Tema 2
Ataques man in the middle

Ataques man in the middle

HTTPS

Bettercap

- **HTTPS: cómo espiar estas conversaciones**
- HTTP: los datos se mandan sin cifrar
- Mediante un ataque man in the middle es relativamente fácil espiar estas conversaciones → **No son seguras**

Bettercap

- **Solución:** HTTPS es una adaptación de http (http seguro)
- Usa TCP Puerto 443
- Cifra usando los protocolos de cifrado TLS o SSL
- Problema para nosotros: la mayoría de los sitios web usan https→ los datos capturados por un sniffer están cifrados
- Necesitamos saltarnos el https (**Bypassing https)**

Bettercap

- **¿Cómo funciona https?**
- Se crea una conexión cifrada entre el cliente y el servidor
- HTTPS nos da confidencialidad (datos cifrados), autenticidad (el servidor se autentica) e integridad (los datos no han sido cambiados durante el trayecto)

HTTPS

- **HTTPS**
- Para preparar un servidor web que acepte conexiones HTTPS, se debe crear un certificado de clave pública para el servidor web.
- Este certificado debe estar firmado por una autoridad certificadora (CA) para que el navegador web lo acepte (puesto que el navegador tiene precargada una lista de CA reconocidas).
- Este certificado puede ser gratuito o con un relativamente bajo coste

HTTPS

- **HTTPS**
- Por ejemplo, de pago como Verisign (https://www.verisign.com/), gratuitos como Let's Encrypt (https://letsencrypt.org/es/),...
- La autoridad certifica que el titular del certificado es quien dice ser.
- Actúa como un notario (dan fe)
- Por ejemplo: ver certificado de apple.es y el de ufv.es

HTTPS

- **HTTPS – Pasos.**
- **Comunicación basada en la confianza.**
- El usuario confía en que el navegador tenga implementado correctamente HTTPS con la lista de entidades certificadoras (CAs) ya cargada
  - Consultar las CAs de confianza que tiene nuestro navegador
  - Por ejemplo, en el Chrome ir a Configuración/certificados
  - Localizar las CAs de Apple (Digicert) y UFV (ver que quién emite este certificado en las propiedades de la conexión es DST Root CA X3)

HTTPS

- **HTTPS – Pasos.**
- **Comunicación basada en la confianza.**
- El ususario confía en que la CA va a responder solo de sitios webs legales.
- El servidor provee un certificado válido, lo que significa que está firmado o reconocido por una CA de confianza

HTTPS

- **HTTPS – Pasos.**
- **Comunicación basada en la confianza.**
- El certificado identifica correctamente al sitio web (por ejemplo, cuando visitamos "https://www.elpais.com", el certificado recibido está emitido para “elpais.com" y no para otro sitio.
- El usuario confía en que el protocol de cifrado (SSL/TLS) es suficientemente seguro para cifrar la comunicación y que no sea vista.

Bettercap

- **Pero justo antes de entrar ocurre esto:**
- Quiero entrar a https://www.marca.com
- El usuario teclea www.marca.com (es decir, envía una petición HTTP no segura)
- El servidor responde via HTTP (TCP 80) y lo redirige para que hable por HTTPS (TCP 443) con un paquete http GET 302

Bettercap

- **Pero justo antes de entrar ocurre esto:**
- El usuario envía una petición segura HTTPS, y empieza la sesión segura.
- Es decir, el primer mensaje es http, y el servidor lo redirige a https
- Eso se ve en el Wireshark, en un paquete que pone redirects https o en el propio sniffer de bettercap

SSL Stripping

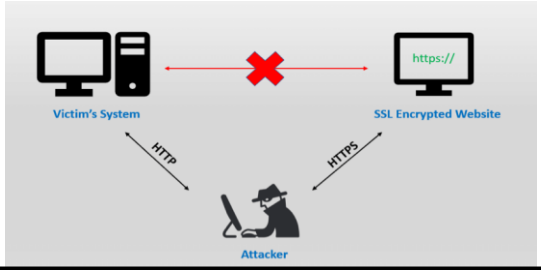
- **Solución:** Downgrade https to http
- Es decir, hacer un man in the middle y darle la versión http de la web en vez de https
- Se puede hacer manualmente con SSL Strip o con un framework (bettercap, mitmf,...) que integre esta aplicación

SSL Stripping

- **SSL Stripping**
- Creado por Moxie Marlinspike (https://moxie.org/software/sslstrip/)
- SSLStrip reemplaza todas las peticiones «https://» de una página web por «http://»
- Necesitamos hacer antes un MITM entre el servidor y el cliente, claro.

