Capítulo 14: Ejecución de programas externos y scripting.

Es posible desde PHP invocar a un programa externo de forma tal de utilizar algún script externo para obtener resultados que luego sean utilizados en un script php.

string=escapeshellcmd(string)

Escapea todos los caracteres que puedan resultar peligrosos en un comando que va a pasarse al shell. En varias ocasiones un cierto input ingresado por el usuario es pasado a un programa externo para cumplir una determinada función. Supongamos que el usuario ingresa un nombre y password y un script php debe pasarle esos datos a un programa externo para cierta validación. El comando podría ser:

$comando="/usr/bin/validator $user $password";

Pero que pasa si el usuario ingresa como password: "pepe;rm –rf /*"; Entonces el comando quedaría:

"/usr/bin/validator nombre pepe;rm –rf/*""

Y al ejecutarse además de hacer lo que el script debe hacer el shell eliminara todos los archivos del disco (ugh!), la función escapeshellcmd evita esto anulando todos los casos peligrosos para la llamada a un comando.

Las funciones de ejecución de comandos son:

string=exec(command, array, var);

Los dos últimos parámetros son opcionales. Ejecuta un comando pasándoselo al shell y devuelve la última línea devuelta por el comando en su standard otuput, si se pasa un nombre de vector como segundo parámetro devuelve cada línea de salida del comando en un elemento del vector. Si se pasa una variable como tercer parámetro devuelve alli el resultado del comando al shell.

passthru(command,var);

La variable es opcional y recibe el valor de retorno del comando, passthru ejecuta el comando y redirecciona su salida al browser en forma directa. Esto es útil por ejemplo para programas externos que generan una imagen o algo similar (antes hay que enviar el header correspondiente).

Uso de php como lenguaje de scripting:

PHP puede usarse tanto como un modulo del webserver como también como lenguaje de scripting interpretado desde la línea de comandos (en cuyo caso funciona con cualquier web server que soporte el protocolo CGi aun cuando no soporte php como modulo). Para usar php desde la línea de comandos basta con compilar una versión de php en la cual no se le pasa la opción de compilarse como modulo de Apache, de esta forma se generara un binario "php" que es el interprete php.

Un script php se puede escribir entonces de la forma:

```
#!/usr/bin/php
<?
  print("Hola mundo\n");
?>
```

La primera línea es la que se conoce en Unix como sheebang line e indica el path del interprete para el código que sigue a continuación, en nuestro caso el binario php reside en /usr/bin/ pero podría estar en otro lado.

Luego escribimos un script standard, manteniendo el código php entre <? y ?>, todo lo que no este entre estos símbolos pasa directamente al standard output del script.

El script se invoca desde la línea de comandos como un programa ejecutable normal (dándole permiso de ejecución), también puede usarse la línea de comandos para probar un script que no tiene la línea sheebang, por ejemplo un script normal que usamos en un web site:

/usr/bin/php /path/nombre.php

Esto es muchas veces útil para chequear cual es la salida de un script que no esta funcionando bien en un web server la salida por la línea de comando al standard output es útil para análisis y debug ya que no pasa por la interpretación del browser.

Es posible pasarle a un script php parámetros por la línea de comandos, por ejemplo:

script.php hola mundo

Estos parámetros serán recibidos por el script en el vector $argv, el elemento [0] de $argv es el nombre mismo del script por lo que los dos parámetros en este caso estarán en $argv[1] y $argv[2].

En algunos casos queremos probar desde la línea de comandos un script que recibe parámetros desde un formulario html, es decir usando método GET o POST. Para ello no sirve el formato anterior ya que el script desconoce la existencia de $argv. Lo que hay que hacer es poner los parámetros en la variable de ambiente "QUERY_STRING" de la forma: "nombre=valor&nombre2=valor2...." y luego invocar al script. Usando bash como shell esto es de la forma:

[path] $QUERY_STRING="hola=mundo&nombre=juan"
[path] $ export QUERY_STRING
[path] $ /usr/bin/php /path/script.php

Automáticamente el interprete de php se encarga de parsear la variable de ambiente QUERY_STRING y convertirla en variables de php por lo que el resultado del script será el mismo que el que produciría si lo llamáramos desde un form html con esas variables seteadas.