Προγραμματισμός Διαδικτύου

Δρ. Μηνάς Δασυγένης
mdasygenis@uowm.gr

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

Προγραμματισμός Διαδικτύου

Ασφάλεια στους Ιστοχώρους

Προηγμένη PHP ασφάλεια

Χρησιμοποιήθηκε υλικό από παρουσίαση του Ilia Alshanetsky

Τι είναι ασφάλεια;

Η ασφάλεια είναι μια μέτρηση, δεν είναι χαρακτηριστικό.

- Είναι επίσης ένα αυξανόμενο πρόβλημα που απαιτεί μια συνεχώς εξελισσόμενη λύση.

- Ένα καλό μέτρο για την ασφαλή εφαρμογή είναι η ικανότητα να προβλέψει και να αποτρέψει τα μελλοντικά προβλήματα ασφάλειας, πριν κάποιος επινοήσει μια εκμετάλλευση.

- Όσον αφορά το σχεδιασμό της εφαρμογής, η ασφάλεια πρέπει να λαμβάνεται υπόψη ανά πάσα στιγμή. Η αρχική προδιαγραφή, η εκτέλεση, η δοκιμή ακόμη και η συντήρηση.

Τι είναι ασφάλεια;

- Η ασφάλεια πρέπει να είναι ισορροπημένη με τα έξοδα.

Είναι εύκολο και σχετικά φθηνό να εξασφαλιστεί επαρκές επίπεδο ασφάλειας για τις περισσότερες εφαρμογές. Ωστόσο, εάν η ανάγκες ασφάλεια σας είναι πολύ απαιτητικές, επειδή προστατεύετε τις πληροφορίες που είναι πολύτιμες, τότε πρέπει να επιτύχετε ένα υψηλότερο επίπεδο ασφάλειας με αυξημένο κόστος. Η δαπάνη αυτή πρέπει να συμπεριληφθεί στον προϋπολογισμό του έργου.

- Η ασφάλεια πρέπει να είναι ισορροπημένη με τη χρηστικότητα.

Δεν είναι ασυνήθιστο ότι τα μέτρα που λήφθηκαν για την αύξηση της ασφάλειας μιας διαδικτυακής εφαρμογής μπορούν επίσης να μειώσουν τη χρηστικότητα. Οι κωδικοί πρόσβασης, τα χρονικά όρια της περιόδου σύνδεσης και ο έλεγχος πρόσβασης δημιουργούν όλα εμπόδια για ένα νόμιμο χρήστη. Μερικές φορές είναι απαραίτητες για την παροχή επαρκούς ασφάλειας, αλλά δεν υπάρχει μια λύση που να είναι κατάλληλη για κάθε εφαρμογή. Είναι σοφό να προσέχετε τους νόμιμους χρήστες σας καθώς εφαρμόζετε μέτρα ασφαλείας.

PHP και ασφάλεια

- Η PHP συνεχίζει να αναπτύσσεται ως γλώσσα, προχωρώντας σε επιχειρηματικές και εταιρικές αγορές.

- Συνεπώς, οι εφαρμογές PHP συχνά καταλήγουν να εργάζονται με ευαίσθητα δεδομένα.

- Η μη εξουσιοδοτημένη πρόσβαση στα δεδομένα αυτά είναι απαράδεκτη.

- Για την αποφυγή προβλημάτων απαιτείται ασφαλής σχεδίαση.

Βασικά βήματα

- Εξετάστε τους παράνομους χρήστες της εφαρμογής σας.
- Εκπαιδεύστε τον εαυτό σας. (ελέγξτε http://phpsec.org/library/)
- Εάν δεν υπάρχει τίποτα άλλο, ΦΙΛΤΡΑΡΕΤΕ ΟΛΑ ΤΑ ΕΞΩΤΕΡΙΚΑ ΔΕΔΟΜΕΝΑ.

Επικύρωση εισόδου

- Μία από τις βασικές έννοιες που πρέπει να δεχτείτε είναι ότι οι είσοδοι των χρηστών είναι αναξιόπιστες και δεν πρέπει να εμπιστεύονται.
- Μερική απώλεια στη μετάδοση μεταξύ διακομιστή και πελάτη.
- Διαφθείρεται από κάποια διαδικασία μεταξύ των δύο.
- Τροποποιείται από τον χρήστη με απροσδόκητο τρόπο.
- Εσκεμμένη προσπάθεια να αποκτήσετε μη εξουσιοδοτημένη πρόσβαση ή να συντρίψετε την εφαρμογή.
- Για το λόγο αυτό, είναι απολύτως απαραίτητο να επικυρώσετε κάθε είσοδο χρήστη πριν από τη χρήση.

Φιλτράρισμα δεδομένων

Βεβαιωθείτε ότι δεν είναι δυνατή η παράκαμψη του φιλτραρίσματος δεδομένων.

Βεβαιωθείτε ότι τα μη έγκυρα δεδομένα δεν μπορούν να συγχέονται με έγκυρα δεδομένα.

