16}), de 0 a 65,535 y su valor es considerado sólo cuando el datagrama es fragmentado.

El campo ID puede contener un valor *arbitrario*, siempre que se asegure que no se repita con el ID de otro datagrama en el lapso en el que el datagrama transita por la red. Esta característica permite efectivamente utilizar el valor del campo ID para realizar una comunicación esteganográfica. De este modo, un algoritmo de inserción de un mensaje esteganográfico, puede modificar el valor del campo ID que posee un datagrama construido por el módulo IP del emisor y asignar en ese campo un valor diferente. Este nuevo valor puede representar un caracter de un alfabeto dentro del mensaje esteganográfico.

Banderas (Flags) Compuesto por 3 bits.

Cada bit tiene el siguiente significado:

- Bit 0: reservado, debe ser cero.
- Bit 1 (*Don't fragment*): `DF` 0 = el datagrama puede ser fragmentado. 1 = no se permite fragmentar el datagrama. La especificación indica que a esta bandera puede asignarse valor 1 (i.e. prohibir la fragmentación del datagrama) en los casos en que el receptor no posea suficientes recursos computacionales para reensamblar el datagrama.
- Bit 2 (*More fragments*): `MF` 0 = último fragmento. 1 = existen más fragmentos.

La especificación no es explícita con respecto a qué acción se debería tomar si el valor del campo reservado, el primer bit de las banderas, no es cero. Se podría suponer, en consecuencia, que este campo puede ser modificado para incorporar un mensaje esteganográfico, sin alterar la semántica del datagrama.

Se propone el siguiente experimento para analizar el comportamiento de la bandera `DF`. Para ello, se captura durante cinco minutos el tráfico de red -se utiliza para ello el software *Wireshark*- de una computadora con el sistema operativo GNU/Linux distribución Kubuntu 10.10 conectada a Internet. Se efectúan actividades de uso de Internet tradicionales.

En dicha captura se observó un total de 11.017 datagramas IP. De ellos, 9.891 datagramas poseen la bandera `DF` con valor 1. Los restantes 1.126 tienen la bandera `DF` con valor 0. La totalidad de los datagramas fue transmitida sin fragmentación, es decir, todos poseen la bandera `MF` con valor 0. No se encuentra, en el análisis de los datagramas obtenidos, un consenso generalizado sobre el uso de la bandera `DF`.

En conclusión, si se decidiera utilizar la bandera `DF` para transmitir un mensaje esteganográfico, a partir de la modificación de dicha bandera en un datagrama ya construido perteneciente a una comunicación genuina, se espera que no se vea alterado el objetivo de la comunicación genuina.

Offset del Fragmento Compuesto por 13 bits. Indica a qué porción del contenido del mensaje del datagrama corresponde el fragmento de mensaje. Contiene la posición de inicio del contenido del datagrama respecto del mensaje total. Esta posición está medida en unidades de 8 octetos (64 bits). El primer fragmento

tiene offset 0.

Si el largo total del datagrama (valor del campo *TL*) es menor o igual al *tamaño máximo de transmisión* (*MTU*) de la subred donde se debe colocar el datagrama, se envía el datagrama completo al siguiente paso en su procesamiento. Si, en cambio, es mayor, se divide el datagrama en dos fragmentos: el primer fragmento del mayor tamaño posible y el segundo del resto. Este procedimiento se repite hasta que el último fragmento obtenido sea menor al MTU.

Del análisis de los procedimientos de fragmentación y reensamblado establecidos por la especificación se propone la aplicación de las siguientes técnicas esteganográficas:

- Supóngase el escenario donde un emisor de un datagrama se encuentra en la misma subred que el gateway que realiza el embebido del mensaje esteganográfico. Este gateway recibe un datagrama originado por el emisor. Dado que cada host conectado a su subred conoce el MTU de la misma, cada emisor generador de datagramas es capaz de colocar en su subred datagramas sin fragmentación.
  Una técnica esteganográfica, podría consistir en colocar en la subred datagramas fragmentados. Donde el tamaño y cantidad de los fragmentos permita comunicar un mensaje esteganográfico.
- En el caso que el gateway que embebe el mensaje esteganográfico se encuentra situado en un nodo entre dos subredes si, además, dadas las características de la segunda subred, el datagrama necesariamente debe ser fragmentado para ser colocado en esa subred, se puede aplicar la siguiente propuesta esteganográfica: El gateway que embebe el mensaje esteganográfico puede generar fragmentos de menor tamaño, y de diferentes tamaños entre sí. El tamaño de los fragmentos representa el mensaje esteganográfico para el gateway de extracción.
- Adicionalmente, la especificación aclara que si se reciben bits *solapados* (es decir, se envían porciones del fragmento original en dos fragmentos) se debe utilizar la copia más recientemente recibida de los datos.
  En este caso, también se propone una aplicación esteganográfica: al momento den generar fragmentos, se puede generar los mismos solapados, el número de bits solapados corresponde a un mensaje esteganográfico.

Tiempo de Vida (TTL) Compuesto por 8 bits. El tiempo de vida indica una cota máxima al datagrama en segundos y en cantidad de saltos de procesamiento (lo que ocurra primero).

Se han realizado pruebas sobre la modificación del campo TTL y se propone como aplicación esteganográfica modificar el valor de este campo para asignar un valor *alto* o *bajo* (i.e. mayor o menor al máximo valor), de acuerdo se quiera transmitir un bit 1 ó 0 de un mensaje esteganográfico.

Opciones Tiene longitud variable, con un máximo de 40 bytes. Puede aparecer en el datagrama o no. Las Opciones incorporan parámetros que actúan sobre el enrutamiento del datagrama. Es obligatorio que el tamaño de este campo sea múltiplo de 4 bytes.

Entre las opciones que pueden ser utilizadas para una comunicación esteganográfica se encuentran:

- *No Operación* y *Fin de la lista de Opciones*: Ambas se utilizan para mantener el encolumnado a 4 bytes. La diferencia entre ellas es que la segunda sólo puede ser usada como última opción.
  Se puede agregar estas opciones en casos en los que no sea necesario (por ejemplo, tres opciones *No Operación* y una *Fin de la lista de Opciones*). La cantidad y tipo de opciones representa un mensaje esteganográfico.
- Opciones para almacenar la ruta del datagrama.
  Cabe mencionar que el uso de estas opciones es actualmente obsoleto, porque la cantidad de gateways que atraviesa un datagrama actualmente en Internet no permite almacenar todas las direcciones en el tamaño previsto para este campo[13]:
  - *Loose Source and Record Route* y *Strict Source and Record Route*: Permiten al emisor indicar el camino que debe realizar el datagrama. La primera permite que se incorporen gateways intermedios, la segunda no.
  - *Record Route* e *Internet Timestamp*: Permiten almacenar las direcciones de los gateways que atraviesa el datagrama.

  El uso esteganográfico propuesto para las mismas consiste en colocar una ruta *ficticia* configurada como si el datagrama ya hubiera pasado por esas direcciones (para el primer caso) o como si fuera las direcciones de los gateways que lo procesaron (para las opciones del segundo caso). El mensaje esteganográfico está representado por la ruta colocada.
  Asimismo, cabe aclarar que el uso de este tipo de opción puede verse afectado en la comunicación del datagrama, por cuestiones de seguridad. Un *firewall* puede quitar esta opción para no revelar la ruta, que puede ser considerada confidencial.