SSL Stripping

- **SSL Stripping**
- La víctima y el atacante se comunican a través de HTTP, mientras que el atacante y el servidor, se comunican a través de HTTPS con el certificado del servidor.
- Por lo tanto, el atacante es capaz de ver todo el tráfico en texto plano de la víctima.



Ataques man in the middle

Bettercap+SSL Strip

Bettercap+SSL Strip

- Bettercap trae ya un caplet preparado para eso, llamado hsthijack.
- Los caplets predefinidos de bettercap se encuentran en */usr/*share/bettercap/caplets
- Este caplet que viene por defecto no funciona bien, y os doy uno modificado basado en este
- En el campus se da este caplet (script) al que se ha añadido una opción para que salga la misma url pero con http en vez de https, y no cambie la url porque cantaría mucho, claro.
- Es una modificación del caplet que viene por defecto en bettercap

Bettercap

- Copiamos el directorio entero, no solo el script, en el directorio donde están los caplets, con el mismo nombre (hacer un backup del que viene por defecto antes)

Bettercap

- Arrancamos bettercap con el script que tenemos inicial (no el bajado para el https) que aranca el sniffer.

Bettercap

- En este script debo añadir varias cosas: una línea antes del net.sniff on que ponga **set sniff.local true**
  - Esto es porque si no bettercap cree que las passwords salen de mi equipo y no las muestra, pensando que son passwords mías.
  - También podemos comentar el net.sniff on y activarlo después, una vez dentro y cargado el caplet

Bettercap

- Una vez que ya tenemos nuestro caplet en su directorio por defecto vamos a usarlo
- Arranco bettercap con el script inicial:
  - bettercap -caplet ~/inicio_bettercap.cap
- En este script inicial, habilitar el filtro de las passwords y redirigir la salida a un fichero
- Ponemos help para ver todos los módulos que se están ejecutando y comprobar así que funciona bien mi script

Bettercap

- Si escribimos caplets.show veo todos los caplets que trae bettercap (salen todos los que hay y su localización)
- Escribo hsts y tabulo para que complete
- Al darle al enter ya carga ese caplet

Bettercap

- Lo probamos en el **cliente**
- En el cliente borrar la caché del navegador (clear browsing data)
- Voy a stackoverflow.com poniéndolo en la barra de navegación directamente
- Ver que en la url es http y no https
- Probar con un login y una password cualquiera, para ver si la vemos

Bettercap

- Buscar el paquete POST siempre
- Recordad que puedo enviarlo a un fichero, escribiendo en el script de inicio:
  - set net.sniff.regexp .*password=.+ (filtrar passwords)
  - set net.sniff.output passwords.cap
- Otros filtros para el sniffer
  - set net.sniff.filter tcp port 443
  - set net.sniff.filter "host 224.0.0.251 and port 5353"

Bettercap

- Si probamos con facebook.com, vemos que no funciona.
- ¿Por qué? Por las cabeceras hsts (**hsts headers**)

Ataques man in the middle

HSTS headers

HSTS

- **HSTS** → HTTP strict Transport security
- Usado por Facebook, Twitter,...
- Los servidores evitan los ataques anteriores (downgrade to http) implementando en el servidor web hsts (HTTP Strict Transport Security)
- El servidor obliga al cliente (navegador) a cargar la página en https
- Esto lo hace en la primera carga de la página

HSTS

- **La clave**: si me hacen un mitm antes de la primera carga este hsts no funcionará
- **Si entra la víctima por primera vez no funcionará salvo que borre el caché**