Προσδιορίστε την προέλευση των δεδομένων.

Πρόσβαση σε δεδομένα εισόδου

Από την PHP 4.1, υπάρχει μια σειρά μεταβλητών (super-globals) που προσφέρουν πολύ απλή πρόσβαση στα δεδομένα εισόδου.

$ _GET - δεδομένα από αιτήσεις λήψης.
$ _POST – δημοσιεύει δεδομένα αιτήματος.
$ _COOKIE - πληροφορίες cookie.
$ _FILES - μεταφορτωμένα δεδομένα αρχείου.
$ _SERVER - δεδομένα διακομιστή
$ _ENV - μεταβλητές περιβάλλοντος
$ _REQUEST - συνδυασμός GET / POST / COOKIE

Register Globals

Αναμφισβήτητα η πιο κοινή πηγή ευπάθειας στις εφαρμογές PHP.

Οποιεσδήποτε παράμετροι εισόδου μεταφράζονται σε μεταβλητές.

```
?foo = bar >> $foo = "bar";
```

Δεν υπάρχει τρόπος να προσδιοριστεί η πηγή εισόδου.

Οι προεπιλεγμένες πηγές όπως τα κούκις (cookies) μπορούν να αντικαταστήσουν τις τιμές GET.

Οι μη-προετοιμασμένες μεταβλητές μπορούν να "εγχυθούν" μέσω εισόδων χρηστών.

Register Globals

```
if (authenticated_user()) {
   $authorized = true;
}
if ($authorized) {
   include '/highly/sensitive/data.php';
}
```

Επειδή το $authorized δεν έχει αρχικοποιηθεί εάν αποτύχει ο έλεγχος ταυτότητας χρήστη, ένας εισβολέας θα μπορούσε να αποκτήσει πρόσβαση σε προνομιακά δεδομένα απλώς μέσω της μετάδοσης της αξίας μέσω του GET.

http://example.com/script.php?authorized=1

Λύσεις για τα Register Globals

Απενεργοποιήστε τα register_globals στο PHP.ini.

Πραγματοποιήθηκε ήδη από προεπιλογή από την PHP 4.2.0.

Ο κώδικας με error_reporting (αναφορά σφάλματος) είναι ρυθμισμένος σε E_ALL.

Σας επιτρέπει να βλέπετε προειδοποιήσεις σχετικά με τη χρήση μη-αρχικοποιημένων μεταβλητών.

Πληκτρολογήστε ευαίσθητες συνθήκες επικύρωσης.

Επειδή η είσοδος είναι πάντα μια συμβολοσειρά, πληκτρολογώντας ευαίσθητη σύγκριση με ένα Boolean ή έναν ακέραιο αριθμό θα αποτύχει πάντα.

```
if ($authorized === TRUE) {
```

Κρυμμένα προβλήματα των Register Globals

```
$var[] = "123";
foreach ($var as $entry) {
   make_admin($entry);
}
```

```
script.php?var[]=1&var[]=2
```

Ο παραπάνω σύνδεσμος θα επιτρέψει στον εισβολέα να εμβάλει δύο τιμές στη μεταβλητή $var. Ακόμα χειρότερα η PHP δεν παρέχει εργαλεία για την ανίχνευση τέτοιων ενέσεων.

$_REQUEST

Το super-global $ _REQUEST συγχωνεύει δεδομένα από διαφορετικές μεθόδους εισόδου, όπως το register_globals είναι ευάλωτο στις συγκρούσεις τιμών.

```
PHP.ini: variables_order = GPCS
```

```
echo $_GET['id']; // 1
echo $_COOKIE['id']; // 2
echo $_REQUEST['id']; // 2
```

$_SERVER

Παρόλο που το $_SERVER super-global έχει συμπληρωθεί με βάση τα δεδομένα που παρέχονται από τον διαδικτυακό διακομιστή (web-server), δεν πρέπει να εμπιστεύεται.

Ο χρήστης μπορεί να εισάγει δεδομένα μέσω κεφαλίδων

```
Host: <script> ...
```

Ορισμένες παράμετροι περιέχουν δεδομένα με βάση την είσοδο χρήστη.

```
REQUEST_URI, PATH_INFO, QUERY_STRING
```

Μπορεί να είναι απομιμήσεις.
Παραπλανητική διεύθυνση IP μέσω της χρήσης ανώνυμων πληρεξουσίων.

Επικύρωση αριθμητικής τιμής

Όλα τα δεδομένα που διαβιβάζονται στην PHP (GET / POST / COOKIE) καταλήγουν να είναι μια συμβολοσειρά. Χρησιμοποιώντας συμβολοσειρές όπου απαιτούνται ακέραιοι αριθμοί είναι όχι μόνο αναποτελεσματικά αλλά και επικίνδυνα.

```
// integer validation
if (!empty($_GET['id'])) {
        $id = (int) $_GET['id'];
} else
        $id = 0;
// floating point number validation
if (!empty($_GET['price'])) {
        $price = (float) $_GET['price'];
} else
        $price = 0;
```

- Το casting είναι ένας απλός και πολύ αποτελεσματικός τρόπος διασφάλισης ότι οι μεταβλητές περιέχουν αριθμητικές τιμές.