3. Análisis esteganográfico del Protocolo HTTP

El protocolo HTTP (*Hypertext Transfer Protocol*) es el estándar para la transferencia en la Web[13]. La versión que se analiza en este trabajo es HTTP/1.1, que constituye la versión vigente, definida en la RFC 2616[9].

3.1. Generalidades del protocolo HTTP

HTTP es un protocolo de capa de aplicación para la distribución y comunicación. Es un protocolo *genérico* y *sin estado*, que puede ser empleado para varias tareas, además de la comunicación de hipertexto en lenguaje HTML (*HyperText Markup Language*).

HTTP es un protocolo de tipo *cliente-servidor* que opera con mensajes pedido/respuesta (*request/reply*). El *cliente* es el programa que establece conexiones con el objetivo de enviar pedidos (*requests*). Se lo denomina *agente de usuario* (o *user agent*, en inglés). El *servidor* es un programa que acepta conexiones entrantes para responder a los pedidos (*requests*), con el envío de respuestas (*replies*).

Un mismo software, o equipo de la red, puede actuar simultáneamente como cliente y servidor.

HTTP provee encabezados (*headers*) para enviar el pedido, con *métodos* para indicar el tipo de pedido, y define a la ubicación del *recurso* referido a partir de su URI (*Uniform Resource Identifier*). El objetivo principal del protocolo es permitir al cliente solicitar la ejecución acciones (a partir de *métodos*) sobre elementos (*recursos*) que se encuentran en el servidor. Los mensajes son transmitidos en formato tipo MIME (*Multipurpose Internet Mail Extensions*).

La comunicación entre el cliente y el servidor puede llevarse a cabo a través de *intermediarios* que reenvían los mensajes de pedido y respuesta. Un *proxy* es un agente intermediario que reenvía los mensajes, recibe los pedidos para una URI, reescribe todo o parte del mensaje y lo reenvía al servidor correspondiente, o a otro intermediario.

El estudio de este protocolo permite proponer técnicas esteganográficas que puedan ser implementadas en dos intermediarios distintos dentro de la cadena de reenvíos de mensajes. Es decir, un proxy realizará la operación de embebido y otro la de extracción esteganográfica. En todos los casos, se preservará la semántica del mensaje HTTP original. Es decir, el servidor recibirá la versión modificada (de forma imperceptible) por el proxy de embebido esteganográfico.

Ambos tipos de mensajes HTTP (*pedido* o *respuesta*) consisten en una línea inicial, cero o más campos de encabezado (conocidos como *headers*), luego una línea en blanco (donde sólo se coloca la marca de fin de línea `CRLF`) que indica el fin de los campos del encabezado y, opcionalmente, el cuerpo del mensaje.

3.2. Aplicaciones esteganográficas sobre HTTP

Los mensajes de pedido contienen la primera línea del encabezado con el tipo de pedido (i.e. el método), conocida como *Request-line*, la versión del protocolo (HTTP/1.1) y la URI del recurso referido, y a continuación líneas con: los modificadores del pedido (*entity-headers*), información del cliente y, opcionalmente, un cuerpo de pedido (*entity-body*). Como ejemplo, se tiene el siguiente mensaje de pedido, que solicita el recurso raíz ("`/`") ubicado en el servidor de nombre *www.fi.uba.ar*:

```
GET / HTTP/1.1
Host: www.fi.uba.ar
User-Agent: Mozilla/5.0 Gecko/20100101 Firefox/12.0
Accept: text/html,application/xhtml+xml,*/*;q=0.8
Accept-Language: fr-fr,es-ar;q=0.8,en-us;q=0.5,en;q=0.3
```

Los mensajes de respuesta contienen un encabezado con una línea de estado, que contiene un código de error o éxito y su descripción en forma de texto, seguido de un mensaje en formato tipo MIME que contiene información del servidor, metainformación del recurso y, opcionalmente, un contenido como parte de cuerpo del mensaje (denominado *content* o *entity-body* en la especificación). Se ilustra con el siguiente mensaje de respuesta:

```
HTTP/1.1 200 OK
```

```
Date: Wed, 16 May 2012 03:38:07 GMT
Server: Apache/2.2.9 (Debian) PHP/5.2.6-1+lenny13 mod_ssl/2.2.9
Content-Length: 6355
Content-Type: text/html; charset=iso-8859-1

... cuerpo del recurso ...
```

Por claridad, se omite el cuerpo del recurso en la respuesta.

En orden de lograr mayor robustez, la especificación establece que los servidores deben ignorar cualquier línea en blanco que preceda a la línea de pedido (*Request-Line*) esperada. Si el servidor espera leer la primera línea de un mensaje de pedido y en su lugar encuentra una secuencia `CRLR`, debe ignorarla.

Esto permite la utilización para una comunicación esteganográfica. Puesto que la primera línea en blanco de un mensaje de pedido es ignorada por el servidor, se puede incorporar a un mensaje construido una línea en blanco que otorgue una interpretación esteganográfica para los intermediarios que realizan la comunicación esteganográfica.

Los campos se especifican: "*nombre del campo*`:`*valor*". Los nombres de los campos son insensibles a mayúsculas (*case-insensitive*). Por lo cual, es posible utilizar esta característica para una comunicación esteganográfica, donde el mensaje esteganográfico surge a partir de la interpretación de las mayúsculas y minúsculas utilizadas. Esto permite, por ejemplo, embeber un bit del mensaje esteganográfico por cada letra del nombre del campo.

Los valores de los campos del mensaje HTTP pueden ser separados en varias líneas (técnica que se denomina *folding*). Para indicar que el contenido del campo fue cortado en varias líneas, la línea que continúa debe comenzar con un caracter espacio o con un caracter tabulador horizontal (que corresponde al caracter de valor 09 en codificación ASCII). Esta marca de separación se denomina `LWS`. De forma general, la especificación aclara que puede haber más de una marca `LWS` consecutivas en la separación del contenido de un campo en varias líneas y el receptor puede reemplazar cualquier marca `LWS` por un caracter espacio (denominado `SP`, que corresponde al valor ASCII 32). Cuando un emisor genera un mensaje, debe esperar que el caracter `LWS` pueda ser reemplazado por un `SP` en la interpretación del texto.

Es característica que concede la especificación para el contenido de los campos del mensaje, permite que sea explotada para realizar una comunicación esteganográfica. El hecho de realizar folding en el campo de un mensaje puede ser interpretado como un mensaje esteganográfico. Por ejemplo, se puede codificar un número en la cantidad de veces que sea realiza folding sobre un mismo campo del mensaje. Esto, siempre que el contenido del mensaje no se vea alterado con el reemplazo posterior de la marca `LWS` por un espacio `SP` (i.e. si el contenido del campo no posee espacios, se alteraría el texto original, o genuino). Por otra parte, se puede utilizar la posibilidad de colocar consecutivamente varias marcas `LWS`, dado que todas ellas son reemplazadas por un único espacio al momento de interpretar el contenido del campo en el módulo HTTP. En este sentido, la cantidad de marcas `LWS` consecutivas constituye un mensaje esteganográfico.

