SIN6U5 : Développement web 2 (Prog. web coté serveur) Sites dynamiques

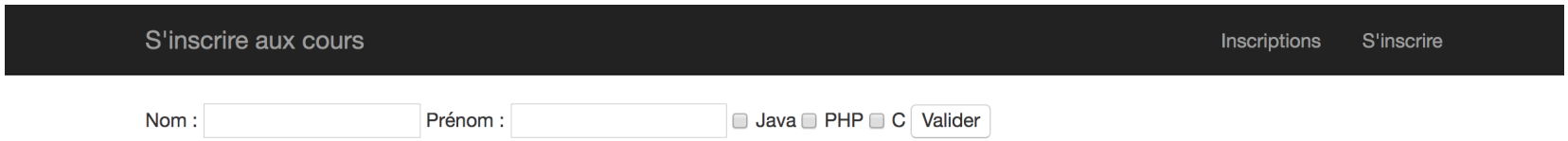
Luigi Santocanale
LIF, Aix-Marseille Université

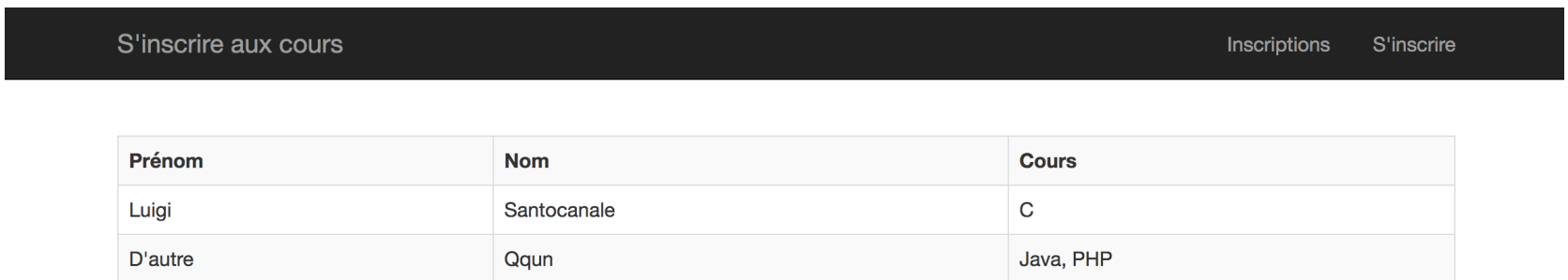
Transparents basés sur le cours de Bertrand Estellon

1er février 2018

Plan

Objectifs


S'inscrire aux cours
Inscriptions
S'inscrire
Nom :
Prénom :
Java
PHP
C
Valider


S'inscrire aux cours
Inscriptions
S'inscrire
Prénom
Nom
Cours
Luigi
Santocanale
C
D'autre
Qqun
Java, PHP

Les fonctionnalités

- Deux pages : inscription, liste des inscrits
- Base de données
- Vérification des formulaires
- Présentation homogène
- Facilité de mise à jour

