# Bypassing HSTS


Sebastián Norberto Mussetta
Dpto Ing. en Sistemas de Información
UTN – Facultad Regional Villa María
smussetta@frvm.utn.edu.ar

Norberto Gaspar Cena
Dpto Ing. en Sistemas de Información
UTN – Facultad Regional Villa María
ngcena@frvm.utn.edu.ar

Ignacio Daniel Favro
Dpto Ing. en Sistemas de Información
UTN - Facultad Regional Villa María
idfavro@frvm.utn.edu.ar



## Abstract

*La implementación del protocolo Http[1] para el servicio de aplicaciones Web ha tenido la necesidad de adaptarse mediante diferentes mecanismos para poder brindar seguridad al mismo. Http utiliza Tcp[2] como protocolo de transporte de los datos de aplicación sin proveer seguridad, permitiendo que intrusos, a través de simples ataques logren observar y/o modificar fácilmente la información que viaja por Http. Se han creado diferentes mecanismos de seguridad tales como SSL[3] y TL[4]S sobre Tcp, que al ser utilizados por Http, proveen integridad, confidencialidad e identificación segura de origen dando origen al protocolo Https. La incorporación de Ssl sobre http no es suficiente para prevenir ataques de tipo Man in the Middle ya que a través de mecanismos SslStrip[5] se pueden realizar estos ataques. Hsts surgió para forzar a que determinados servicios web implementados con Http sean accedidos exclusivamente a través de https impidiendo realizar ataques SslStrip. Lo que se pretende en este trabajo es demostrar la posibilidad de realizar ataques Man in the Middle por medio de diferentes aplicaciones a sitios que utilizan Hsts[6].*


## 1. Palabras claves


Hsts, Man in the Middle, Http, Https, Ssl, Tls, Tcp, SslStrip


## 2. Introducción

El surgimiento del protocolo http contribuyó notablemente en el cambio de la forma en que usuarios hacen uso de Internet. Aplicaciones Webs orientadas a diferentes ámbitos tales como Webmails, Home Banking, Redes Sociales, E-Commerce, Educación, entre otras, han sido implementadas desde ya hace varios años mediante el protocolo Http. Este protocolo utiliza como transporte TCP, el cual no garantiza integridad, confidencialidad ni identificación segura de origen. Diferentes mecanismos de ataques a este protocolo se han diseñado con el paso del tiempo a través de intrusiones generadas por un atacante para observar y/o modificar tráfico http de manera indiscreta. A partir de la necesidad de brindarle seguridad al protocolo han surgido las implementaciones de capas sobre TCP tales como SSL y TLS para soportar canales seguros. Así surge Https, el cual provee un canal seguro a nivel de aplicación entre el servidor y cliente Web. De esta manera, un intruso podrá capturar tráfico Https pero el mismo será ilegible. Con el tiempo, las organizaciones fueron migrando sus servicios originales implementados con Http hacia Https. Desde hace ya un tiempo, Google ha aumenta el ranking de búsqueda a sitios que utilicen Https, esto favorece la migración de Http a Https. Cabe destacar que hoy en día muchos de los servicios webs en Internet aún están siendo accedidos a través de Http. Esto es considerado un problema grave de seguridad ya que por medio de diferentes ataques Man in the Middle se puede capturar trafico legible http muy fácilmente sin que la víctima tome conocimiento. En el año 2009 se ha presentado el ataque SslStrip1 donde mediante un ataque Man in the Middle, un atacante puede reemplazar todas las conexiones Https por Http logrando capturar todo el tráfico generado de manera legible. En este caso, las conexiones entre la víctima y el atacante se realizan a través del protocolo Http, y el tráfico entre el atacante y el destino se realiza a través de Https. Habitualmente, la

víctima no advierte el ataque ya que sólo se cambia en la Url del navegador Https por Http. Existen numerosas aplicaciones para diferentes sistemas operativos que automatizan este ataque permitiendo a un atacante poder ejecutarlas y lograr su objetivo sin tener demasiados conocimientos técnicos de cuál es la vulnerabilidad en cuestión, ni cómo se realiza el ataque. Estas herramientas están diseñadas para usuarios finales sin grandes conocimientos técnicos sobre seguridad informática. En este sentido, la implementación de Https no es garantía de seguridad para prevenir este tipo de ataques. Como se mencionó anteriormente, el problema de este tipo de ataque radica en que el cliente termina realizando una conexión Http en lugar de Https. Para resolver este problema se diseñó Hsts, el cual es un mecanismo donde el servidor Web le anuncia al User Agent del cliente que dicho sitio debe ser accedido de manera exclusiva a través de Https. En principio, el cliente solo podrá acceder al sitio a través de Https. Cabe destacar que actualmente, muchos de los sitios conocidos con servicios de Home Banking, Webmail, entre otros, aún no utilizan cabeceras Hsts. En las secciones posteriores de este trabajo se pondrán de manifiesto ataques Man in the Middle a sitios que utilizan la Hsts.