Adicionalmente, la especificación establece que el contenido del campo puede estar precedido por cualquier cantidad de secuencias `LWS`, aunque se prefiere un

sólo caracter SP. Esto puede ser utilizado de la misma forma descripta para el caso del folding.

Algunos campos del encabezado permiten contener comentarios. Éstos se colocan rodeados por paréntesis. En los campos en los que no se permiten comentarios, los paréntesis se interpretan como parte del texto.

Los comentarios dentro de los campos del encabezado del mensaje también pueden ser utilizados para realizar una comunicación esteganográfica. El texto colocado dentro del comentario puede ser destinado al receptor esteganográfico.

La especificación aclara que el orden en el que son colocados los campos del encabezado no es relevante. Aunque establece que es una "buena práctica" colocar en primer lugar los campos generales, seguido por los campos de pedido o respuesta y por último los campos de entidad.

La posibilidad de envío de los campos del encabezado en diferente orden puede ser también utilizada para la interpretación de un mensaje esteganográfico. Dado que es posible realizar una comparación entre el orden de los campos recibidos en un portador con respecto a un ordenamiento preestablecido que tendrían los mismos (e.g. al orden alfabético), es posible, por ejemplo, cuantificar la cantidad de posiciones que hay que mover a cada campo del mensaje recibido para que se ubique en su posición de acuerdo a ese ordenamiento (e.g. alfabéticamente).

La especificación establece que es posible colocar varias veces el mismo campo del encabezado, con el mismo nombre si y sólo si el contenido de ese campo es una lista separada por comas, y es posible combinar todos los campos en uno sólo de la forma "*nombre del campo: valor*", donde el *valor* es la concatenación de cada uno de los valores separados por una coma. El orden en el que se realiza la concatenación es el orden en el que se reciben los campos separados dentro del encabezado. En el caso de la existencia de múltiples campos con el mismo nombre, un proxy no debe modificar el orden de ellos.

Se puede realizar otra aplicación esteganográfica a partir del empleo de la posibilidad de escribir la lista de valores separados por comas en un sólo campo o en múltiples campos. Esto permite la interpretación de un mensaje esteganográfico de un bit (el caso del contenido del campo escrito en un sólo campo corresponde a un valor del bit y el de varios campos con el mismo nombre para escribir el contenido corresponde al otro valor del bit).

La especificación establece que se puede extender los campos encabezado sin modificar el protocolo, pero que no debe asumirse que el receptor del mensaje interprete el encabezado extendido. Los encabezados no reconocidos deben ser ignorados por el receptor, pero deben ser reenviados por los proxys transparentes (i.e. que no modifican el mensaje que reenvían). La extensión de estos campos puede permitir realizar comunicaciones esteganográficas (i.e. a partir del agregado de campos *propietarios* con un significado dentro de la comunicación esteganográfica).

Versión HTTP El protocolo utiliza un esquema de numeración de la versión del protocolo con la forma: "*mayor*" . "*menor*". La versión define el formato del

mensaje. El formato de la versión es el siguiente:
`"HTTP" "/" 1*DIGIT "." 1*DIGIT`
(i.e. ambos campos pueden estar compuestos por uno o más dígitos). En el caso del protocolo en estudio, corresponde a `HTTP/1.1`. Los valores *mayor* y *menor* que componen la versión del protocolo pueden ser incrementados por separado.

La especificación aclara que los ceros al inicio de cada uno de los valores que componen la versión deben ser ignorados por el receptor y que no deben ser enviados. Esta aclaración permite utilizar esta característica como una aplicación esteganográfica. El envío de uno o más ceros al comienzo de cada uno de los valores puede codificar un mensaje.

Uniform Resource Identifiers (URI) La URI (Identificación Uniforme del Recurso) consiste en una cadena de caracteres que identifica a partir del nombre, ubicación u otra característica a un recurso.

La URI puede ser representada en forma absoluta o en forma relativa a partir de otra URI base conocida, según el contexto en el que se use. Las URIs absolutas comienzan con el nombre del esquema seguido del caracter *dos puntos*.

Para el protocolo HTTP se utiliza el esquema *http*, con la siguiente sintaxis:
`"http:" "//" host [ ":" port ] [ abs_path [ "?" query ]]`

El valor del puerto (*port*) es opcional y, si no se especifica, se asume 80. Esto permite una implementación esteganográfica en un proxy transparente, que agregue el puerto 80 (se agrega `:80` luego del nombre del host) en los casos en los que el puerto no se coloque. El mensaje esteganográfico radica en la interpretación de la presencia del valor 80 o no, cuando se trata del puerto estándar.

El valor de *abs_path* representa la ubicación absoluta del recurso. Si no se coloca *abs_path*, se asume "`/`". Esto permite una aplicación esteganográfica: cuando el valor de *abs_path* es nulo, puede elegirse colocar el caracter "`/`" o no hacerlo.

Los caracteres que no están identificados como *reservados* o *inseguros* son equivalentes a su codificación en secuencias de escape. Una *secuencia de escape*, como `%7E`, proporciona un mecanismo general para representar caracteres invisibles o difíciles de escribir[3] (en este caso, el caracter "~"). La secuencia de escape se forma con el caracter "`%`" seguido de dos dígitos correspondientes a la codificación ISO-8859-1 en base hexadecimal del caracter. Los caracteres no numéricos de la escritura hexadecimal pueden colocarse tanto en mayúsculas cuanto en minúsculas. En las implementaciones es común escribir la forma "`%7e`" que corresponde al caracter "~".

La especificación ejemplifica la equivalencia de URIs con las siguientes:

```
http://abc.com:80/~smith/home.html
http://ABC.com/%7Esmith/home.html
http://ABC.com:/%7esmith/home.html
```

Se propone aplicar la propiedad de indiferencia a mayúsculas de la comparación de nombres de host y nombres de esquema para el uso esteganográfico. Se puede realizar una comunicación esteganográfica a partir de la interpretación de mayúsculas o minúsculas de cada caracter de estas dos partes de la URI.

Por otra parte, dada la equivalencia de caracteres con su representación en secuencias de escape, se tiene otra posibilidad de aplicación de esteganografía a partir de la modificación de la escritura de la URI. Una función que embeba un mensaje esteganográfico puede reemplazar (e.g. aleatoriamente) caracteres no reservados, ni inseguros por su versión en secuencia de escape. Por ejemplo, se puede reemplazar letras del abecedario por su correspondiente secuencia de escape. El mensaje esteganográfico estaría representado, por ejemplo, por la cantidad de letras del abecedario escritas como secuencia de escape. Nótese, además, que el reemplazo en secuencias de caracteres es insensible a mayúsculas (e.g. "`%7e`" es equivalente a "`%7E`"), por lo que también puede usarse para la interpretación de un mensaje esteganográfico.

En los casos propuestos de modificación de la URI, la semántica de la misma permanece inalterada, es decir, la versión modificada y la no modificada de la URI son equivalentes (i.e. refieren al mismo recurso).

Formato de Fecha y Hora Las marcas de fecha y hora se representan en *Greenwich Mean Time* (GMT), sin excepción.