Plan

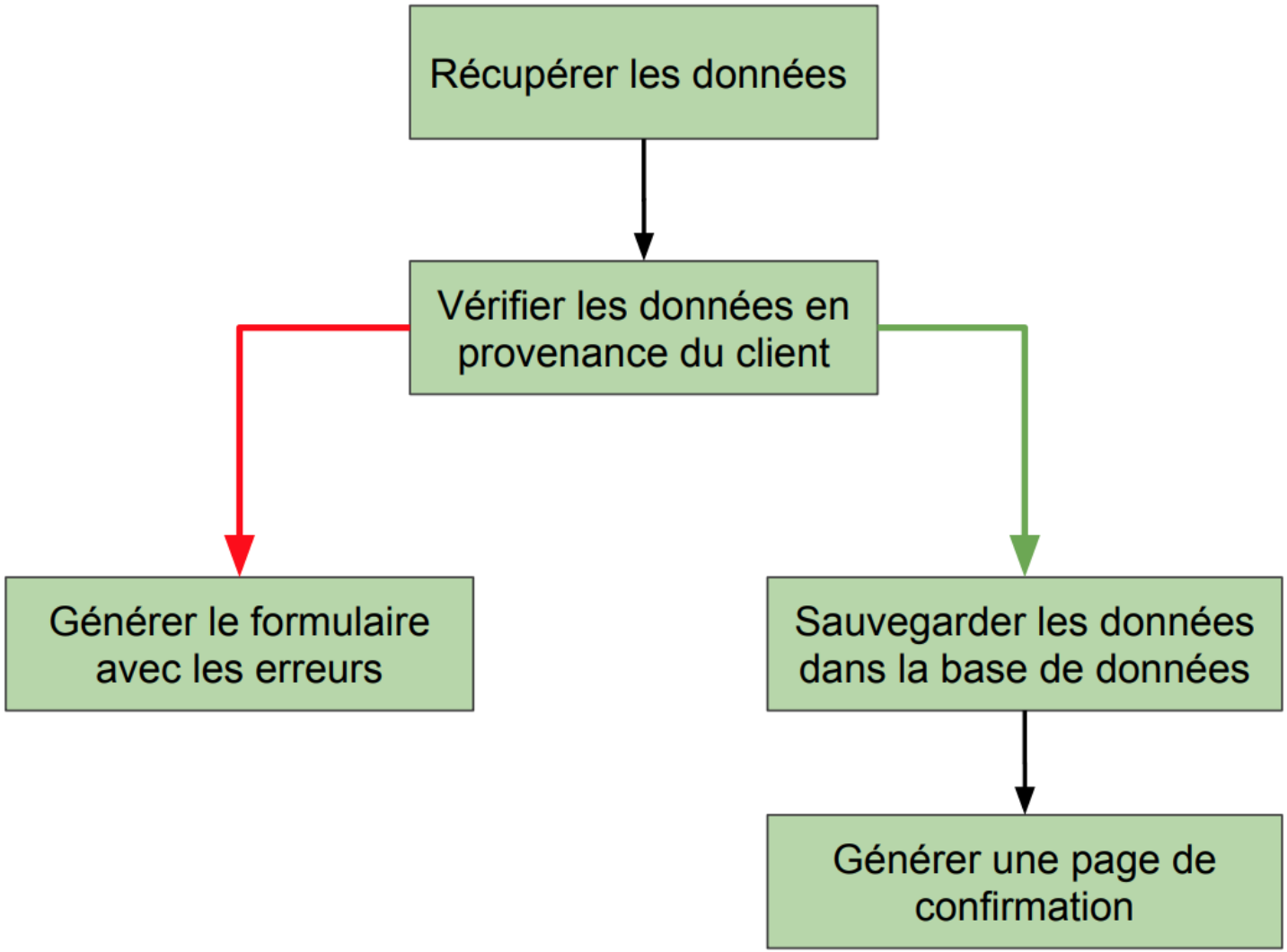
Les formulaires en HTML

Définition d'un formulaire en HTML :

```
<form method="post" action="index.php">
     <input type="text" name="firstname"
            value="Bob" placeholder="Votre nom">
    <input type="checkbox" name="choices[]" value="0"> Java
    <input type="checkbox" name="choices[]" value="1"> C
    <button type="submit">Valider</button>
</form>
```

Attention : la mise en forme est omise.

