Tasca 1.

a) Quin és el protocol de xifrat que fa servir HTTPS?

HTTPS utilitza TLS (Transport Layer Security) com a protocol de xifrat, que és l'evolució moderna de SSL.

b) Quina és la diferència entre SSL i TLS?

SSL (Secure Sockets Layer) és el protocol original, ara obsolet. TLS (Transport Layer Security) és la versió moderna i més segura que el va substituir. TLS ofereix millors algoritmes de xifrat i més seguretat.

c) De quines dues fases consta el protocol TLS?

Handshake (encaixada de mans): fase d'autenticació i negociació

Transmissió de dades xifrades: fase on s'intercanvien les dades protegides

d) A la fase de hadshake, quin extrem HTTP proporciona el certificat SSL/TLS?

El servidor és qui proporciona el certificat SSL/TLS al client per demostrar la seva identitat.

e) Com s'aconsegueix un certificat SSL/TLS?

Sol·licitant-lo a una Autoritat Certificadora (CA) reconeguda.

Creant un certificat autosignat (com fareu a la pràctica amb openssl).

f) Com es pot garantir que la clau de sessió generada pel client a la fase de handshake només podrà ser desencriptada pel servidor que ha presentat el certificat?

El client xifra la clau de sessió amb la clau pública del servidor (que ve en el certificat). Només el servidor, que té la corresponent clau privada, pot desxifrar-la.

g) Quina clau es fa servir per encriptar els missatges HTTP: la clau pública o la privada (clau de sessió)?

S'utilitza la clau de sessió (també anomenada clau simètrica), que és generada durant el handshake.

h) El procés d'encriptació és simètric o asimètric?

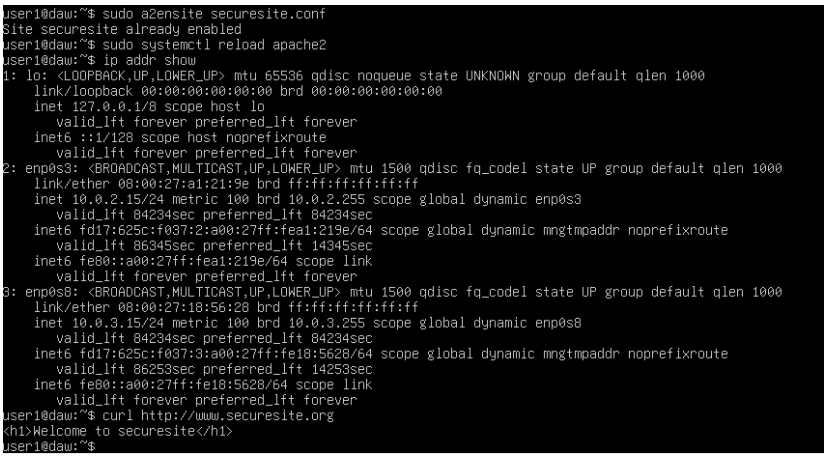
Asimètric durant el handshake (per intercanviar la clau de sessió).

Simètric per a la transmissió de dades HTTP (més eficient).

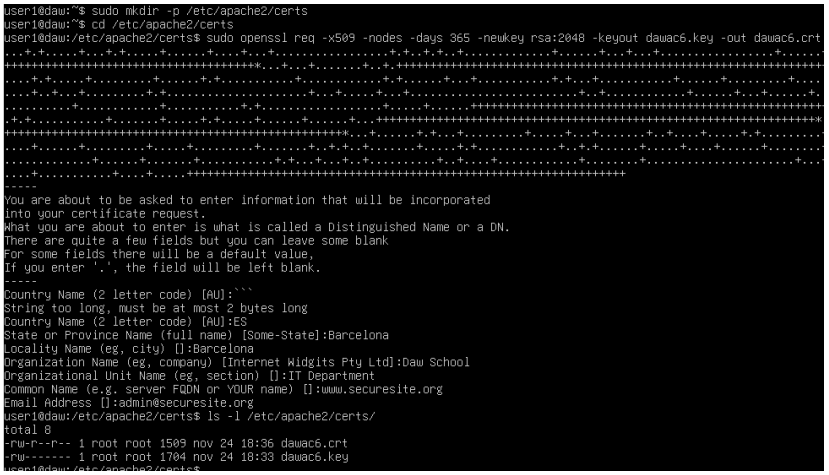
i) Què vol dir que el protocol TLS proporciona també integritat de dades?

Significa que TLS pot detectar si les dades han estat modificades durant la transmissió, utilitzant funcions hash (HMAC) per verificar que el missatge rebut és exactament el que es va enviar.

Tasca 2. Crea un servidor virtual pel domini www.securesite.org. Comprova que pots accedir-hi des del navegador i que aquest la considera una web no segura.



Tasca 3. A continuació, aconseguirem un certificat (més una clau pública) per a que el nou servidor virtual en faci ús.



d) Abans de continuar, analitzem la comanda que hem fet servir: per què hem fet servir l'opció -x509?

L'opció `-x509` indica a openssl que generi directament un certificat autosignat en format X.509, en lloc de generar només una sol·licitud de signatura de certificat (CSR). Això significa que el certificat està signat per la pròpia clau privada que acabem de crear, sense necessitat d'enviar-lo a una autoritat certificadora (CA) per a la seva signatura.