El protocolo permite tres formas diferentes de representar las marcas de fecha y hora. Se muestra a continuación el ejemplo enunciado en la especificación:

```
Sun, 06 Nov 1994 08:49:37 GMT  ; RFC 822, updated by RFC 1123
Sunday, 06-Nov-94 08:49:37 GMT ; RFC 850, obsoleted by RFC 1036
Sun Nov  6 08:49:37 1994       ; ANSI C's asctime() format
```

Dado que existen tres formatos para la escritura de marcas de fecha y hora que deben interpretar los servidores, se puede utilizar para una comunicación esteganográfica, sólo en el sentido del cliente hacia el servidor, porque el servidor debe emitir las marcas de tiempo en uno sólo de los formatos. De esta forma, la escritura de las marcas de tiempo en cada uno de los formatos puede tener una interpretación dentro de un mensaje esteganográfico para los intermediarios involucrados.

Conjuntos de caracteres El conjunto de caracteres refiere al método utilizado junto con tablas de conversión de secuencias de octetos en secuencias de caracteres. No todos los conjuntos de caracteres pueden representar todos los caracteres.

Los conjuntos de caracteres de HTTP son definidos por nombres en forma insensible a mayúsculas (lo que permite la aplicación de las técnicas esteganográficas ya descriptas), cualquier valor es permitido, y se recomienda a las aplicaciones utilizar sólo los definidos en *IANA Character Set registry* (*http://www.faqs.org/rfcs/rfc1700.html*).

Dado que es posible codificar el encabezado en diferentes conjuntos de caracteres, se puede hacer uso de esta libertad para realizar una comunicación esteganográfica donde el mensaje correspondiente surja a partir de la interpretación del conjunto de caracteres utilizado. Una función de aplicación esteganográfica puede recodificar un mensaje HTTP a otro conjunto de caracteres, siempre que

esto sea posible (por ejemplo, en ISO-8859-1 puede representarse todos las letras del idioma español, las vocales con sus tildes, en mayúsculas y minúsculas, mientras que en ISO-8859-2 no puede representarse la letra "ñ").

Valores de Calidad HTTP utiliza parámetros de calidad para indicar la importancia relativa (o *peso*) de diferentes opciones en la negociación de parámetros. El valor de peso es normalizado como valor real entre 0 y 1, donde 0 representa el mínimo y 1 el máximo. Si un parámetro tiene un valor de calidad igual a 0, indica que ese valor del campo es *inaceptable* para el cliente. El valor de calidad puede tener tres dígitos decimales como máximo.

Los ceros finales son opcionales. Esto permite aplicarlo en el marco de una comunicación esteganográfica, donde la cantidad de ceros innecesarios que son explícitamente colocados sea interpretada como un mensaje. Por ejemplo, puede colocarse `1` ó `1.000` con equivalencia en la comunicación genuina. Adicionalmente, se puede modificar de forma imperceptible los valores de calidad, para otorgar un significado dentro de la comunicación esteganográfica. Por ejemplo `0.699` puede considerarse equivalente a `0.7` dentro de la comunicación genuina.

4. Conclusiones y futuras líneas de investigación

4.1. Conclusiones

Se ha analizado los protocolos de comunicación IP y HTTP desde el punto de vista de su capacidad para transmitir información a partir de la aplicación de esteganografía, es decir, a través de reglas no definidas en la especificación del protocolo que permitan realizar una comunicación *disimulada*. A partir del estudio pormenorizado de la especificación de los protocolos, se propusieron técnicas que aprovechan cada una de las vulnerabilidades esteganográficas encontradas. Las modificaciones al mensaje portador se han sustentado en que las mismas no alteran la comunicación genuina sobre el protocolo.

Se ha desarrollado un artefacto de comunicación esteganográfica que implementa algunas de las técnicas analizadas sobre ambos protocolos. Análogamente, se ha desarrollado un guardián activo (esto es, un intermediario que modifica el contenido de los mensajes que reenvía, para evitar las comunicaciones esteganográficas). El objetivo del guardián activo desarrollado consiste en evitar la transmisión de los mensajes esteganográficos con las técnicas implementadas. En el Apéndice se comentan las pruebas realizadas. La construcción del artefacto permitió comprobar la factibilidad de la aplicación de las técnicas implementadas para la comunicación esteganográfica. Al mismo tiempo, la construcción del guardián activo permitió contrarrestar la aplicación de las técnicas implementadas. Se pudo observar que, en todas las pruebas, las comunicaciones basadas en los protocolos funcionaron correctamente.

A lo largo del análisis de los protocolos de comunicación se ha notado que las vulnerabilidades esteganográficas presentes en ellos responden principalmente a ambigüedades, vaguedades o libertades en las especificaciones de los mismos. Esto es, puntos en los que la definición del protocolo no es estricta, sino que, en algunos casos, deja a criterio de las implementaciones concretas la acción

a tomar bajo ciertos valores en algunos campos de los mensajes, o permite el agregado de campos que en ciertos casos no poseen un objetivo real.

En general, la posibilidad de existencia de *canales encubiertos* no puede ser completamente eliminada[11], aunque puede ser reducida a partir del consideraciones de diseño del protocolo de comunicación.

Por último, el desarrollo del presente trabajo permitió cumplir con un importante objetivo propuesto: la formación de un *espíritu crítico* al estudiar las especificaciones de protocolos. Esto se logró a partir de buscar otras formas de utilizar los mecanismos y estructuras que proveen los protocolos, diferentes a las que se establecen en sus especificaciones. A partir de este tipo de análisis es posible proponer mejoras a los protocolos, anticipar posibles problemas o vulnerabilidades (en el caso del presente trabajo, violar políticas de seguridad a partir del filtrado inadvertido de información).

4.2. Futuras líneas de investigación

Como futuras líneas de investigación han surgido los siguientes temas a lo largo del desarrollo del trabajo:

- En el presente trabajo se ha analizado el protocolo IP de forma *aislada*, es decir, con independencia del protocolo utilizado en sus capas superior e inferior. Se podría analizar la posibilidad de aplicar técnicas esteganográficas sobre el protocolo IP con consideraciones adicionales sobre el protocolo existente en la capa superior, o en protocolos de varias capas superiores.
- Analizar la factibilidad de aplicar sobre *IPv6* las técnicas esteganográficas propuestas para el protocolo IP estudiado.
- Elaborar definiciones de los protocolos estudiados que minimicen las vulnerabilidades esteganográficas.
  Por ejemplo, en el protocolo IP *modificado* se podría especificar que si el datagrama no debe ser fragmentado (bandera *Don't fragment* con valor 1) o no tiene fragmentos, el encabezado del protocolo *no* debe llevar campo de identificación (ID). Puesto que este campo se utiliza sólo a los fines de la fragmentación.
- Analizar la posibilidad de implementar funcionalidades adicionales sobre el artefacto desarrollado. Estas funcionalidades permitirían cumplir objetivos de seguridad sobre la comunicación esteganográfica, tales como:
  - *Autenticación*: para que el receptor esteganográfico pueda tener certeza de que su par emisor es quién él espera que sea.
  - *Cifrado*: para evitar que un tercero que acceda al mensaje esteganográfico pueda interpretarlo[8].
  - *Integridad*: para evitar el agregado, eliminación o modificación de mensajes esteganográficos.
  - *Protección frente a repeticiones del mensaje*: para proteger el esquema de comunicación esteganográfica frente al ataque de un tercero que envíe portadores que haya almacenado anteriormente.