Les étapes du traitement du formulaire



La méthode POST de HTTP

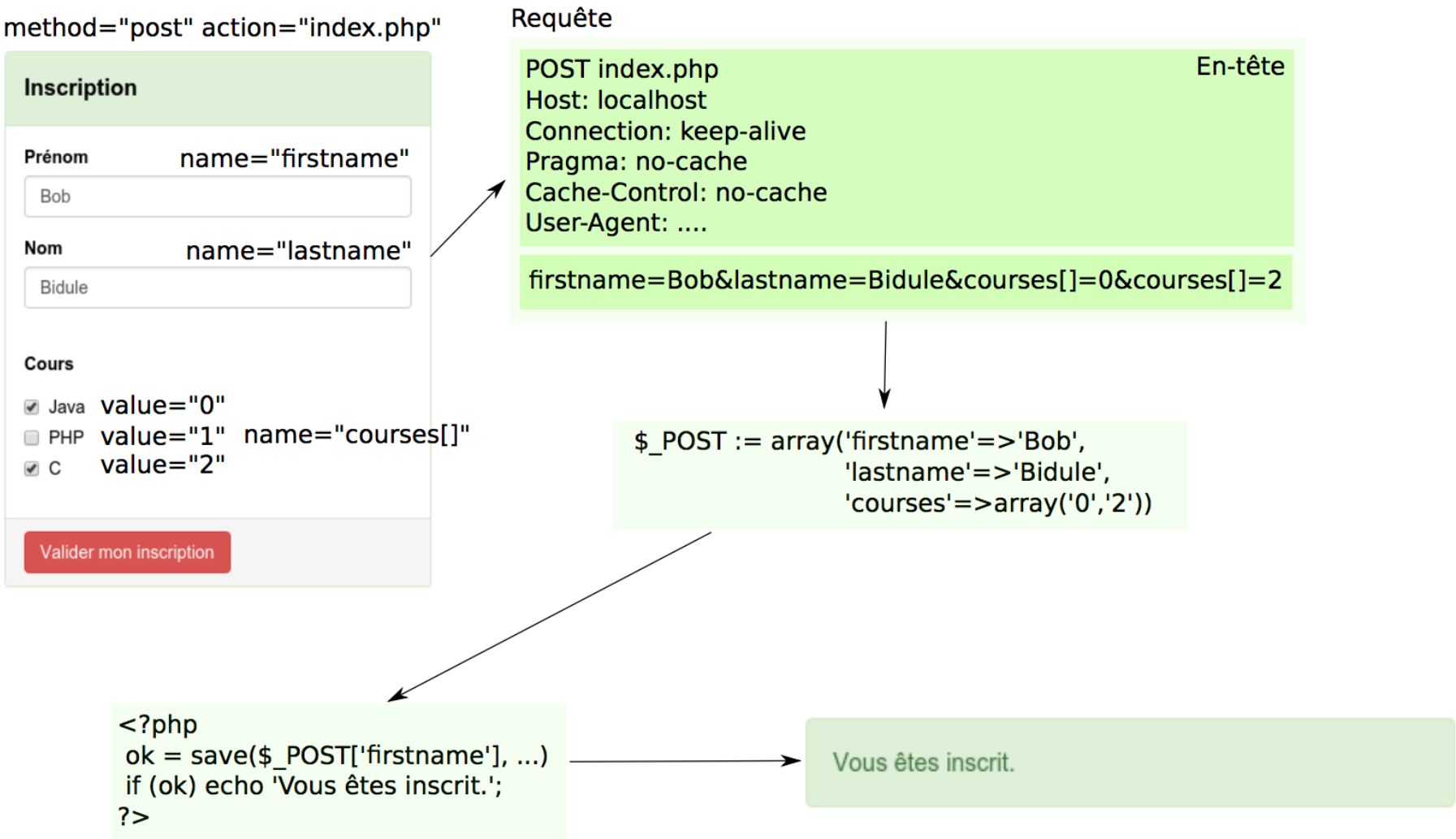
Considérons le formulaire suivant :

```
<form method="post" action="index.php">
    <input type="text" name="firstname">
    <button type="submit">Valider</button>
</form>
```

Lorsque l'utilisateur clique sur le bouton "submit", le navigateur envoie :

- une requête HTTP au serveur ;
- en demandant la page `index.php` (action) ;
- via la méthode POST du protocole ;
- en plaçant les données saisies à la suite de l'en-tête de la requête.

Méthode POST et variable super-globale `$_POST`



Plan

Filtrage des données

```php
<?php
$firstname = filter_input(
    INPUT_POST,
    'firstname',
    FILTER_VALIDATE_REGEXP,
    'options' => ['regexp'=>'/^[A-Za-z\' -]{1,20}$/']);
?>
```

Après l'exécution de ces lignes, la variable `$firstname` vaut :

- NULL si `$_POST['firstname']` n'est pas défini
- false si la valeur ne passe pas le filtre
- la valeur de `$_POST['firstname']` sinon

Filtrage des données

Vérification et formatage des données envoyées par le client :

```php
<?php
$filters = [/* définition des filtres */];
$form_data = filter_input_array(INPUT_POST,$filters);
?>
```

Un exemple de tableau de filtres :

```php
<?php
$filters = [ 'firstname' =>
    [
        'filter' => FILTER_VALIDATE_REGEXP,
        'flags' => FILTER_NULL_ON_FAILURE,
        'options' => ['regexp'=>'/^[A-Za-z\' -]{1,20}$/'],
    ],
            // autres filtres associés à d'autres noms
 ↪  d'inputs
];
?>
```

