

Capítulo 10. Algunas medidas de seguridad

10.1. Permisos en archivos y carpetas

Esta sección le concierne únicamente a aquellas personas que utilizan PHP-Nuke bajo Linux/Unix (esto es una realidad para la gran mayoría de los sitios PHP-Nuke hospedados por proveedores en Internet, y también a aquellos que utilicen localmente Linux).

Burzi ha mencionado que a los directorios o carpetas se les deberá asignar el modo 777, mientras que a los archivos se les deberá asignar el modo 666, sin embargo podemos lograr que nuestro PHP-Nuke funcione correctamente trabajando con permisos mejor restringidos, como se muestra a continuación:

- config.php (666)
- backend.php (666)
- ultramode.txt (666)
- Todos los directorios (755)
- Otros archivos (644)

Los archivos config.php, backend.php y ultramode.txt deberán tener permisos de escritura, para así:

- Poder editar el archivo config.php con el cual podremos modificar las preferencias de nuestro sitio.
- Para el backend y ultramode del lado del servidor, ya que de una manera automática escribiremos en ellos, modificando los títulos y abstractos de las noticias.

Sin embargo existe algo que debemos tomar en cuenta: si utilizamos módulos que suben archivos en algunos directorios, se tendrían que incrementar sus permisos. Consideremos como ejemplo el módulo IndyNews, un módulo no estándar que hace posible insertar archivos e imágenes en artículos. La estructura del módulo es la siguiente:

modules/indynews/media

Dentro de la carpeta "indynews", los permisos de la carpeta "media" tendrían que ser 777, debido a un problema de acceso, los permisos 777 tendrían que ser aplicados en toda carpeta por debajo de "modules". Por ésta razón, todo lo que resida en "modules" tendrá permisos 777 y causaría una gran vulnerabilidad. Una solución es mover afuera de la carpeta "modules" la carpeta que recibe los archivos, inclusive moverla a la raíz, cambiando dentro del módulo cualquier referencia a esta.

Al hacer esto solamente una carpeta tendrá permisos 777.

10.2. Cookies - expiración y configuración

PHP-Nuke hace bastante uso de las llamadas cookies, ya sea para la autenticación de usuarios, o para autenticación de administradores. Las cookies son archivos de texto que se almacenan en las computadoras del usuario y contienen información considerable, la cual es leída por el servidor cuando se entra a algún sitio. En el caso de PHP-Nuke la información que ahí se almacena pertenece al usuario, el tema que haya escogido y el idioma.

La cookie también es el instrumento que evita tener que rescribir nuestra contraseña cada vez que nos conectamos al sitio. De esta manera, cuando accedemos a un sitio PHP-Nuke, la cookie se encarga de la conexión con el sitio.

El problema es que si esta cookie no tiene un tiempo de expiración lo suficientemente corto, alguien podrá robarla y podrá acceder al sitio ya sea como usuario o como administrador.

Esto es posible por una serie de razones:

1. La cookie de PHP-Nuke tiene un tiempo de vida cercano al infinito (31536000 segundos)
2. Microsoft Internet Explorer (desafortunadamente el navegador más utilizado) tiene ciertas vulnerabilidades que permiten la ejecución de scripts o código malicioso, con el objetivo de robar la cookie del usuario.
3. PHP-Nuke no ha logrado tener éxito en filtrar todos los scripts maliciosos (y para empeorar la situación, Internet Explorer es tan tonto que corrige aquellos scripts cuya sintaxis este mal, para así poderlos leer).

Veamos un ejemplo en concreto de cómo un script kiddie (aquellos que alegan ser hackers pero no lo son en realidad) puede intentar obtener permisos de administrador en nuestro sitio:

1. El script kiddie inserta un script que supuestamente contiene noticias:

```
< vb script dame la cookie a mí y envíala al servidor xyz>
```

El cual no está filtrado por la función check_words() de PHP-Nuke.

2. El administrador de PHP-Nuke abre la página con Internet Explorer (Esta intrusión no funciona si utilizas Mozilla o aún mejor, si utilizas cualquier navegador de Linux). La lista de noticias que esperan ser aprobadas para su publicación es vista por el administrador. Cuando se dirige a ver las entradas, Internet Explorer tontamente corrige el vbscript de la siguiente manera:

```
<vbscript>(los comandos del script kiddie van aquí)
```

De esta manera se logra interpretar la sintaxis incorrecta, con lo cual se manda la cookie al script kiddie.

3. El script kiddie coloca la cookie obtenida entre las suyas, se conecta al sitio...¡Y es reconocido como el administrador!

¿Pero cómo nos podemos proteger de este tipo de ataques?

Existen algunas soluciones que incrementan nuestra seguridad en el área de administración:

1. Primero que nada, DETENGA el uso de Internet Explorer como navegador, y ceda su lugar a Mozilla. Este navegador soporta todos los sitios en Internet de una manera óptima sin estar plagada de todas vulnerabilidades de Microsoft. Si utiliza Linux en su lugar, no tendrá problemas de este tipo...

- a. En caso de que usted elija seguir utilizando Internet Explorer, debería al menos descargar los parches correspondientes de .

2. Desactive, donde sea posible, la inserción de etiquetas HTML (Por ejemplo, en el foro de Splatt.it).

3. Reduzca el tiempo de vida de sus cookies. Si por ejemplo configuramos el tiempo de vida de una cookie por dos horas, el script kiddie se verá forzado a usar esa cookie dentro de ese período de tiempo, esto limita mucho el que pueda actuar a tiempo.

Si dejamos el tiempo de vida de la cookie a su valor por defecto, el script kiddie puede utilizar esa cookie hasta por un mes después de haber sido robada.

¿Cómo configurar el tiempo de vida de la cookie del administrador?

La cookie se puede configurar en el archivo includes/auth.php y la función para modificarlo es como sigue:

```
if ((isset($aid)) && (isset($pwd)) && ($op == "login")) {
    if($aid != "" AND $pwd!="") {
        $pwd = md5($pwd);
        $result=sql_query("select pwd, admlanguage from "$prefix."_authors where
aid='$aid'", $dbi);
        list($pass, $admlanguage)=sql_fetch_row($result, $dbi);
        if($pass == $pwd) {
            $admin = base64_encode("$aid:$pwd:$admlanguage");
            setcookie("admin", "$admin",time()+7200);
            unset($op);
        }
    }
}
```

Hemos modificado el tiempo de vida de la segunda cookie de 259000 segundos (un mes) a 7200 segundos (dos horas). Como puede observarse, hemos reducido el período de tiempo en el que el script kiddie puede actuar, de un mes a dos horas.

4. Un filtro de etiquetas mucho más efectivo se encuentra en la fase de estudio, aunque por el momento, las propuestas han dado respuesta definitiva al problema. Las etiquetas permitidas estan definidas en el archivo config.php en la variable \$AllowableHTML, estas etiquetas son válidas para los comentarios y para la insercion de noticias en la función check_html() que por el momento deja pasar algunas etiquetas.

Todas estas acciones y la correcta configuración de los permisos que fueron mostrados en la sección 10.1 nos debe garantizar una buena seguridad a nuestro sitio. También es importante seguir muy de cerca todos los avisos de seguridad que se presenten para PHP-Nuke.