  Esto implica, además, analizar el impacto de estos mecanismos en el tamaño de los mensajes y en la capacidad esteganográfica efectiva.
- Analizar la demora introducida en la comunicación debido a la aplicación de las técnicas esteganográficas implementadas en el artefacto.

- Analizar la aplicación de técnicas de Inteligencia Artificial para la detección de actividades esteganográficas sobre los protocolos estudiados o genéricos: se podría estudiar la posibilidad de desarrollar *guardianes activos genéricos*. Esto es, algoritmos que sin un conocimiento de la estructura del protocolo de comunicación utilizado como portador, puedan encontrar patrones de comportamiento y emitir alertas cuando consideran que se aplican técnicas esteganográficas sobre los mensajes.
- Por último, se podría diseñar un plugin para el software analizador de tráfico utilizado (*Wireshark*) que interprete las comunicaciones esteganográficas implementadas en el artefacto para los protocolos IP y HTTP.

Referencias

[1] Ingemar J. Cox, Matthew L. Miller, Jeffrey A. Bloom, Jessica Fridrich, and Ton Kalker. *Digital Watermarking and Steganography*. Morgan Kaufmann Publishers, edition edition, 2008.

[2] Defense Advanced Research Projects Agency. RFC 791 - Internet Protocol, September 1981.

[3] Brian W. Kernighan and Dennis M. Ritchie. *El lenguaje de Programación C*. Pearson Educación, segunda edition, 1991.

[4] Butler W. Lampson. A Note on the Confinement Problem. *Xerox Palo Alto Research Center*, 1973.

[5] James Pease Steve J. Chapin Norka B. Lucena, Douglas F. Calvert. Semantics-Preserving Application-Layer Protocol Steganography. *Center for Systems Assurance, Syracuse University, Syracuse University, 111 College Place 3-114, Syracuse, NY 13244*.

[6] Fabien A. P. Petitcolas, Ross J. Anderson, and Markus G. Kuhn. Information Hiding - A Survey. *Proceedings of the IEEE, special issue on protection of multimedia content*, July 1999.

[7] Birgit Pftizmann. Information Hiding Terminology - Results of an informal plenary meeting and additional proposals -. *Proc.Information Hiding Workshop*, UK, 30.5.1.6,LNCS, Springer-Verlag, 1996.

[8] Piyush Marwaha, Paresh Marwaha. Visual cryptographic steganography in images. *2010 Second International conference on Computing, Communication and Networking Technologies*, 2010.

[9] R. Fielding and UC Irvine and J. Gettys and J. Mogul and H. Frystyk and L. Masinter and P. Leach and T. Berners-Lee. RFC 2616 - Hypertext Transfer Protocol – HTTP/1.1uni, June 1999.

[10] W3C Recommendation. Extensible Markup Language (XML) 1.0 (Fifth Edition), November, 26 2008.

[11] CISSP Robert C. Newman. Covert Computer and Network Communications. 2007.

[12] William Stallings. *Comunicaciones y Redes de Computadoras*. Séptima edition, 2004.

[13] Andrew S. Tanenbaum. *Redes de Computadoras*. Prentice-Hall Hispanoamericana S.A., tercera edition, 1997.

Apéndice A. Diseño de un artefacto

En este apéndice se presenta el diseño de un artefacto de software que muestre por un lado la factibilidad de la comunicación de mensajes esteganográficos de tipo texto sobre los protocolos estudiados y, por otra parte, la forma de evitar estas técnicas a partir de una guardián activo.

Esto se ha desarrollado para cada uno de los dos protocolos estudiados en forma independiente. Es decir, se implementa un artefacto que cumple los roles de escritor, lector y guardián esteganográfico sobre los mensajes del protocolo IP, y otro artefacto que cumple los mismos roles sobre los mensajes del protocolo HTTP.

El artefacto de software está diseñado para funcionar en la computadora que actúa como gateway entre la red interna de una organización y su conexión a Internet. El rol de *guardián* del artefacto actúa como guardián esteganográfico activo entre los participantes de la comunicación genuina a través de los protocolos estudiados. Preserva la semántica de los mensajes de protocolo para no malograr la comunicación genuina a través del protocolo.

Cabe destacar que la totalidad de los componentes desarrollados para el artefacto de software se encuentran a disposición para ser consultados (esto es extensivo tanto al código fuente cuanto a los archivos ejecutables generados y su documentación de uso).

El artefacto desarrollado sobre el protocolo IP está diseñado para funcionar en un nodo de la red que actúe como enrutador en el camino de la comunicación (éste puede ser un equipo informático que vincule la red de una organización con Internet). Análogamente, el artefacto desarrollado sobre el protocolo HTTP está diseñado para funcionar como proxy HTTP en el camino entre el cliente y el servidor HTTP (puede ser utilizado como un proxy de *uso obligatorio* dentro de una organización). En consecuencia, se puede tomar como consideración que todas las comunicaciones informáticas, a través de los protocolos estudiados, entre la organización y el exterior de la misma deben transitar por la computadora que ejecuta el artefacto.

Apéndice A.1. Consideraciones generales de diseño

El artefacto de software construido fue desarrollado en el lenguaje de programación C++. Se ha utilizado como plataforma para el desarrollo, el sistema operativo GNU/Linux Kubuntu versión 11.10. Para compilar el código fuente se ha utilizado el compilador GCC versión 4.6.1.

En ambas versiones del artefacto (para cada uno de los protocolos), se ha implementado la comunicación esteganográfica de forma unidireccional (*half-duplex*). Es decir, una vez que se ejecuta en una computadora el artefacto con el rol de *escritor* esteganográfico, el mismo realizará las modificaciones a los mensajes portadores para aplicar el mensaje, sin realizar lectura esteganográfica. Lo mismo ocurre con el rol del lector esteganográfico: se desempeña como lector del mensaje esteganográfico durante toda la ejecución del artefacto. Esta decisión no implica una pérdida de generalidad ni disminución de los objetivos propuestos (mostrar la aplicación de las técnicas sobre los protocolos), respecto de una

implementación que permita la comunicación esteganográfica en forma bidireccional (*full-duplex*). La diferencia consistiría en que en la versión full-duplex, los participantes alteran sus roles de emisor y receptor esteganográfico: cada vez que reciben un mensaje del protocolo deben aplicarle el algoritmo de lectura esteganográfica y cada vez que envían un mensaje, deben aplicarle el algoritmo de escritura esteganográfica

Se implementó la comunicación de caracteres con un alfabeto de 32 elementos. Cada elemento se representa por 5 bits ($2^5 = 32$). Los caracteres del alfabeto son (en orden): las letras en mayúsculas de la A a la Z (26 caracteres), el punto, la coma, el caracter espacio, el signo de interrogación de cierre, el salto de línea y un símbolo para representar el fin del mensaje (e.g. la letra `A` está representada por 00000, mientras que el caracter de fin de mensaje está representado por 11111).

El conjunto de caracteres escogido permite comunicar mensajes de tipo texto en español (excepto la representación de la letra "*ñ*"), mientras que minimiza la cantidad de bits necesarios para representar cada caracter. Por ejemplo, el alfabeto ASCII representa a cada caracter con 7 bits, mientras que el ASCII extendido requiere de 8 bits por caracter.