## 3. Elementos de Trabajo y Metodología

El equipamiento físico que se detalla a continuación fue el utilizado para dar soporte de Hardware a todas las actividades del proyecto.

- Equipo de computación: procesador Intel Core i5, 8 Gb Ram, 1 Tb Disco Rígido
- Switch Hp Serie 1910
- Router Cisco 1841 (Conexión a Internet 5Mb Dowstream / 1 Mb Upstream)

Para simplificar y optimizar los recursos de Hardware se virtualizaron diferentes sistemas operativos por medio de la implementación de Proxmox[7] Ve 4.2. A continuación se describe la topología del despliegue utilizado.

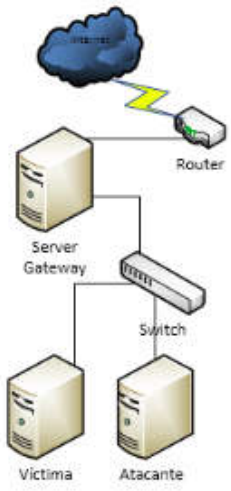


**Figura 1. Topología**

Se han creado 2 máquinas virtuales:

- Máquina Virtual 1 Víctima: Microsoft Windows 7, Navegadores Mozilla Firefox 48.0.1, Microsoft Internet Explorer 8, Google Chromme 52. Ip 192.168.100.133/24, Gateway 192.168.100.1
- Máquina Virtual 1 Atacante: Gnu/Linux Debian 8.5, Ip: 192.168.100.233/24, Gateway 192.168.100.1

Server Gateway: Gnu/Linux Debian 8.5, Ip eth0: 192.168.100.1/24, Ip eth1: 192.168.2.2/30, Gateway 192.168.2.1/30

### Software Utilizado

Se ha realizado un intenso estudio de las herramientas disponibles para la realización de ataques Man in the Middle con sitios que además de implementar Https utilizan Hsts. Se han seleccionado 2 herramientas para la realización de los ataques. Estas herramientas se han elegido por su flexibilidad, desempeño y por sobre todas las cosas, su facilidad de utilización, además de ser open source. En este trabajo se pretende demostrar la simplicidad en la realización de los ataques sin requerir mayores esfuerzos ni conocimientos exhaustivos sobre seguridad informática. Por otro lado se intenta señalar que a medida que la tecnología avanza en aplicaciones, servicios, protocolos, también se desarrollan nuevos ataques automatizados para explotar las vulnerabilidades generadas por el avance de la tecnología. Un claro ejemplo de ello es la evolución de los protocolos Http, Https y Hsts.

Las aplicaciones seleccionadas son Bettercap[8] y Mitmf[9]. Estas aplicaciones utilizan para realizar el ataque man in the middle arp soofing[10], dns spoofing y un proxy server. Cuando un cliente intenta realizar una conexión http o https, el atacante realiza arp spoofing y comienza con la captura y reenvió de paquetes a la puerta de enlace, luego realiza dos posibles ataques:

Ataque 1 (Downgrade http): sslStrip1 si el servidor web de destino no provee cabeceras Hsts. Este ataque consiste en que el atacante a través de man in the middle logre realizar una conexión Https contra el servidor web de destino y una conexión Http con el cliente. De esta manera implementando un proxy podrá realizar el ataque y capturar toda la información que se transporta en texto plano a través de http.

Ataque 2 (Downgrade http con Hsts bypass) sslStrip2: este ataque es similar al anterior salvo que cuando el cliente se intenta conectar con un servicio web, puede que el servidor le envíe las cabeceras Hsts indicando que solo se podrá conectarse a ese dominio a través de Https. En algunos casos, dependiendo de la versión y tipo de navegador, el cliente puede tener configurado de antemano una lista de dominios a los que se tiene que conectar exclusivamente por Https. A partir de la restricción de acceso exclusivo a través de Https a determinado dominio, ya sea porque el servidor envía cabeceras hsts o porque el navegador ya se encuentre configurado para ese dominio el acceso exclusivos por medio del protocolo Https, el ataque se realiza de la siguiente manera: cuando se detecta cabeceras hsts en la conexión el atacante realiza arp spoofing y dns spoofing en conjunto con la utilización de un proxy cambiando el subdominio o dominio de la conexión con el cliente y accediendo al dominio real a través de https. El cliente, no utilizará https ya que para él, la conexión se realiza a otro dominio que no tiene ninguna restricción de acceso. El tráfico es redirigido al servidor web a través de Https de manera transparente para el usuario. El atacante ha podido capturar la información de manera legible. Ej.

```
<a
href="https://www.outlook.com/">Login<
/a>
Bypass              Hsts:              <a
href="http://wwww.outlook.com/">Login<
/a>
```

A continuación se describen los pasos necesarios para la realización de los ataques por medio de la herramienta Bettercap y luego Mitm Framework que se ejecutan en el equipo atacante.