HSTS

- Los navegadores modernos (Chrome, Firefox, etc…) están obligados a solo hablar por https con los sitios web que aparecen en una lista
- Es decir, llevan precargadas una lista de sitios web y sólo cargan estas webs si van sobre https, no sobre http, por lo que no puedo hacer el download
- Para consultar esta lista: https://hstspreload.org/

HSTS

- La lista de Chrome se puede ver en: https://src.chromium.org/viewvc/chrome/trunk/src/net/http/transport_security_state_static.json
- Pero claro, no pueden tener precargadas todos los sitios web de Internet, sólo los más comunes.
- Para que te incluyan en esta lista: https://hstspreload.org/
- Ejemplo: allmusic.com (meterlo primero en https://hstspreload.org)

Bettercap

- Otra forma de saltárselo es engañar al navegador y cambiar la conexión de facebook.com a facebook.corn, por ejemplo (y no se va a notar)
- Otro ejemplo: puedo usar twiter.com en vez de twitter.com por ejemplo
- Ver el caplet de hsthijack (/usr/share/bettercap/caplets/hsthijack)

Bettercap

- Abrirlo para verlo con un editor primero.
- Ver las líneas targets y los puedo modificar.
- Obfuscade y encode está en false porque por ejemplo Firefox no carga páginas si esto está en true
- Ver dns.spoof.domains para que cuando vaya a twiter.com le mande yo el website clonado

Bettercap

- Entrar en bettercap con el sript inicial y luego hsts y tabulador para cargar el caplet
- Ir al cliente y entrar en facebook.com (eliminar antes la caché)
- Pero no puedo ponerlo en la barra de direcciones porque lo va a cargar en https.
- Ir a google primero, ver que está en http y no en https.
- Buscar facebook y acceder desde ahí, y ver que me ha llevado a http y .corn, no .com.
- Puedo cambiarlo como quiera, sed creativos (facebok, faceboook,faceboock, …)

Bettercap

- Poner el login y password 123456789 y ver que sale en bettercap y fijarse en el host (.corn)
- Pero claro, debe acceder a facebook desde otro website al que yo le haya hecho el download, no directamente.
- No es una solución total, es parcial, pero es lo único que hay ahora ☹

Ataques man in the middle

DNS spoofing

DNS spoofing

- **DNS Spoofing**
- ¿Qué es el servicio DNS?
- ¿Qué es el DNS spoofing? Engañar a la víctima con un ataque mitm y redirigirlo a mi propio servidor web cuando él quiera ir al banco o a otra web.

DNS spoofing

- **Pasos**
- Arrancar el apache de Kali (service apache2 start).
- Modificamos la index (/var/www/html) – copiarlo y en el index borrarlo y ponemos lo que queramos (Hola, estás siendo atacado ☺)
- Arranco el bettercap con el script inicial de siempre
- help → módulo dns.spoof

DNS spoofing

- Options: help dns.spoof
  - dns.spoof.adress – a qué ip quiero redirigirlo. Es la mía, y aparece por defecto
  - dns.spoof.all – set dns.spoof.all true (responde a todas las peticiones DNS)
  - set dns.spoof.domains ufv.es, *.marca.com
- dns.spoof – lanzo el módulo
- Probarlo en el cliente. Ojo a la caché del navegador si no funciona.
- Esto se suele usar clonando páginas de login o sitios webs completos

Ataques man in the middle

Inyectar código javascript

Bettercap

- **Injecting Javascript Code**
- Ejemplo de javascript (alert.js) → alert('Hola, te estoy espiando');
- Vamos a la carpeta /usr/share/bettercap/caplets/hstshijack, y abro el fichero hstshijack.cap
- ver en ese fichero la opción set hstshijack.payloads que inyecta un fichero llamado keylogger.js.
- Añadir una coma y poner *:ruta_del_fichero_javascript (para decir que lo inyecte en cualquier web a la que acceda el cliente)

Bettercap

- **Injecting Javascript Code**
- Arranco bettercap como siempre, con el script de inicio
- Cargo el caplet hstshijack y veo que ha cargado ese payload (alert.js)
- Probarlo con una web http (vulnweb.com) y otra con https (stackopverflow.com)