Apéndice A.2. Artefacto sobre el protocolo IP

Para la manipulación de los datagramas IP (i.e. lectura y modificación) que son enviados a través de las operaciones de comunicación del sistema operativo, dentro del ámbito del artefacto desarrollado, se decidió utilizar la biblioteca *libnetfilter_queue* (*http://www.netfilter.org/projects/libnetfilter_queue/*) del proyecto *netfilter.org*. Esta biblioteca provee una interfaz (*API*) que permite acceder en espacio de usuario (en inglés *userspace*) a los datagramas que se encuentran en la cola de procesamiento del kernel del sistema operativo. Permite modificar el contenido de los datagramas y reinyectarlos en el procesamiento del kernel, y también permite rechazar el procesamiento del datagrama. Es decir, permite descartar datagramas.

Esta versión del artefacto se desarrolla en un único proceso del sistema operativo, es decir, es *monoproceso*. Dado que IP es un protocolo sin conexión, la recepción de cada datagrama es independiente. Cuando ocurre la recepción de un datagrama, es invocada la ejecución de una función que provee la biblioteca *libnetfilter_queue*. Esta función es de tipo *callback* (*retrollamada*), y realiza la invocación a los métodos que implementan los algoritmos esteganográficos (lector, escritor y guardián). La administración del mensaje esteganográfico (tanto para su envío, cuanto para su recepción) se sustenta sobre una clase que implementa el patrón *Singleton*.

La implementación interna del encabezado del datagrama está sustentada en un atributo de tipo *struct* del lenguaje, con cada uno de los campos del datagrama. De este modo, la representación en memoria de este atributo, visto como una cadena de bits, coincide con la cadena de bits correspondientes al datagrama que se comunica por la red. Cabe comentar que la forma en la que se ordenan los *bytes* (o *palabras*) que componen cada uno de los campos del datagrama puede ser diferente a la forma en la que se ordenan en la memoria de la computadora. Esta característica se denomina *endianness*. En las comunicaciones de red se almacena el byte más significativo en la posición de memoria

de menor dirección (esta forma se denomina *big-endian*), mientras que en la computadora de arquitectura Intel, como la utilizada para el desarrollo, el byte más significativo se coloca en la posición de memoria de mayor dirección (esta forma se denomina *little-endian*)[13]. El sistema operativo provee funciones para cambiar la representación de los números entre una forma y otra de almacenarlo en memoria.

El método implementado para el embebido del mensaje esteganográfico actúa sólo sobre datagramas IP de versión 4. Obtiene bits del mensaje esteganográfico a partir de la única instancia de la clase singleton y realiza las siguientes operaciones sobre el datagrama:

- Modifica el campo ID. Incorpora dos bits del mensaje sobre este campo. De los 16 bits que componen a este campo, modifica el bit de la posición 0 y el de la posición 8. En cada caso coloca un bit leído del mensaje.
- Modifica el campo TTL para incorporar un bit del mensaje esteganográfico. La interpretación esteganográfica que se realiza sobre este campo es de dos estados: es un valor *alto* (mayor a 120), o es un valor *bajo* (menor o igual a 120), como se mencionó en la explicación de este campo.
  Si el bit del mensaje esteganográfico que se debe colocar es un 1, se asigna en este campo el valor 240 (que representa un valor alto), de modo que al llegar el datagrama al receptor y ser este campo decrementado en cada nodo de procesamiento permanezca en un valor *alto*. Si el bit del mensaje esteganográfico que se debe colocar es un 0, se asigna en este campo el valor 120 que representa un valor *bajo* (si el valor de este campo previamente es menor a 120, no se modifica).
- Modifica el valor de la bandera reservada (corresponde al bit de la posición 0 del campo *Banderas*). Coloca en este campo un bit extraído del mensaje esteganográfico.
- Modifica la bandera `DF`, que indica el permiso de fragmentar el datagrama. Realiza esta operación sólo si el datagrama no está fragmentado, es decir, si la bandera `MF` y el campo *Offset* tienen valor 0. En este campo coloca el valor del bit leído del mensaje esteganográfico.
- El campo *Opciones* es utilizado para la comunicación esteganográfica de forma parametrizable al lanzar el ejecutable. Se implementó la interpretación de un parámetro de configuración que indica usar alta o baja capacidad esteganográfica. Este parámetro puede ser visto como una clave esteganográfica, de tipo binaria, que modifica el comportamiento de los algoritmos de embebido y extracción.
  Si se elige "alta", se agrega la opción *Record Route* al campo de Opciones del datagrama. Esta operación se realiza sólo si el campo Opciones se encuentra vacío. Si se indica aplicar baja capacidad esteganográfica, no se realiza operación sobre el campo Opciones. Se puede notar que el receptor esteganográfico debe estar configurado de forma equivalente, para interpretar correctamente el mensaje.
  Esta aplicación es la mencionada en la descripción de este tipo de opción: se coloca un ruta inicialmente asignada, como si el datagrama ya hubiera transitado por esos nodos. De esta forma, es posible almacenar hasta 9 direcciones IP, cada una de 32 bits. Las direcciones IP que se colocan

corresponden a 32 bits que se extraen del mensaje esteganográfico. Es por esto que las mismas pueden ser direcciones inexistentes o inválidas en el ámbito de la subred o conjunto de subredes que atraviesa el datagrama. Si en el mensaje esteganográfico hay menos de 32 bits, se completa con bits 0 las posiciones remanentes.

De este modo, este campo permite colocar hasta $9 \cdot 32 = 288$ bits del mensaje esteganográfico.

El lector esteganográfico realiza las operaciones de lectura análogas a las modificaciones introducidas por el emisor, para así obtener el mensaje esteganográfico.

Se implementó una versión que funciona como guardián activo esteganográfico. Éste realiza las modificaciones mencionadas para incorporar valores aleatorios, que impidan la correcta interpretación del mensaje esteganográfico por parte del receptor.

Se realizaron pruebas con la configuración de alta y baja capacidad esteganográfica y se pudo comprobar que los mensajes de la comunicación genuina arribaron a destino y fueron interpretados correctamente por el receptor.

Adicionalmente, se realizaron pruebas con el guardián esteganográfico, además de los gateways de embebido y extracción esteganográfica. Las cuales, permitieron comprobar nuevamente el correcto funcionamiento de la comunicación genuina, al tiempo que se vio frustrada la comunicación esteganográfica (que es el objetivo del guardián).

A partir de las técnicas aplicadas con la configuración del artefacto en baja capacidad esteganográfica (i.e. sin usar el campo Opciones), se comunican 5 bits por datagrama. Es decir, la *capacidad esteganográfica* es de 5 bits / mensaje portador. La longitud del datagrama es de 20 bytes, es decir, 160 bits. Por lo tanto, la *tasa esteganográfica* del algoritmo de embebido, medida en bits del mensaje esteganográfico sobre la cantidad de bits del portador, con esta configuración es:

$$\frac{5 \text{ bits}}{160 \text{ bits}} = 0,03125$$

Para el caso de la configuración con alta capacidad esteganográfica, se puede embeber un total de 293 bits por datagrama (288 bits en el campo de direcciones IP de *Record Route* dentro del campo Opciones y 5 bits en el resto del encabezado). En consecuencia, la *capacidad esteganográfica* es de 293 bits / datagrama IP. El datagrama modificado tiene una longitud de 60 bytes (es la máxima longitud posible para el datagrama, con el campo Opciones completo en su totalidad), lo que equivale a 480 bits. La *tasa esteganográfica* del algoritmo de embebido, para este caso es:

$$\frac{293 \text{ bits}}{480 \text{ bits}} = 0,610\ldots$$

Apéndice A.3. Artefacto sobre el protocolo HTTP

La implementación del artefacto sobre este protocolo actúa como proxy HTTP en la comunicación entre el cliente (un navegador web, e.g. *Mozilla Firefox*) y un servidor (e.g. *Apache Web Server*).

