



ESPAM MFL

ESCUELA SUPERIOR POLITÉCNICA
AGROPECUARIA DE MANABÍ MANUEL FÉLIX LÓPEZ



Autenticación y autorización

Facilitador: Thomas Loor

CARRERA DE COMPUTACIÓN

Correo: pipos_rgt@hotmail.com

Autenticación

Tipo de autenticación web:

1. Básica
2. Digest
3. Certificados
4. Formularios

Autenticación y autorización

Autenticación: identificar al usuario, es decir, **¿Quién es el usuario?**

Lo que sabe:

- Login
- password.

Lo que tiene:

- Certificado digital , ya sea software o hardware como una Smart, card o un Token USB)
- Sistemas biométricos: huellas dactilares, voz, iris, etc.

Autorización: comprobar si el usuario tiene permiso para acceder al recurso. ¿El usuario autenticado puede ver el documento?

Autenticación y autorización

- Las directivas de configuración de Apache permiten definir autenticación de usuarios (básica y digest), así como comprobaciones de autorización en los recursos.
- Además de autenticación y autorización, Apache proporciona otras herramientas de control de acceso a los recursos.
- Para conseguir realizar autenticación, autorización o control de acceso, Apache utiliza diferentes cabeceras HTTP.
- Para realizar autenticación, autorización o control de acceso de Apache, siempre es necesario cargar el módulo **mod_authz_core**.
- Cuando no se supera la autenticación, autorización o control de acceso impuesto por las directivas de Apache, se obtiene una respuesta con código 401

Autenticación HTTP básica

- Es una autenticación basada en login y password.
- Los usuarios válidos se pueden almacenar en un fichero, con un login y un password por línea. El password se almacena en MD5, SHA-1 o con crypt para dificultar su obtención en claro.
- Para definir este fichero con los login y password válidos se puede definir un fichero con la aplicación htpasswd (en apache/bin)

```
C:\xampp\apache\bin> htpasswd -cb c:/xampp/apache/conf/passw thomas thomas123
```

Crea un nuevo fichero llamado passw con un usuario llamado thomas y con password thomas123.

Autenticación HTTP básica

Opciones de htpasswd	
htpasswd [opciones] passwdfile username password	
-b	Introduce la contraseña desde la línea de comandos en vez de ser de entrada obligada, no es recomendable pues la contraseña va en texto plano
-c	Crea el archivo passwdfile
-m	Usa MD5 como método de encriptación de las contraseñas. Este método está por defecto.
-s	Usa SHA como método de encriptación de las contraseñas
-p	Use texto plano para almacenar las contraseñas. Método muy inseguro.
-D	Borra al usuario especificado
passwdfile	Nombre del archivo
username	Nombre de usuario.
password	Contraseña.

Configurar Apache para autenticación HTTP básica

- Cargar el módulo ***mod_auth_basic*** para incluir las directivas.
- Se puede configurar en el fichero *httpd.conf* o en el *.htaccess* del directorio que requiera autenticación (siempre que esté habilitado con AllowOverride all o con AllowOverride AuthConfig).
- **Ejemplo:** para pedir autenticación en el directorio “Secreto” del DocumetRoot, se puede definir en el fichero httpd.conf de la siguiente manera:

```
<Directory "c:/xampp/htdocs/ProyectosSexto/">  
  AuthType Basic  
  AuthName "Restricted Files"  
  AuthBasicProvider file  
  AuthUserFile "c:\xampp\apache\conf\passwd"  
  Require valid-user  
</Directory>
```

Configurar Apache para autenticación HTTP básica

- El mismo ejemplo se puede conseguir definiendo el fichero .htaccess del directorio “Secreto” de la siguiente manera:

```
AuthType Basic
AuthName "Restricted Files"
AuthBasicProvider file
AuthUserFile C:\xampp\apache\conf\passwd
Require valid-user
ErrorDocument 401 /authrequired.php
```


Configurar Apache para autenticación HTTP básica

- **AuthType:** método de autenticación
- **AuthName:** dominio de la autenticación
 - Aparece en el diálogo en el que se pide login y password
 - Para los recursos con el mismo dominio, el browser ya conoce el login y password a enviar (no lo vuelve a pedir)
- **AuthBasicProvider:** proveedor de la información de los usuarios. Por defecto es file, pero puede especificarse una base de datos o un ldap.
- **AuthUserFile:** ruta en donde se encuentra el fichero (fuera del DocumentRoot)
- **Require:** condición que debe cumplirse. En este caso, un usuario válido cualquiera.

Autenticación HTTP básica

- También se puede pedir autenticación para un determinado fichero o ficheros. Por ejemplo, definimos en el .htaccess de un directorio lo siguiente:

```
<FilesMatch "^ (admin) \.php">  
    AuthType Basic  
    AuthName "Restricted Files"  
    AuthBasicProvider file  
    AuthUserFile C:\xampp\apache\conf\passwd  
    Require valid-user  
</FilesMatch>
```

En este ejemplo, sólo el fichero *admin.php* requerirían autenticación.

Autenticación HTTP básica

- En el siguiente ejemplo se define un **.htaccess** que los usuarios deban autenticarse para acceder a los contenidos del Document Root.
- Como excepción, se define un directorio público, en el que no se pide autenticación.

```
<Directory C:/xampp/htdocs>
  AuthType Basic
  AuthName "Restricted Files"
  AuthBasicProvider file
  AuthUserFile C:/xampp/apache/conf/passw
  Require valid-user
</Directory>

<Directory C:/xampp/htdocs/publico>
  Require all granted
</Directory>
```

Vulnerabilidades autenticación HTTP básica

- **Ataques de diccionario y ataques por fuerza bruta:** uso de herramientas para adivinar passwords: Hydra, Brutus, etc.
- **Escuchas (Eavesdropping):** el problema es que el login y el password se transmiten en claro (simplemente están codificados en Base 64). Para mitigar este riesgo, se recomienda utilizar el protocolo TLS/SSL.