Filtrage des données

La méthode `filter_input_array` retourne :

- NULL si aucune donnée n'a été fournie par le client
- Un tableau qui associe à chaque nom de champs une valeur

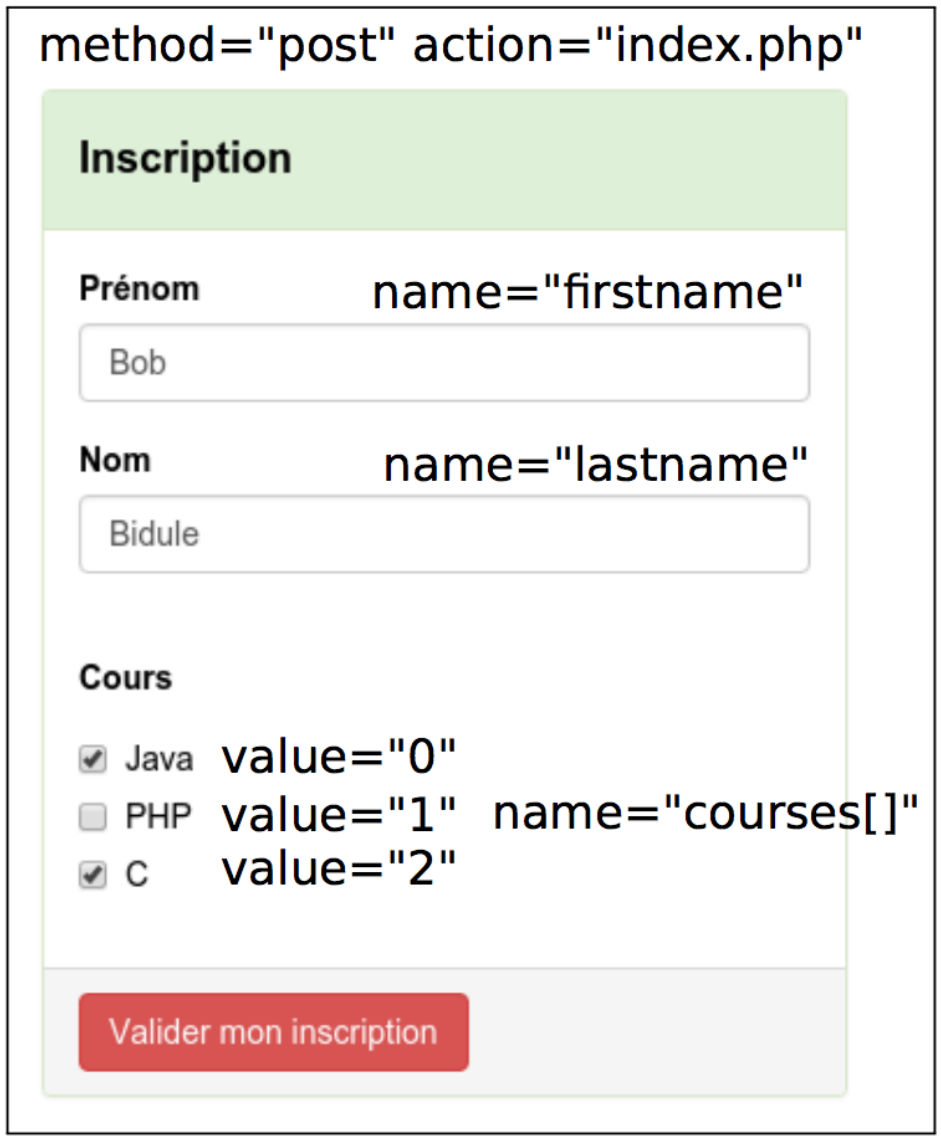
La valeur associée au nom d'un champ est égale à :

- FALSE (null avec FILTER_NULL_ON_FAILURE) si le filtre a échoué ;
- la valeur fournie par le client sinon.