Se desarrolló un proxy que puede ser configurado para actuar con alguno de los siguientes roles: proxy transparente (reenvía los mensajes que recibe, sin modificarlos), escritor esteganográfico (embebe porciones del mensaje esteganográfico en el mensaje HTTP), lector esteganográfico (interpreta el mensaje esteganográfico en el mensaje recibido) o guardián activo esteganográfico (introduce *ruido* en el mensaje para evitar la aplicación de las técnicas esteganográficas, sin alterar las comunicaciones genuinas).

Se puede observar en este caso que es necesario que el proxy que realiza la escritura esteganográfica realice su comunicación hacia otro proxy, y que la cadena de reenvíos entre proxys debe incluir al receptor esteganográfico. A diferencia de lo implementado para el protocolo IP, el emisor esteganográfico aplica su algoritmo sobre *todos* los mensajes que debe reenviar, independientemente del destino final. Por ejemplo, si debe actuar sobre los mensajes de pedido, aplica su mensaje a *todos* los mensajes de pedido que deba reenviar. Lo mismo ocurre para el receptor esteganográfico: aplica el algoritmo de lectura sobre todos los mensajes que recibe, independientemente del origen. La diferencia con el caso de IP es que los mensajes HTTP están dirigidos al servidor y deben atravesar los reenvíos de los proxys, mientras que los datagramas IP pueden estar destinados a un host en una subred intermedia y no necesariamente atravesar todos los gateways.

La implementación de esta versión del artefacto se ha sustentado en el uso de *sockets* que provee la API del sistema operativo utilizado. A diferencia del caso de IP, el artefacto sobre el protocolo HTTP actúa con los roles de cliente y servidor: el proxy actúa como servidor al esperar recibir conexiones de otros clientes, y actúa como cliente al momento de reenviar el mensaje al nodo siguiente en la cadena de reenvíos.

Dado que debe atender múltiples conexiones, esta versión del artefacto se desarrolla en múltiples procesos de ejecución. La comunicación entre los distintos procesos se sustenta a partir del uso de estructuras IPC (*Inter-process communication*) que provee el sistema operativo: memoria compartida y semáforos.

A continuación se describe las técnicas esteganográficas aplicadas y el protocolo propuesto que se ha implementado sobre el mensaje de pedido, en el orden en que se realizan sobre el portador:

- Se agrega el campo de encabezado correspondiente a la dirección de correo electrónico (el campo *From*). Si este campo ya existía en el mensaje, no se aplica la técnica. En este campo se embebe un sólo bit del mensaje esteganográfico: si se debe colocar un bit de valor 1, se coloca como contenido de campo la dirección `example1@example.com`.
  Si se debe colocar un bit 0, se coloca la dirección `example0@example.com`. Cabe mencionar que, al ser la dirección de correo electrónico un campo formado por una cadena de caracteres, podría ser utilizado para embeber mayor cantidad bits del mensaje esteganográfico. Con el artefacto desarrollado se pretende mostrar la factibilidad del uso de este campo con estos fines.
- Se incorpora un campo propietario. En este campo se embebe un bit del mensaje esteganográfico. Si se debe agregar un bit de valor 1, se agrega un campo con el nombre *Example1*, si se debe agregar un bit de valor 0, se

agrega un campo con el nombre *Example0*.

- Cuando se trata de un mensaje dirigido a un servidor en el puerto 80, se embebe un bit a partir de indicar explícitamente o no el valor 80, que es el valor de puerto estándar. Esta modificación se realiza tanto en la URI de la línea de pedido, cuanto en el contenido del campo *Host*. Si se debe colocar un bit de valor 1, se indica explícitamente el valor 80, en caso contrario, se quita el número 80.
- Se transforma un campo que contiene una lista de valores en múltiples campos, cada uno con un único valor de la lista. El campo que se ha decidido modificar de esta forma es *Accept-Charset*. Este campo contiene una lista de codificaciones de caracteres que el cliente puede aceptar. En este campo se embebe un bit. Si se debe colocar un bit de valor 1, se convierte el campo con la lista a múltiples campos. Para embeber un bit de valor 0, se coloca la lista en un único campo.
- Se agregan comentarios, que son indicados entre paréntesis en el valor del campo. Se decidió colocar comentarios en el campo *User-Agent*, que contiene información acerca del software que utiliza el usuario. Se coloca un comentario compuesto por 3 letras. Cada letra está formada por 5 bits que se extraen del mensaje esteganográfico. Para obtener un valor que corresponda a un caracter imprimible, el valor de los 5 bits (que puede estar comprendido entre 0 y 31) se suma al valor del caracter arroba (`@`) en la codificación ASCII utilizada por el lenguaje de programación.
  En consecuencia, con esta técnica se embeben 15 bits del mensaje esteganográfico.
- Se reordenan los campos del encabezado. Los campos se reordenan de a pares. Para ello, se contabiliza la cantidad de campos de nombre diferente que contiene el mensaje (los campos de igual nombre son colocados juntos). Luego, se divide esa cantidad por 2 (si se obtiene un valor no entero, se toma la parte entera). La cantidad obtenida es la cantidad de bits que se embeben con esta técnica en el mensaje portador específico. Se extrae del mensaje esteganográfico tantos bits como los que se debe colocar. Inicialmente se ordenan los campos alfabéticamente. Se toman los campos de a pares y se reordenan entre sí de la siguiente manera: si el bit del mensaje esteganográfico es de valor 1, se coloca primero el campo mayor según el orden alfabético, y si el bit es de valor 0, se coloca primero el campo menor en orden alfabético. Se repite el procedimiento hasta colocar todos los bits.
- Se aplica *folding* al contenido de los campos. El folding permite representar un caracter espacio (`SP`) por un salto de línea más un espacio. Se contabiliza la cantidad de campos que poseen al menos un caracter espacio en su contenido. Ésta es la cantidad de bits del mensaje esteganográfico que se embebe con esta técnica. Se arma una lista de bits (extraídos del mensaje esteganográfico) cuyo tamaño es la cantidad de bits a colocar. Por cada campo con un espacio, si el bit a colocar tiene valor 1, se aplica folding. Si, en cambio, el bit tiene valor 0, se quita el folding del campo.
- Se agregan caracteres `LWS` (caracter espacio o tabulador) al comienzo del valor de cada campo (dado que son ignorados en la interpretación del contenido del campo). En cada campo se embeben 3 bits. Si el último bit

de los tres es un 1, se agregan tabuladores, si es un 0, se agregan espacios. Con los otros dos bits se forma un número (de 0 a 3), se lo incrementa en una unidad y el valor obtenido es la cantidad de espacios o tabuladores que se colocan al comienzo del valor del campo.