### Etapa 1: Bettercap

Paso 1: Instalación en Debian Gnu/Linux 8.5.
Para que la aplicación pueda ejecutarse se deben satisfacer las siguientes dependencias de librerías, luego se agregan los repositorios de Kali[11] Linux y por último se instala la aplicación:

- aptitude install build-essential ruby-dev libpcap-dev.
- echo deb http://http.kali.org/kali kali-rolling main contrib non-free >> /deb/apt/sources.list
- apt-get update
- apt-get install bettercap

Paso 2: Ejecución del ataque

- bettercap -T 192.168.100.34 --proxy -P POST

### Etapa 2: Mitm Framework

Paso 1: Instalación en Debian Gnu/Linux 8.5.
Para la instalación de la ejecución se debe descargar la última versión desde el repositorio github, luego ejecutar el instalador, y por último descargar e instalar componentes requeridos de Python.

- git clone https:/ /github.com/byt3bl33d3r/MITMf.git
- ./setup.sh
- pip2 install --upgrade -r requirements.txt

Paso 2: Ejecución del ataque
python mitmf.py -i eth0 --spoof --arp --hsts --dns --gateway 192.168.100.1 --target 192.168.100.34

## 4. Resultados

Las pruebas realizadas fueron en su mayoría satisfactorias respecto al éxito obtenido con el ataque. Se han logrado capturar credenciales de acceso de diferentes servicios webs aún en aquellos que tienen implementado Hsts. Se han realizado pruebas desde diferentes navegadores hacia diversos sitios webs, algunos de ellos muy populares como Hotmail, Gmail, Facebook y otros no tan conocidos pero no de menor importancia como servicios de Home Banking nacionales y servicios de correo web Institucionales públicos y privados. A modo de ejemplo se presentarán capturas de pantallas de algunos de los ensayos realizados con las herramientas Bettercap y Mitm Framework presentadas anteriormente. La preparación del ambiente y la realización de los ataques no requirieron mayores esfuerzos ni profundos conocimientos en materia de seguridad informática.

- Bettercap Ejemplo 1

Este caso de prueba a modo de ejemplo se realizó desde Sistema Operativo Ms Windows 7 y Navegadores Mozilla 48.0.1 y Google Chromme 52 hacia el servicio web de correo Hotmail-Outlook.


**Figura 2. Ejecución de la herramienta**


**Figura 3. Consulta soporte hsts**

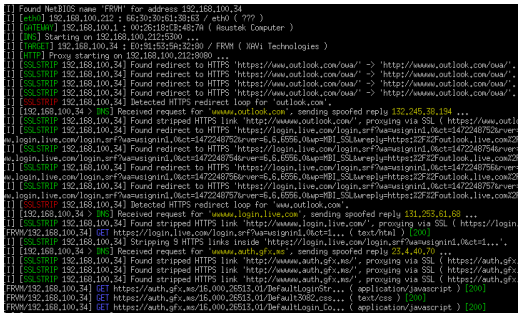

**Figura 4. Ejecución del ataque**


**Figura 4. Redirección wwww**

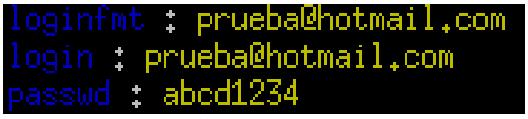


**Figura 5. Credenciales capturadas**

- Bettercap Ejemplo 2

En este ejemplo se realizó el ataque contra el sitio Gmail.com por medio de la utilización de Internet Explorer 8


**Figura 6. Redirección wmail.google.com**


**Figura 7. Credenciales capturadas**

- MitmF Ejemplo 1 Facebook.com

Con la utilización de esta herramienta se han ejecutado los mismos ataques que con Bettercap y los resultados fueron los similares. A continuación se muestran dos ataques, uno a Facebook.com y otro a Gmail.com pero con las últimas versiones de los navegadores, Mozilla Firefox 48.0.1 y Google Chromme 52. En este caso, como estos navegadores tienen de manera predeterminada la configuración Hsts a ambos sitios, el ataque en un principio no funcionó. Una vez eliminadas las entradas de configuración Hsts de manera manual en el navegador para esos sitios, el ataque comenzó a funcionar. Luego, se han dejado las entradas predeterminadas nuevamente y ejecutados los ataques. Los ataques fueron exitosos cuando, el cliente en vez de escribir de manera manual la url en la barra de direcciones, realizaba una búsqueda de los dominios Facebook.com y Gmail.com en algún buscador y luego accedía a través del link de resultado de la búsqueda realizada.