Επικύρωση συμβολοσειρών

Η PHP έρχεται με την επέκταση ctype που προσφέρει έναν πολύ γρήγορο μηχανισμό για την επικύρωση του περιεχομένου των συμβολοσειρών.

```
if (!ctype_alnum($_GET['login'])) {
        echo "Only A-Za-z0-9 are allowed.";
}
if (!ctype_alpha($_GET['captcha'])) {
        echo "Only A-Za-z are allowed.";
}
if (!ctype_xdigit($_GET['color'])) {
        echo "Only hexadecimal values are allowed";
}
```

Επικύρωση διαδρομής

- Οι τιμές που μεταφέρονται σε εφαρμογές PHP χρησιμοποιούνται συχνά για να καθορίσουν ποιο αρχείο θα ανοίξει. Και αυτό πρέπει να επικυρωθεί για να αποφευχθεί η αυθαίρετη πρόσβαση στο αρχείο.

```
http://example.com/script.php?path=../../etc/passwd
```

```
<?php
$fp = fopen("/home/dir/{$_GET['path']}", "r");
?>
```

ΠΡΟΣΟΧΗ: εκμετάλλευση προβλήματος ασφάλειας

Επικύρωση διαδρομής

Η PHP περιλαμβάνει μια συνάρτηση basename() που θα επεξεργαστεί μια διαδρομή και θα αφαιρέσει όλα τα άλλα από το τελευταίο στοιχείο της διαδρομής, συνήθως ένα όνομα αρχείου.

```
<?php
$_GET['path'] = basename($_GET['path']);

// only open a file if it exists.
if (file_exists("/home/dir/{$_GET['path']}")) {
   $fp = fopen("/home/dir/{$_GET['path']}", "r");
}
?>
```

Καλύτερη επικύρωση διαδρομής

Μια ακόμη καλύτερη λύση θα αποκρύψει τα ονόματα των αρχείων από το χρήστη όλα μαζί και θα συνεργαστεί με μια white-list (λευκή λίστα) αποδεκτών τιμών.

```
// make white-list of templates
$tmpl = array();
foreach(glob("templates/*.tmpl") as $v) {
        $tmpl[md5($v)] = $v;
}
if (isset($tmpl[$_GET['path']]))
        $fp = fopen($tmpl[$_GET['path']], "r");
```

```
http://example.com/script.php?path=57fb06d7...
```

Η μέθοδος αποστολής
(Dispach method)

- Ενιαίο σενάριο PHP διαθέσιμο απευθείας από τον ιστό.
- Όλα τα υπόλοιπα είναι μια ενότητα που περιλαμβάνεται με τη συμπερίληψη ή την απαίτηση.
- Επιτρέπει:

  - Εφαρμόζει ορισμένα συνολικά μέτρα ασφαλείας στην κορυφή του dispatch.php .

  - Βλέπει εύκολα ότι το φιλτράρισμα δεδομένων πραγματοποιείται όταν είναι απαραίτητο.

http://example.org/dispatch.php?task=print_form

Η μέθοδος αποστολής

(Dispach method)

```
<?php
/* Global security measures */
switch ($_GET['task'])
{     case 'print_form':
          include '/inc/presentation/form.inc';
          break;
      case 'process_form':
          $form_valid = false;
          include '/inc/logic/process.inc';
          if ($form_valid)
          {               include '/inc/presentation/end.inc';
}
          else
          { include '/inc/presentation/form.inc'; }    break;
      default:
          include '/inc/presentation/index.inc';
          break;}?>
```

Η μέθοδος συμπερίληψης
(Include method)

```
<?php
switch ($_POST['form'])
{
    case 'login':
        $allowed = array();
        $allowed[] = 'form';
        $allowed[] = 'username';
        $allowed[] = 'password';

        $sent = array_keys($_POST);
        if ($allowed == $sent)
        {
            include '/inc/logic/process.inc';
        }
        break;
}
?>
```

Έλεγχος για μεταβλητές που στέλνονται.

magic_quotes_gpc

- Η PHP προσπαθεί να σας προστατεύσει από επιθέσεις, ξεφεύγοντας αυτομάτως από όλους τους ειδικούς χαρακτήρες μέσα στην είσοδο του χρήστη. (',', \, \ 0 (NULL))

- Αναστέλλει την επεξεργασία εισόδου.
  - Μπορούμε να το κάνουμε καλύτερα χρησιμοποιώντας χύτευση για ακέραιους αριθμούς.
  - Απαιτεί 2 φορές μνήμη για κάθε στοιχείο εισόδου.

- Δεν μπορεί πάντα να είναι διαθέσιμη.
  - Θα μπορούσε να απενεργοποιηθεί στη διαμόρφωση PHP.

- Γενική λύση.
  - Άλλοι χαρακτήρες μπορεί να απαιτούν διαφυγή.