- Se modifican las mayúsculas y minúsculas de los nombres de cada campo, dado que según la especificación se ignora si el nombre se encuentra en mayúsculas o minúsculas. Se embebe un bit en cada campo. Si el bit del mensaje es un 1, se coloca la primera letra del nombre del campo en mayúsculas, si el bit es un 0, se coloca la primera letra del nombre del campo en minúsculas.

  En este caso también podría haberse decidido modificar las mayúsculas y minúsculas de más de una letra del nombre del campo, con la finalidad de embeber más de un bit en cada campo.
- Se embebe un bit del mensaje esteganográfico en una línea adicional antes de la línea de pedido del mensaje. Si el bit del mensaje es un 1, se agrega una línea adicional antes de la línea de pedido. En caso contrario, se quita la posible línea adicional.
- Se embebe un bit en la forma de escribir la versión HTTP del mensaje en la línea de pedido. Dado que es equivalente colocar `HTTP/1.1` que `HTTP/01.1`, si se debe colocar un bit de valor 1 del mensaje, se utiliza la segunda forma, y para colocar un bit de valor 0, se utiliza la primera. Esta técnica se aplica sólo si la versión HTTP del mensaje portador es 1.1.

A continuación se enumeran las técnicas aplicadas sobre los mensajes de respuesta por parte del escritor esteganográfico.

- Se incorpora un campo propietario, de forma equivalente a la realizada para el mensaje de pedido.
- Se agregan caracteres `LWS` al comienzo de los valores de todos los campos. Se embeben 3 bits del mensaje esteganográfico por cada campo.
- Se modifican las mayúsculas y minúsculas del nombre de los campos, de forma equivalente a lo que se realiza para el mensaje de pedido. Se embebe un bit en cada campo.
- Se agregan comentarios en el campo *Server*, que contiene información sobre la identificación del servidor, de forma análoga a la información sobre el cliente en el campo *User-Agent*. Al igual que para el mensaje de pedido, se embeben 15 bits con esta técnica.

Por otra parte, se implementó una configuración de guardián activo, que realiza las modificaciones mencionadas para embeber bits aleatorios, que impidan la comunicación esteganográfica propuesta.

Se realizaron diversas pruebas a partir de configurar el navegador *Mozilla Firefox* para que utilice como proxy al artefacto desarrollado, el cual, a su vez se conecta a otro proxy que actúa como receptor esteganográfico de los mensajes del anterior.

Se realizaron accesos a diversos sitios de Internet. Se comprobó que la navegación funcionó correctamente y, por otra parte, se observó que se desarrolló la comunicación esteganográfica entre ambos proxies.

Por otra parte, se realizaron pruebas con un proxy adicional configurado con el rol de guardián, ubicado en el lugar intermedio en el camino entre los

otros dos proxies. Se realizó el mismo uso de acceso a páginas web que en la prueba anterior. Se pudo comprobar nuevamente el correcto funcionamiento de la navegación, mientras que se observó que el proxy receptor esteganográfico no fue capaz de interpretar el mensaje que pretendía enviarle su par emisor. Lo cual, muestra que el guardián esteganográfico cumplió con su objetivo de impedir la realización de esa comunicación, sin afectar la comunicación legítima.

Para calcular la capacidad y tasa esteganográfica de esta versión del artefacto, se analizó un mensaje capturado con el analizador de tráfico *Wireshark* (*http://www.wireshark.org/*).

Se observó que en un mensaje HTTP portador en particular se pudieron comunicar 68 bits con las técnicas implementadas. A diferencia del caso de IP, la cantidad de bits que es posible embeber a partir de algunas de las técnicas implementadas para HTTP depende del mensaje, por ejemplo: el ordenamiento de los campos permite embeber una cantidad de bits igual a la mitad de la cantidad de campos del encabezado.

En ese caso, el mensaje portador (i.e. el encabezado HTTP del mensaje de pedido) tiene una longitud de 511 bytes, lo que equivale a 4.088 bits. Esta longitud corresponde al mensaje modificado por la función de embebido, tal como es el que procesa el servidor. La tasa esteganográfica del algoritmo de embebido implementado aplicado sobre el portador analizado es:

$$\frac{68 \text{ bits}}{4{,}088 \text{ bits}} = 0,016\ldots$$

La *capacidad esteganográfica* es de 68 bits / mensaje HTTP.

Apéndice B. Ejemplos de ubicación de los participantes esteganográficos

En el presente apéndice se muestran ejemplos de ubicación de los participantes esteganográficos para comunicaciones en los dos protocolos.

La figura 2 muestra un conjunto de tres redes conectadas a través de dos intermediarios (*Alicia* y *Walter*).

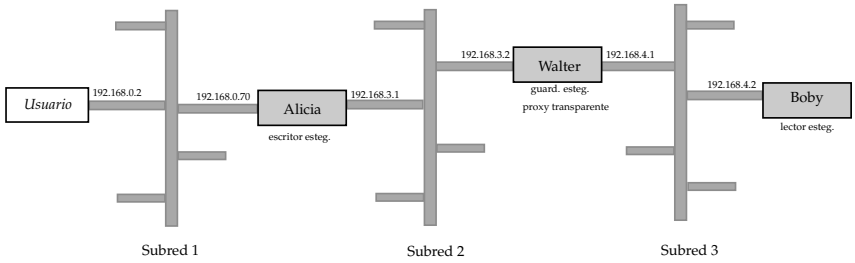

Figura 2: Esquema de una comunicación IP con dos subredes

Si se desea realizar una comunicación esteganográfica dentro de la ruta que recorre los datagramas en el camino entre la computadora identificada como

Usuario y *Boby*, se puede aplicar el embebido esteganográfico en el nodo *Alicia* y la extracción en el equipo *Boby*. En este caso, el mismo receptor IP realiza la extracción del mensaje esteganográfico antes de la interpretación del datagrama de acuerdo al protocolo.

El nodo identificado con el nombre Walter actúa como guardián activo, impidiendo la realización de la comunicación esteganográfica entre los que realizan el embebido y la extracción de la misma.

En todos los casos, el receptor del mensaje legítimo del protocolo IP recibe el datagrama con la semántica inalterada, respecto del generado por el emisor del protocolo.

La figura 3 esquematiza la comunicación a través del protocolo HTTP con una cadena de reenvíos de tres proxys entre el emisor y receptor. El cliente ejecuta un browser (como Mozilla Firefox) y el servidor ejecuta un software servidor HTTP como (como Apache HTTP Server).

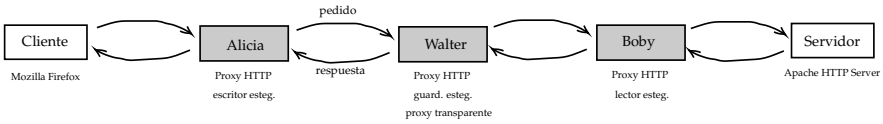

Figura 3: Esquema de una comunicación HTTP con dos subredes

Para los mensajes de pedido HTTP (que se transmiten desde el cliente hacia el servidor), el proxy *Alicia* realiza el embebido de su mensaje esteganográfico, sin alterar la semántica del mensaje HTTP portador. Mientras que *Boby* realiza la extracción (i.e. la interpretación) de ese mensaje esteganográfico.

El nodo Walter actúa como guardián activo esteganográfico para impedir que se realice la comunicación esteganográfica entre *Alicia* y *Boby*, pero sin afectar la comunicación entre el cliente y el servidor.