Autenticación

Digest

Autenticación HTTP Digest

- Cargar el módulo ***mod_auth_digest*** para incluir las directivas.
- Es más segura que HTTP Básica porque la password no se transmite en claro, sino que se transmite un resumen (MD5 por defecto).

```
htdigest -c c:/xampp/apache/passdig DOCUMENTOS thomas  
A continuación pide el password para thomas  
En este ejemplo DOCUMENTOS es el dominio
```

```
En .htaccess del directorio /Secreto  
AuthType Digest  
AuthName "DOCUMENTOS"  
AuthBasicProvider file  
AuthUserFile C:\xampp\apache\passdig  
Require valid-user
```

Facilitador: **Thomas Loor**

Correo: pipos_rgt@hotmail.com

Control de acceso HTTP

- En el siguiente ejemplo se utiliza la directiva <RequireAll> para exigir dos condiciones: autenticación y procedencia de la petición.

```
<RequireAll>
  AuthType Basic
  AuthName "Restricted Files"
  AuthBasicProvider file
  AuthUserFile
  C:\xampp\apache\bin\passwords
  Require valid-user
  Require ip 155.23.1.10
</RequireAll>
```

Control de acceso HTTP

- En el ejemplo siguiente, se define que los ficheros con extensión .inc no sean accesibles. De otra manera el servidor apache se limitará a devolver su contenido, con el peligro de que pudieran tener información relevante.

No accesibles archivos .inc

```
<FilesMatch "\.inc$">  
    Require all denied  
</FilesMatch>
```

Accesibles solo archivos .php

```
<FilesMatch "\.php$">  
    Require all granted  
</FilesMatch>
```


Control de acceso HTTP

- No permitir el listado del contenido de un directorio:

```
Options -Indexes
```

- No permitir el acceso a ficheros .htaccess::

```
<Files ".ht*">  
    Require all denied  
</Files>
```

Autenticación

Formulario

Autenticación Web con Formularios


- Es la propia aplicación la que muestra un formulario de autenticación en el que pide el login y el password.
- Los usuarios válidos se encuentran almacenados en una base de datos.



REGISTRAR USUARIO

Usuario:

Password:



LOGIN

Autenticación Web con Formularios

- Por seguridad, se recomienda utilizar POST en lugar de GET. Así se evita almacenar la URL con el login y el password en el historial del navegador, o enviarlo por email, foro, etc.
- Al igual que en autenticación HTTP, las passwords de los usuarios válidos no se almacenan en claro en la base de datos, sino un resumen, obtenido con un algoritmo hash.
- La autenticación se basa en el mecanismo de sesión del usuario

Algoritmos hash

- Los algoritmos o funciones hash son conocidos también como funciones resumen o funciones digest.
- Los más conocidos son MD5 (128 bits), SHA-1 (160 bits) y SHA-2 (256 bits, 512 bits)

password: hola

MD5: 4d186321c1a7f0f354b297e8914ab240

SHA-1: 99800b85d3383e3a2fb45eb7d0066a4879a9dad0

Algoritmos hash

- Las funciones md5 (string), sha1 (string) y hash(algoritmo, string) de PHP calculan estos valores.

```
$resumen = md5 ("hola") ;  
echo $resumen;  
4d186321c1a7f0f354b297e8914ab240
```

```
$resumen = hash ("sha256", "hola");
```

En la base de datos no se almacenan las passwords, sino sus correspondientes hash. Si un atacante obtiene el hash no tiene la password; aunque hay herramientas que en algunos casos lo consiguen, sobre todo si el password es sencillo.

Algoritmos hash: uso de SALT

- Uso de un valor impredecible (SALT) para evitar que sea fácil encontrar la password a partir del valor hash.

```
$salt = time();  
$hash = md5 ("hola" . $salt) ;
```

- *Con este valor es mucho más complicado obtener el password.*
- Para almacenar passwords se **recomienda utilizar algoritmos hash diseñados para este propósito como bcrypt, pbkdf2 o scrypt.** Para lo cual en PHP se dispone de las funciones `password_hash` y `hash_pbkdf2`. *En todos los casos se utiliza un salt.*

Identificador de sesión

- Un usuario sólo debe autenticarse en la primera petición, el resto de peticiones ya se encuentra autenticado. Se genera una **variable de sesión**, que es único para cada usuario y se mantiene durante toda la sesión.
- Para saber cuáles son los datos de sesión de cada usuario, se asigna a cada usuario un identificador de sesión, que se almacena en el navegador del cliente en forma de cookie, y se transmite al servidor cada vez que realiza una petición.

Identificador de sesión

Al comprobar que el login y el password son correctos, creará una sesión de ese usuario que está autenticado.

```
session_start();  
// Comprobar en la base de datos el login y password recibido.  
if ( UsuarioCorrecto ) {  
  
    // En el array de sesión del usuario se recuerda que está autenticado  
    $_SESSION['autenticado'] = 'correcto';  
}else{  
    echo "Autenticación incorrecta" ;  
}
```

Identificador de sesión

Al solicitar una página de la aplicación que requiera autenticación:

```
session_start()

// Se recupera el identificador de sesión del usuario desde la cookie.
if (!isset($_SESSION['autenticado']) ||
    ($_SESSION['autenticado'] != "correcto"))
{
    header ("Location: login.php");
    exit;
}
```

Uso de Captchas

- Se utilizan con frecuencia en aplicaciones web para evitar que un programa (robot):
- Existen muchos tipos de captchas: relación entre imágenes y caracteres, problema matemático, preguntas con imágenes, etc.