Κανονικοποίηση των magic quotes

```
if (get_magic_quotes_gpc()) { // check magic_quotes_gpc state
        function strip_quotes(&$var) {
                if (is_array($var)
                        array_walk($var, 'strip_quotes');
                else
                        $var = stripslashes($var);
        }

          // Handle GPC
        foreach (array('GET','POST','COOKIE') as $v)
                if (!empty(${"_".$v}))
                        array_walk(${"_".$v}, 'strip_quotes');

         // Original file names may contain escaped data as well
        if (!empty($_FILES))
                foreach ($_FILES as $k => $v) {
                  $_FILES[$k]['name'] = stripslashes($v['name']);
}
```

Με τις συναρτήσεις strip_quotes() {user-defined} ,stripslashes() απομακρύνουμε τα escape strings π.χ. \" γίνεται "

Αξιοποίηση κώδικα της προηγούμενης διαφάνειας

- Ενώ ο κώδικας στην προηγούμενη διαφάνεια λειτουργεί, μπορεί να εκμεταλλευτεί ασήμαντα, λόγω της χρήσης αναδρομικών λειτουργιών!

```
<?php
$qry = str_repeat("[]", 1024);
$url = "http://site.com/script.php?a{$qry}=1";
file_get_contents($url);

// run up in memory usage, followed by a prompt crash
?>
```

Πιο αξιόπιστη και ταχύτερη λύση

```
if (get_magic_quotes_gpc()) {
    $in = array(&$_GET, &$_POST, &$_COOKIE);
    while (list($k,$v) = each($in)) {
        foreach ($v as $key => $val) {
            if (!is_array($val)) {
                $in[$k][$key] = stripslashes($val);
                continue;
            }
            $in[] =& $in[$k][$key];
        }
    }
    unset($in);
}
```

Παραδείγματα φιλτραρίσματος: έγκυρο ηλεκτρονικό ταχυδρομείο (e-mail)

```
<?php

$clean = array();

$email_pattern = '/^[^@\s<&>]+@([-a-z0-9]+\.)+[a-
z]{2,}$/i';

if (preg_match($email_pattern, $_POST['email']))
{
    $clean['email'] = $_POST['email'];
}

?>
```

\s: whitespace (κενά, tabs, και διαχωριστικά)
\d: digits (ψηφία)
\w: word characters (λεκτικοί χαακτήρες) (γράμματα, ψηφία, και κάτω παύλες)

Παραδείγματα φιλτραρίσματος: έγκυρες τιμές

```
<?php

$clean = array();

switch ($_POST['color'])
{
    case 'red':
    case 'green':
    case 'blue':
        $clean['color'] = $_POST['color'];
        break;
}

?>
```

Παραδείγματα φιλτραρίσματος: έγκυροι ακέραιοι

```
<?php

$clean = array();

if ($_POST['num'] == strval(intval($_POST['num'])))
{
    $clean['num'] = $_POST['num'];
}

?>
```

intval — Παίρνει την ακέραια τιμή της μεταβλητής
strval — Παίρνει την τιμή της συμβολοσειράς της μεταβλητής

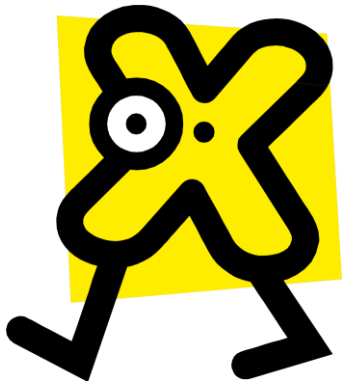
Παραδείγματα φιλτραρίσματος: πραγματικοί

```
<?php

$clean = array();

if ($_POST['num'] ==
strval(floatval($_POST['num'])))
{
    $clean['num'] = $_POST['num'];
}

?>
```

XSS

- Το Cross Scripting (XSS) είναι μια κατάσταση όπου ο εισβολέας εισάγει κώδικα HTML, ο οποίος στη συνέχεια εμφανίζεται στη σελίδα χωρίς περαιτέρω επικύρωση.

  - Μπορεί να οδηγήσει σε αμηχανία.
  - Ανάληψη συνόδου.
  - Κλοπή κωδικού πρόσβασης.
  - Παρακολούθηση χρηστών από τρίτα μέρη.



Πρόληψη του XSS

- Η πρόληψη του XSS είναι τόσο απλή όσο το φιλτράρισμα δεδομένων εισόδου μέσω ενός από τα παρακάτω:

- htmlspecialchars ()
  - Κωδικοποιεί ',', <,> &
- htmlentities ()
  - Μετατρέψτε οτιδήποτε υπάρχει οντότητα HTML.
- strip_tags ()
  - Αφαιρεί οτιδήποτε μοιάζει με ετικέτα HTML.