**Figura 8. Ejecución de la Herramienta**

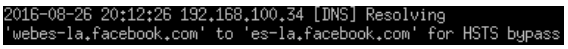


**Figura 9. Ejecución del Ataque**


**Figura 10. Credenciales capturadas**

- MitmF Ejemplo 2 gmail.com


**Figura 11. Credenciales capturadas**

## 5. Discusión

Los resultados generados a partir de las diferentes pruebas que se han realizado ponen de manifiesto la problemática existente a la hora de garantizar una conexión https entre el cliente y el servidor de extremo a extremo. Algunos sitios muy populares en Internet, tales como Facebook, Gmail, Twitter, entre otros, están configurados por defecto en las últimas versiones de los navegadores, para que la conexión hacia los mismos sea exclusivamente Https. De esta manera, se dificulta el ataque ya que no se podrá realizar el mismo utilizando la técnica ssltrip. Cuando el usuario ingrese la url, el navegador sabrá que a ese dominio sólo se podrá acceder a través de https. En el caso que el sitio soporte hsts pero no se encuentre en la lista predeterminada, el atacante se puede anticipar y realizar el ataque antes que el servidor le envíe las cabeceras hsts al navegador y en este caso el ataque resulta exitoso. A partir de lo mencionado se puede afirmar que existen diversos factores que contribuyen a la inseguridad en las comunicaciones http y https como por ejemplo:

- El servidor web de destino no soporta https: en este caso un simple ataque man in the middle a través de arp spoofing podrá capturar tráfico legible.
- El servidor soporta https pero no hsts: en este caso un ataque sslstrip resultará exitoso.
- El servidor soporta hsts pero no está preconfigurado en el navegador para forzar conexiones https: en este caso el ataque puede ser exitoso a través de un ataque sslstrip2. Esto puede darse porque el navegador se encuentra desactualizado o porque la organización propietaria del servicio web no negoció con el navegador para ser incluido en la lista.
- El servidor soporta hsts y está preconfigurado en el navegador para forzar conexiones https: este ataque resulta más difícil. Una técnica sería a través de un ataque man in the middle al cliente ntp de la víctima. De esa manera, al cambiar la fecha del equipo víctima, la configuración hsts predeterminada para ese destino en el navegador no será válida. Esto se debe a que uno de los campos de la configuración hsts es el tiempo de duración de la validez para las conexiones exclusivas https.

## 6. Conclusión

Se ha podido demostrar que los ataques man in the middle a los protocolos http y https continúan siendo un problema de seguridad importante ya que para realizarlos de manera exitosa no se requieren grandes esfuerzos. Existen diversos factores que contribuyen a que estos ataques sean posibles. Algunos de ellos se deben a que muchos de los servicios webs aun no soportan https, y si lo soportan puede que no soporten hsts. No obstante, al soportar hsts tampoco se garantiza evitar estos ataques. Son muy pocos los dominios que se encuentran precargados de manera automática en los navegadores. Por otro lado, es común que los clientes no tengan las últimas versiones instaladas de los navegadores. Si bien hsts contribuye a impedir estos ataques, no los elimina. Otro factor importante es el desconocimiento de los usuarios acerca de estos problemas de seguridad. Habitualmente los usuarios se conectan a servicios manipulando información confidencial a través de redes inseguras. Además, cabe destacar la cantidad de herramientas existentes, muchas de ellas de software libre, que simplifican los mecanismos de ataques permitiendo a que usuarios sin conocimientos técnicos profundos puedan realizar estos tipos de ataques muy fácilmente.

## 7. Referencias


[1] https://tools.ietf.org/html/rfc2616, Agosto 2016
[2] https://tools.ietf.org/html/rfc793, Agosto 2016
[3] https://tools.ietf.org/html/rfc6101, Agosto 2016
[4] https://tools.ietf.org/html/rfc5246, Agosto 2016
[5] https://moxie.org/software/sslstrip/, Agosto 2016
[6] https://tools.ietf.org/html/rfc6797, Agosto 2016
[7] https://www.proxmox.com/en/, Agosto 2016
[8] https://www.bettercap.org/, Agosto 2016
[9] https://github.com/byt3bl33d3r/MITMf, Agosto 2016
[10]ww.cisco.com/c/en/us/products/collateral/switches/catalyst-6500-series-witches/white_paper_c11_603839.html, Agosto 2016
[11]https://www.kali.org/, Agosto 2016