Tasca 4.

a) La principal limitació dels certificats autosignats és que els **navegadors no confien en ells automàticament**. Això passa perquè:

- **No estan verificats per una CA reconeguda**: Els navegadors tenen una llista de CAs de confiança preinstal·lades. Un certificat autosignat no està en aquesta llista.
- **El navegador mostra avisos de seguretat**: Apareix un missatge d'error amb text com "La connexió no és privada", "Risc potencial de seguretat", etc. Això espanta als usuaris.
- **L'usuari ha d'acceptar manualment el risc**: Per accedir al lloc, l'usuari ha de clicar diversos cops per acceptar el certificat, cosa que és incòmoda i perillosa (acostuma els usuaris a ignorar advertències de seguretat).

- **Inadequats per a webs públiques**: Són acceptables només per a:
  Entorns de desenvolupament i proves
  Xarxes internes corporatives
  Ús personal/local

- **Problemes amb APIs i aplicacions**: Moltes aplicacions i serveis rebutgen directament connexions amb certificats autosignats.

b) Les principals autoritats certificadores (CAs) actuals són:

- **DigiCert**: És considerada una de les autoritats certificadores amb major credibilitat a nivell mundial, reconeguda per oferir xifrat fort i atenció al client d'alta qualitat
- **Sectigo** (abans Comodo): És una autoritat de certificació consolidada que ofereix una àmplia gamma de certificats SSL i és especialment popular entre petites i mitjanes empreses pels seus preus competitius SSL Dragon
- **Let's Encrypt**: Ha guanyat popularitat per simplificar el procés d'obtenció de certificats SSL gratuïts i promoure l'adopció generalitzada d'HTTPS.
- **GlobalSign**: Es considera una autoritat de certificació innovadora que introdueix diverses funcions per millorar el rendiment dels llocs web protegits amb SSL.
- **Thawte**: Protegeix transaccions i identitats en més de 240 països, oferint certificats SSL de baix cost amb una reputació mundialment reconeguda.
- **GeoTrust**: Compta amb més de 100.000 clients de 150 països, i és ideal per a empreses que busquen protegir els seus llocs web sense sortir-se del pressupost.
- **RapidSSL**: És una empresa especialitzada en certificats SSL de qualitat a preus molt econòmics, amb un procés d'emissió totalment automatitzat i ultraràpid.
- Aquestes CAs estan reconegudes pels principals navegadors i sistemes operatius, i ofereixen diferents nivells de validació segons les necessitats de cada web.

Tasca 5.

```
user1@daw:~$ sudo apache2ctl configtest
AH00558: apache2: Could not reliably determine the server's fully qualified domain na
is message
Syntax OK
user1@daw:~$ sudo systemctl restart apache2
user1@daw:~$ curl https://www.securesite.org
curl: (60) SSL certificate problem: self-signed certificate
More details here: https://curl.se/docs/sslcerts.html

curl failed to verify the legitimacy of the server and therefore could not
establish a secure connection to it. To learn more about this situation and
how to fix it, please visit the web page mentioned above.
user1@daw:~$ curl -k https://www.securesite.org
<h1>Welcome to securesite</h1>
```

c) Si tens accés al navegador, aniràs a `https://www.securesite.org` i veuràs un avís de seguretat tipus "La connexió no és segura" o "Risc potencial de seguretat".

d) Perquè el certificat és autosignat i no està verificat per cap autoritat certificadora (CA) de confiança. El navegador no pot verificar l'autenticitat del certificat, per tant mostra l'avís de seguretat.

```
user1@daw:~$ curl -k https://www.securesite.org
<h1>Welcome to securesite</h1>
user1@daw:~$ curl -kv https://www.securesite.org 2>&1 | grep -E "SSL|TLS|cipher"
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
* TLSv1.3 (IN), TLS handshake, Server hello (2):
* TLSv1.3 (IN), TLS handshake, Encrypted Extensions (8):
* TLSv1.3 (IN), TLS handshake, Certificate (11):
* TLSv1.3 (IN), TLS handshake, CERT verify (15):
* TLSv1.3 (IN), TLS handshake, Finished (20):
* TLSv1.3 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.3 (OUT), TLS handshake, Finished (20):
* SSL connection using TLSv1.3 / TLS_AES_256_GCM_SHA384 / X25519 / RSASSA-PSS
*  SSL certificate verify result: self-signed certificate (18), continuing anyway.
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
* old SSL session ID is stale, removing
```

```
user1@daw:~$ curl -v https://www.securesite.org 2>&1 | head -20
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0* Host www.securesite.org:443 was resolved.
* IPv6: (none)
* IPv4: 127.0.0.1
*   Trying 127.0.0.1:443...
* Connected to www.securesite.org (127.0.0.1) port 443
* ALPN: curl offers h2,http/1.1
} [5 bytes data]
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
} [512 bytes data]
*  CAfile: /etc/ssl/certs/ca-certificates.crt
*  CApath: /etc/ssl/certs
{ [5 bytes data]
* TLSv1.3 (IN), TLS handshake, Server hello (2):
{ [122 bytes data]
* TLSv1.3 (IN), TLS handshake, Encrypted Extensions (8):
{ [25 bytes data]
* TLSv1.3 (IN), TLS handshake, Certificate (11):
{ [1006 bytes data]
user1@daw:~$
```