Πρόληψη του XSS

```
$str = strip_tags($_POST['message']);
// encode any foreign & special chars
$str = htmlentities($str);
// maintain new lines, by converting them to <br />
echo nl2br($str);

// strip tags can be told to "keep" certain tags
$str = strip_tags($_POST['message'], '<b><p><i><u>');
$str = htmlentities($str);
echo nl2br($str);
```

- Τα δικαιώματα ετικετών στα strip_tags () είναι επικίνδυνα, επειδή τα χαρακτηριστικά αυτών των ετικετών δεν επικυρώνονται με κανέναν τρόπο.

Description — Report a bug

string **nl2br** (string *$string* [, bool *$is_xhtml* = *true*])

Returns *string* with '
' or '
' inserted before all newlines (\r\n, \n\r, \n and \r).

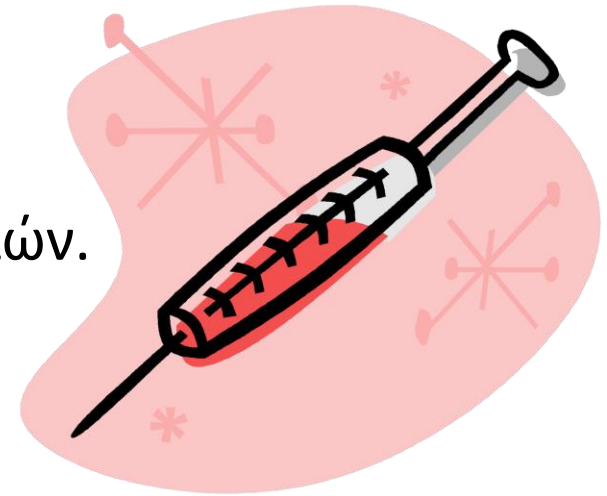
Προβλήματα δικαιωμάτων ετικετών

```
<b style="font-size: 500px">
TAKE UP ENTIRE SCREEN
</b>

<u onmouseover="alert('JavaScript is allowed');">
<b style="font-size: 500px">Lot's of text</b>
</u>

<p style="background: url(http://tracker.com/image.gif)">
Let's track users
</p>
```

SQL Injection (Ένεση SQL)

- Η ένεση SQL είναι παρόμοια με την XSS, καθώς χρησιμοποιούνται μη επικυρωμένα δεδομένα. Αλλά σε αυτή την περίπτωση τα δεδομένα αυτά διαβιβάζονται στη βάση δεδομένων.

- Εκτέλεση αυθαίρετου ερωτήματος.
  - Κατάργηση δεδομένων.
  - Τροποποίηση των υφιστάμενων τιμών.
  - Άρνηση εξυπηρέτησης.
  - Αυθεντική έγχυση δεδομένων.



SQL Escaping (Διαφυγή SQL)

- Εάν η επέκταση διεπαφής της βάσης δεδομένων προσφέρει αφιερωμένες λειτουργίες που διαφεύγουν, ΧΡΗΣΙΜΟΠΟΙΗΣΤΕ ΤΙΣ!

- MySQL

mysql_escape_string()

mysql_real_escape_string()

- PostgreSQL

pg_escape_string()

pg_escape_bytea()

SQLite

sqlite_escape_string()

SQL Escaping στην πράξη (Διαφυγή SQL)

```
// undo magic_quotes_gpc to avoid double escaping
if (get_magic_quotes_gpc()) {
  $_GET['name'] = stripslashes($_GET['name'];
  $_POST['binary'] = stripslashes($_GET['binary']);
}


$name = pg_escape_string($_GET['name']);
$binary = pg_escape_bytea($_POST['binary']);


pg_query($db, "INSERT INTO tbl (name,image)
            VALUES('{$name}', '{$image}')");
```

Escaping Shortfall
(Διαφυγή ελλείμματος)

- Όταν πεπερασμένοι ακέραιοι αριθμοί μεταβιβάζονται σε ερωτήματα SQL, οι λειτουργίες που διαφεύγουν δεν θα σας σώσουν, αφού δεν υπάρχουν ειδικοί χαρακτήρες για να ξεφύγουν.

```
http://example.com/db.php?id=0;DELETE%20FROM%20users
```

```
<?php
$id = sqlite_escape_string($_GET['id']);
// $id is still 0;DELETE FROM users

sqlite_query($db,
       "SELECT * FROM users WHERE id={$id}");
// Bye Bye user data...
?>
```

Προετοιμασμένες δηλώσεις

- Οι προετοιμασμένες δηλώσεις είναι ένας μηχανισμός που εξασφαλίζει και βελτιστοποιεί την εκτέλεση επανειλημμένων ερωτημάτων.

- Λειτουργεί κάνοντας SQL “compile" (μεταγλώττιση) στο ερώτημα και στη συνέχεια να υποκαταστήσει στις μεταβαλλόμενες τιμές για κάθε εκτέλεση.

- Αυξημένη απόδοση, 1 compile έναντι 1 ανά ερώτημα.

- Καλύτερη ασφάλεια, τα δεδομένα που είναι "type set" δεν θα αξιολογηθούν ποτέ ως ξεχωριστό ερώτημα.

