Certificados, certificaciones, Entidades de Certificación

/vía: http://www.rinconastur.com/php/php21.php

Creación entidad certificadora

1. Creación de clave privada: `$ openssl genrsa -des3 -out CA_privada.key 2048` (poner contraseña)
2. Creación solicitud de certificado: `$ openssl req -new -key CA_privada.key -out CA_solicitud.csr`
    1. rellenar los campos solicitados
    2. en Organizational y Common Name usar el nombre que se mostrará (aka «certMate»)
3. Creación certificado: `$ openssl x509 -days 3650 -signkey CA_privada.key -in CA_solicitud.csr -req -out CA_certificado.crt` (pedirá la contraseña del CA_privado.key)

Creación certificado servidor

1. Creación de clave privada: `$ openssl genrsa -out Servidor_privada.key 2048`
2. Creación solicitud de certificado: `$ openssl req -new -key Servidor_privada.key -out Servidor_solicitud.scr`
    1. rellenar los campos solicitados
    2. Common Name usar la URL del servidor que queremos certificar
3. Creación certificado: `$ openssl x509 -days 3650 -CA CA_certificado.crt -CAkey CA_privada.key -set_serial 01 - in Servidor_solicitud.csr -req -out Sevidor_certificado.crt` (pedirá la contraseña del CA_privado.key)

Creación certificado cliente

1. Generación clave privada: `$ openssl genrsa -out Cliente_privada.key 2048`
2. Generación solitud de certificado: `$ openssl req -new -key Cliente_privada.key -out Cliente_solicitud.csr`
    1. en organizational y common name usar el nombre del usuario
3. Generación certificado: `$ openssl x509 -days 3650 -CA CA_certificado.crt -CAkey CA_privada.key - set_serial 02 -in Cliente_solicitud.csr -req -out Cliente_certificado.crt`
    1. los *set_serial* establecen un número de orden apra el control de los certificados de la CA
4. exportación a pkcs12 (para importar en el navegador): `$ openssl pkcs12 -export -out Cliente_certificado.pfx -inkey Cliente_privada.key -in Cliente_certificado.crt -certfile CA_certificado.crt`

formatos de los certificados

funciones PHP acceso certificados

```
<?php
```

```
/* empezamos leyendo el fichero que contiene el certificado y recogiendo su
contenido en una
variable que llamaremos $cert */
$f = fopen("juan_certificado.cer", "r");
 $cert = fread($f, 8192);
 fclose($f);
 /* la funcion openssl_x509_parse nos extrae los datos y los convierte en un array
*/
 $datos = openssl_x509_parse($cert,0);
?>
```

```
<?php
/* empezamos comprobando si la petición se hizo en modo seguro o no. La variable de
entorno HTTPS recoge es condición.
   Si estamos en modo seguro recogerá en la variable cert el valor transferido en
la peticion
   y en caso contrario leería el fichero de ejemplo que tenemos en el directorio
cursophp y al que se accede en modo no seguro */

if (getenv('HTTPS')=='on'){
   $cert=$_SERVER['SSL_CLIENT_CERT'];
 }else{
   $f = fopen("juan_certificado.cer", "r");
 $cert = fread($f, 8192);
 fclose($f);
 }

/* Extraemos los datos del certificado usando nombres cortos para los índices */

 $datos = openssl_x509_parse($cert,0);
?>
```

modificaciones apache

- en httpd.conf o ports.conf o apache2.conf:
  - añadir `Listen 443`
  - añadir o activar `LoadModule ssl_module modules/mod_ssl.so`
    - también puede activarse en `mods-enabled`
  - añadir `NameVirtualHost *:443`
- en definición de nuevo VirtualHost:

sslvirtualhost.conf

```
<VirtualHost *:443>
        SSLEngine On
        SSLOptions +StdEnvVars +ExportCertData
        SSLCertificateFile
/etc/apache2/certificados/Server_Certificado.crt
        SSLCertificateKeyFile
/etc/apache2/certificados/Server_Private.key
        SSLCACertificateFile
/etc/apache2/certificados/CertificadosRaiz.crt
        DocumentRoot /home/www/ssl.intranet.havasww.es/public
        ErrorLog /home/www/logs/ssl.intranet.havasww.es-error.log
```

```
        CustomLog /home/www/logs/ssl.intranet.havasww.es-access.log
common
        LogLevel warn
        <Directory "/home/www/ssl.intranet.havasww.es/public">
                AllowOverride All
                Options -Indexes
        </Directory>

        <Directory "/home/www/ssl.intranet.havasww.es/public/solocerts">
                SSLVerifyClient require

                SSLVerifyDepth 2
                # para DNIe, son CA subordinadas de otra CA

                SSLRequire ( %{SSL_CLIENT_V_REMAIN}>="0" )
                # certificado vigente
        </Directory>
</VirtualHost>
```

- SSLOptions +ExportCertData → nos permite acceder a datos del certificado
- SSLOptions +StdEnvVars → nos permite acceder a variables relacionadas (<?php print getenv('SSL_CLIENT_S_DN');?>
- SSLCACertificateFile <path> → nos permite incorporar certificados de CA (reales o ficticias)
- SSLVerifyClient require → obliga a seleccionar un certificado para acceder (en este caso, al directorio solocerts)
- SSLVerifyDepth <n> → por si la CA ha delegado en otras subs-CA la certificación (como en DNIe)
- SSLRequire (%{SSL_CLIENT_V_REMAIN}>=«0») → certificado no caducado
    - se pueden concatener varios requisitos con AND y OR