Capítulo 18: Networking y FTP usando PHP

Funciones de FTP:

Uno de los problemas no solucionados por la mayoría de los lenguajes de scripting para la web es la transferencia de archivos entre diversos sites, usando NFS (Network file system) es posible crear zonas de un file-system que sean compartidas por más de un site, pero esto obliga a usar NFS, a tener una sobrecarga administrativa y además insertar varios problemas de seguridad delicados relacionados con NFS. El método más seguro y confiable para transferir archivos, PHP provee funciones nativas para usar FTP en forma directa desde un script php, las funciones más usadas son las siguientes:

ftp_handler=ftp_connect(host);

Ejemplo

$ftp=ftp_connect("100.100.100.100");

Si el puerto de FTP no es el default (21) se puede pasar como segundo parámetro de ftp_connect, la función devuelve un handler a ser usado por el resto de las funciones de FTP.
Devuelve true/false segun el éxito o fracaso de la conexión.

int=ftp_login(ftp_handler,string_user,string_password);

Realiza un login a la conexión FTP indicada por el handler. (devuelve true/false según el éxito o fracaso del login)

int=ftp_chdir(ftp_handler, string_directory);

Cambia al directorio especificado como segundo parámetro dentro de la conexión ftp abierta para el ftp_handler pasado.Devuelve true/false

int=ftp_get (ftp_handler, string_local_file, string_remote_file, int mode);

Transfiere el archivo string_remote_file a la maquina donde corre el script creandolo con el path indicado en string_local_file, el modo debe ser una constante que puede ser FTP_BINARY o FTP_ASCII

Ejemplo

$ret=ftp_get($ftp,"/temp/cosa.dat","cosa.dat",FTP_BINARY);

Notar que no deben usarse comillas en el cuarto parámetro por tratarse de una constante y no de un string.

int=ftp_fget (ftp_handler, file_handler, string_remote_file, int mode)

Idem anterior pero el segundo parámetro es un handler a un archivo abierto con fopen en donde se guardaran los datos leídos desde la conexión ftp.

int=ftp_put (ftp_handler, string_remote_file, string_local_file, int mode)

Transfiere un archivo local indicado por string_local_file mediante la conexión ftp abierta creando un archivo de nombre string_remote_file.

int=ftp_fput (ftp_handler, string_remote_file, file_handler, int mode)

Idem anterior pero el archivo a trasnferir se indica por el file_handler de un archivo abierto con fopen.

string=ftp_mkdir (ftp_handler, string_directory)

Crea un directorio en la conexión ftp abierta, devuelve el nombre del directorio creado o falso si no pudo crearlo.

int=ftp_rmdir (ftp_handler, string directory)

Elimina el directorio indicado, devuelve true/false.

int=ftp_cdup (ftp_handler)

Cambia al directorio padre del actual.

array=ftp_rawlist (ftp_handler, string_directory)

Devuelve un vector con la lista de archivos presentes en el directorio indicado de acuerdo al comando FTP ftp_list, el resultado es un vector donde cada elemento del vector es una línea devuelta por ftp_list.

int=ftp_size (ftp_handler, string_remote_file)

Devuelve el tamaño del archivo indicado.

ftp_quit(ftp_handler);

Se desconecta de la conexión ftp realizada con ftp_connect.

Networking en PHP:

PHP dispone de varias funciones de networking la más usada y la más flexible es fsockopen que permite conectarse a un socket en un host determinado por una dirección IP y un puerto, mediante esta funcion es posible conectarse a servidores HTPP, FTP, Telnet, IMAP, POP3 y otros protocolos.

Es de destacar que la funcionalidad de Networking de PHP es como CLIENTE, PHP no puede crear un socket con nombre y hacer un "listen" de conexiones a dicho port por lo que no puede funcionar como servidor.

La sintaxis de fsockopen es:

file_handler=fsockopen (string_hostname, int port , int errno , string_errstr , double timeout)

Los tres últimos parámetros son opcionales.
Hostname es el nombre o dirección IP del host al cual conectarse.
Port es el número de puerto al cual conectarse en el host.
errno debe ser una referencia a una variable en donde se guarda el número de error en caso de no poder conectarse.
errstr es una referencia a una variable en donde se guarda un mensaje de error en caso de no poder conectarse
El timeout es el tiempo máximo a esperar por la conexión en segundos.

Devuelve un file handler o false según pueda o no conectarse. El file hanlder devuelto puede luego usarse como un archivo normal usando fgets, fputs, feof, fclose, etc...

Ejemplo:

```
 1
 2 $fp = fsockopen ("www.php.net", 80, &$errno, &$errstr, 30);
 3 if (!$fp) {
 4     echo "$errstr ($errno)<br>\n";
 5 } else {
 6     fputs ($fp, "GET / HTTP/1.0\n\n");
 7     while (!feof($fp)) {
 8         echo fgets ($fp,128);
 9     }
10     fclose ($fp);
11 }
12
```

En este ejemplo abrimos el puerto 80 (Protocolo HTTP) de un host (www.php.net)
Luego ponemos en el socket un request de HTTP y entramos en un loop recuperando el contenido que devuelve el server. Es un mini simulador de browser HTTP.