- Υποστηρίζεται από τα περισσότερα συστήματα βάσεων δεδομένων.
  - Οι χρήστες της MySQL θα πρέπει να χρησιμοποιούν την έκδοση 4.1 ή νεότερη.
  - Η επέκταση SQLite επίσης δεν το υποστηρίζει αυτό.

Προκατασκευασμένες δηλώσεις

```
<?php
$data = "Here is some text to index";

pg_query($db, "PREPARE my_stmt (text) AS
                INSERT INTO search_idx (word) VALUES($1)");
foreach (explode(" ", $data) as $word) {
        // no is escaping needed
        pg_query($db, "EXECUTE my_stmt({$word})");
}

// de-allocte the prepared statement
pg_query($sb, "DEALLOCATE my_stmt");
?>
```

- Εάν δεν έχουν καταργηθεί ρητά, οι προκατασκευασμένες δηλώσεις "παραμείνουν ζωντανές" ανάμεσα στις επίμονες συνδέσεις.

Αποθηκευμένες διαδικασίες

- Οι αποθηκευμένες διαδικασίες έχουν νόημα για επαγγελματικές εφαρμογές (IE επιχειρηματικού βαθμού), στις οποίες:
- Θέλετε να επιτρέψετε στον μηχανικό της βάσης δεδομένων σας να βελτιστοποιήσει τα ερωτήματα για την απόδοση.
- Θέλετε να αφαιρέσετε την πολυπλοκότητα των ερωτημάτων σε απλά API's.
- Θέλετε τη λογική σας κατανεμημένη, επειδή μερικά από αυτά που συμβαίνουν στη βάση δεδομένων μπορεί να είναι πνευματική ιδιοκτησία που δεν θέλετε να εκθέσετε σε άλλα μέρη.
- Θέλετε τη λογική σας κατανεμημένη, γιατί αυτή είναι η φύση του κατανεμημένου, n-tier computing.
- Ίσως θέλετε ο μηχανικός βάσης δεδομένων ή ο DBA να τροποποιήσουν το σχήμα χωρίς να τροποποιήσουν τον κώδικα εφαρμογής (αποθηκευμένα procs, λόγω της παροχής API, παρέχουν ένα επίπεδο αφαίρεσης)

Αναφορά σφαλμάτων

- Από προεπιλογή, η PHP θα εκτυπώσει όλα τα σφάλματα στην οθόνη, θα καταπλήξει τους χρήστες σας και σε ορισμένες περιπτώσεις θα αποκαλύψει προνομιακές πληροφορίες.
- Διαδρομές αρχείων.
- Μη-αρχικοποιημένες μεταβλητές.
- Ευαίσθητες λειτουργίες επιχειρημάτων, όπως κωδικοί πρόσβασης.
- Ταυτόχρονα, η απενεργοποίηση της αναφοράς σφαλμάτων θα καθιστούσε την παρακολούθηση σφαλμάτων σχεδόν αδύνατη.

Λύση σφαλμάτων

- Αυτό το πρόβλημα μπορεί να επιλυθεί απενεργοποιώντας την εμφάνιση μηνυμάτων σφάλματος στην οθόνη.

```
ini_set"display_errors", FALSE);
```

- Και επιτρέποντας την καταγραφή των σφαλμάτων.

```
ini_set("log_errors", TRUE);
```

σε ένα αρχείο

```
ini_set("error_log", "/var/log/php.log");
```

ή στην κεντρική εγκατάσταση παρακολούθησης σφαλμάτων του συστήματος

```
ini_set("error_log", "syslog");
```

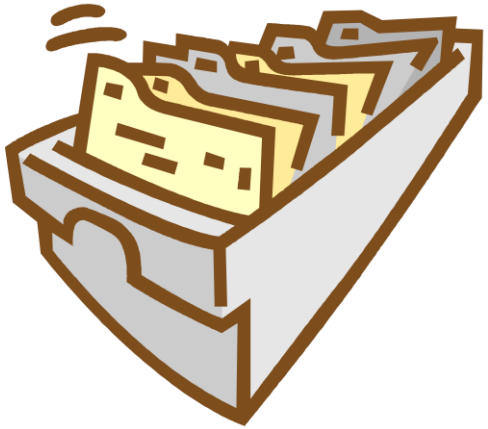
Ασφάλεια αρχείων

- Πολλές εφαρμογές PHP συχνά απαιτούν διάφορα αρχεία χρησιμότητας και διαμόρφωσης για λειτουργία.
- Επειδή αυτά τα αρχεία χρησιμοποιούνται μέσα στην εφαρμογή, καταλήγουν να είναι αναγνώσιμα από τον κόσμο.
- Αυτό σημαίνει ότι αν τα αρχεία βρίσκονται σε καταλόγους ιστού, οι χρήστες θα μπορούν να πραγματοποιούν λήψη & προβολή του περιεχομένου τους.

Ασφαλίζοντας τα αρχεία σας

- Μην τοποθετείτε τα αρχεία στη ρίζα ιστού που δεν χρειάζεται να βρίσκονται εκεί.
- Αν το αρχείο δεν εκπέμπει τίποτα, δώστε του μια επέκταση .php.
- Χρησιμοποιήστε το .htaccess για να αποκλείσετε την πρόσβαση σε αρχεία / καταλόγους.

```
<Files ~ "\.tpl$">
Order allow,deny
Deny from all
</Files>
```



Διασφάλιση αρχείων διαμόρφωσης

- Τα σενάρια διαμόρφωσης περιέχουν συνήθως ευαίσθητα δεδομένα που πρέπει να διατηρούνται ιδιωτικά.
- Απλώς αρνούμενοι την πρόσβαση στον ιστό, εξακολουθεί να είναι αναγνώσιμη σε όλους τους χρήστες του συστήματος.
- Τα ιδανικά αρχεία ρυθμίσεων θα είναι αναγνώσιμα μόνο από τον ιδιοκτήτη.

Λύση #1

Εάν το αρχείο ρυθμίσεων αποθηκεύει μόνο ρυθμίσεις σύνδεσης βάσεων δεδομένων, μπορείτε να τις ορίσετε μέσω οδηγιών ini, οι οποίες στη συνέχεια θα φορτωθούν από το httpd.conf μέσω οδηγίας Include.

mysql.cnf

```
mysql.default_host=localhost
mysql.default_user=forum
mysql.default_password=secret
```

httpd.conf

```
<VirtualHost 1.2.3.4>
Include "/site_12/mysql.cnf"
</VirtualHost>
```

- Το Apache αναλύει τα αρχεία ρυθμίσεων ως "root", οπότε το αρχείο ρυθμίσεων SQL μπορεί να έχει περιορισμένα δικαιώματα (0600) και να εξακολουθεί να λειτουργεί.

Λύση #2

Για όλες τις άλλες ρυθμίσεις, οι μεταβλητές περιβάλλοντος Apache μπορούν να χρησιμοποιηθούν για την "απόκρυψη" δεδομένων.

misc_config.cnf

```
SetEnv NNTP_LOGIN  "login"
SetEnv NNTP_PASS   "passwd"
SetEnv NNTP_SERVER "1.2.3.4"
```

httpd.conf

```
<VirtualHost 1.2.3.4>
Include "misc_config.cnf"
</VirtualHost>
```

```
echo $_SERVER['NNTP_LOGIN']; // login
echo $_SERVER['NNTP_PASS']; // passwd
echo $_SERVER['NNTP_SERVER']; // 1.2.3.4
```

Ασφάλεια περιόδου λειτουργίας (Session)

- Οι περίοδοι σύνδεσης είναι ένα κοινό εργαλείο για την παρακολούθηση χρηστών σε έναν ιστότοπο.
- Κατά τη διάρκεια μιας επίσκεψης, η συνεδρία είναι στην πραγματικότητα η ταυτότητα του χρήστη.
- Εάν μια ενεργή σύνοδος μπορεί να ληφθεί από τρίτο μέρος, μπορεί να αναλάβει τον προσδιορισμό του χρήστη, ο οποίος έχει συμβιβαστεί.

Εξασφάλιση αναγνωριστικού περιόδου σύνδεσης (Securing Session ID)

Για να αποφευχθεί η κλοπή ταυτότητας περιόδου σύνδεσης, το αναγνωριστικό μπορεί να τροποποιηθεί σε κάθε αίτημα, ακυρώνοντας παλιές τιμές.

```
<?php
session_start();
if (!empty($_SESSION)) { // not a new session
 session_regenerate_id(TRUE); // make new session id
}
?>
```

- Επειδή η περίοδος σύνδεσης αλλάζει σε κάθε αίτημα, το κουμπί "πίσω" σε ένα πρόγραμμα περιήγησης δεν θα λειτουργεί πλέον, καθώς θα κάνει ένα αίτημα με το παλιό αναγνωριστικό περιόδου σύνδεσης.

Επικύρωση περιόδου σύνδεσης (Session Validation)

- Μια άλλη τεχνική ασφαλείας περιόδου σύνδεσης είναι να συγκρίνετε τις κεφαλίδες υπογραφής του προγράμματος περιήγησης.

```
session_start();
$chk = @md5(
        $_SERVER['HTTP_ACCEPT_CHARSET'] ,
        $_SERVER['HTTP_ACCEPT_ENCODING'] ,
        $_SERVER['HTTP_ACCEPT_LANGUAGE'] ,
        $_SERVER['HTTP_USER_AGENT']);

if (empty($_SESSION))
        $_SESSION['key'] = $chk;
else if ($_SESSION['key'] != $chk)
        session_destroy();
```

Ασφαλέστερη αποθήκευση συνεδριών

- Από προεπιλογή, οι συνεδρίες PHP αποθηκεύονται ως αρχεία μέσα στον κοινό /tmp κατάλογο.

- Αυτό συχνά σημαίνει ότι οποιοσδήποτε χρήστης στο σύστημα θα μπορούσε να δει ενεργές περιόδους σύνδεσης και να τις "αποκτήσει" ή ακόμα και να τροποποιήσει το περιεχόμενό τους.

- Λύσεις;

- Ξεχωριστός κατάλογος αποθήκευσης συνεδριών μέσω session.save_path.
- Μηχανισμός αποθήκευσης βάσεων δεδομένων, mysql, pgsql, oci, sqlite.
- Κοινόχρηστη μνήμη αποθήκευσης περιόδου λειτουργίας "mm".
- Προσαρμογέας χειρισμού συνόλων που επιτρέπει την αποθήκευση δεδομένων οπουδήποτε.

Κοινή φιλοξενία (Shared Hosting)

- Οι περισσότερες εφαρμογές PHP εκτελούνται σε κοινόχρηστα περιβάλλοντα όπου όλοι οι χρήστες "μοιράζονται" τις ίδιες παρουσίες διακομιστή ιστού.

- Αυτό σημαίνει ότι όλα τα αρχεία που εμπλέκονται στην προβολή περιεχομένου πρέπει να είναι προσβάσιμα στον εξυπηρετητή ιστού (αναγνώσιμος από τον κόσμο).

- Συνεπώς, αυτό σημαίνει ότι οποιοσδήποτε χρήστης θα μπορούσε να διαβάσει το περιεχόμενο των αρχείων όλων των άλλων χρηστών.

Η PHP λύση

- Η λύση της PHP σε αυτό το πρόβλημα είναι 2 οδηγίες INI.

open_basedir - περιορίζει την πρόσβαση στο αρχείο σε έναν ή περισσότερους καθορισμένους καταλόγους.

- Σχετικά αποτελεσματική.
- Απλό.

safe_mode - περιορίζει την πρόσβαση σε αρχεία με βάση το uid / gid του σεναρίου εκτέλεσης και του αρχείου που πρέπει να έχετε πρόσβαση.

- Αργή και πολύπλοκη προσέγγιση.
- Μπορεί να παρακαμφθεί με λίγη προσπάθεια.

Ασφάλεια μέσω της ανασφάλειας

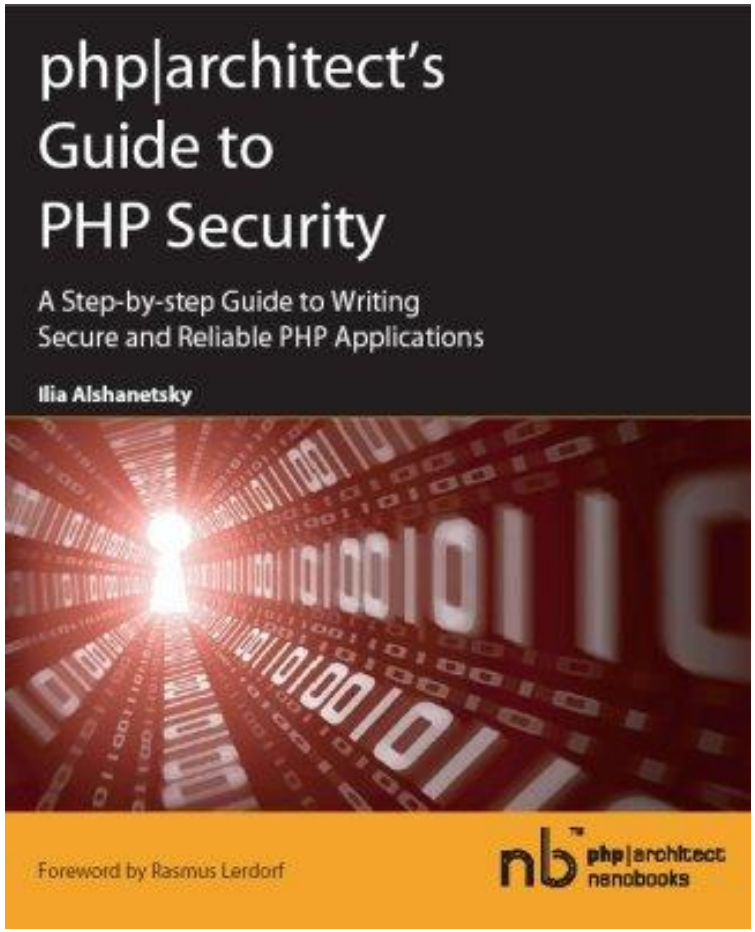
- Ενώ από μόνο του δεν είναι μια καλή προσέγγιση για την ασφάλεια, ως προσθήκη στα υπάρχοντα μέτρα, η σκοτεινότητα μπορεί να είναι ένα ισχυρό εργαλείο.

- Απενεργοποιήστε την κεφαλίδα αναγνώρισης PHP.
  expose_php = off

- Απενεργοποιήστε την κεφαλίδα αναγνώρισης του Apache.
  ServerSignature = off

- Αποφύγετε τα προφανή ονόματα για τους περιορισμένους πίνακες ελέγχου.

<?php include “/book/plug.inc”; ?>



Ερωτήσεις;