Exemples :

`$_POST`	valeur retournée
[]	NULL
`['firstname'=>'2']`	`['firstname'=>NULL]`
`['firstname'=>'Bob']`	`['firstname'=>'Bob']`

Filtrage des données



Filtrage des données

```php
<?php
$filters = [
    'firstname' => [
        'filter' => FILTER_VALIDATE_REGEXP,
        'flags' => FILTER_NULL_ON_FAILURE,
        'options' => ['regexp'=>'/^[A-Za-z][A-Za-z\' -]*$/']
    ],
    'lastname' => [
        'filter' => FILTER_VALIDATE_REGEXP,
        'flags' => FILTER_NULL_ON_FAILURE,
        'options' => ['regexp'=>'/^[A-Za-z][A-Za-z\' -]*$/']
    ],
    'courses' => [
        'filter' => FILTER_VALIDATE_INT,
        'flags'  => FILTER_REQUIRE_ARRAY |
                    FILTER_NULL_ON_FAILURE,
    ] ];
?>
```

Cross-site scripting (XSS)

Les XSS sont des failles permettant l'injection de contenus dans une page déclenchant des actions du navigateur Web (JavaScript, balises, etc.).

Les risques :

- Redirection (hameçonnasse)
- Vol d'informations (sessions et cookies)
- Actions sur le site sous l'identité de la victime

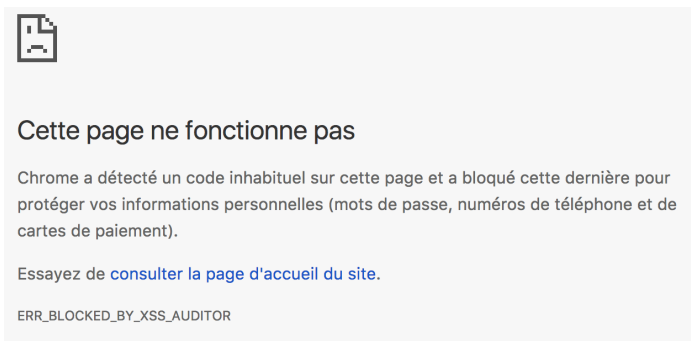
XSS, un premier exemple

```php
<?php
// test.php
echo "Votre nom est : " . $_GET['nom'];
?>
```

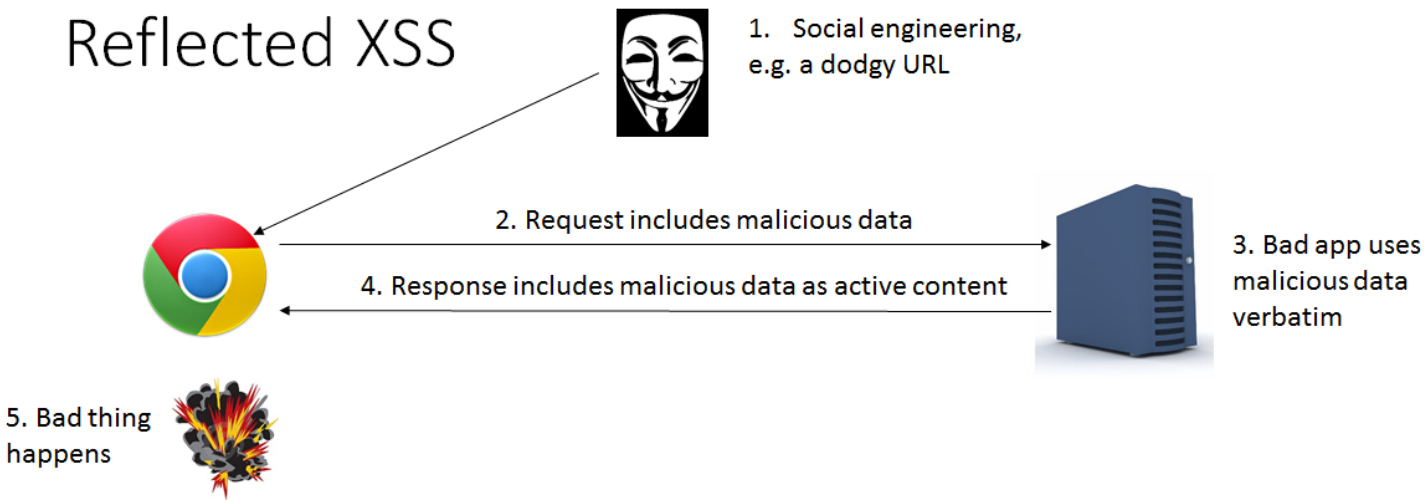
```
http://localhost:8080/test.php?nom=<script>alert('Voici un XSS');</script>
```

Firefox :

Chrome :



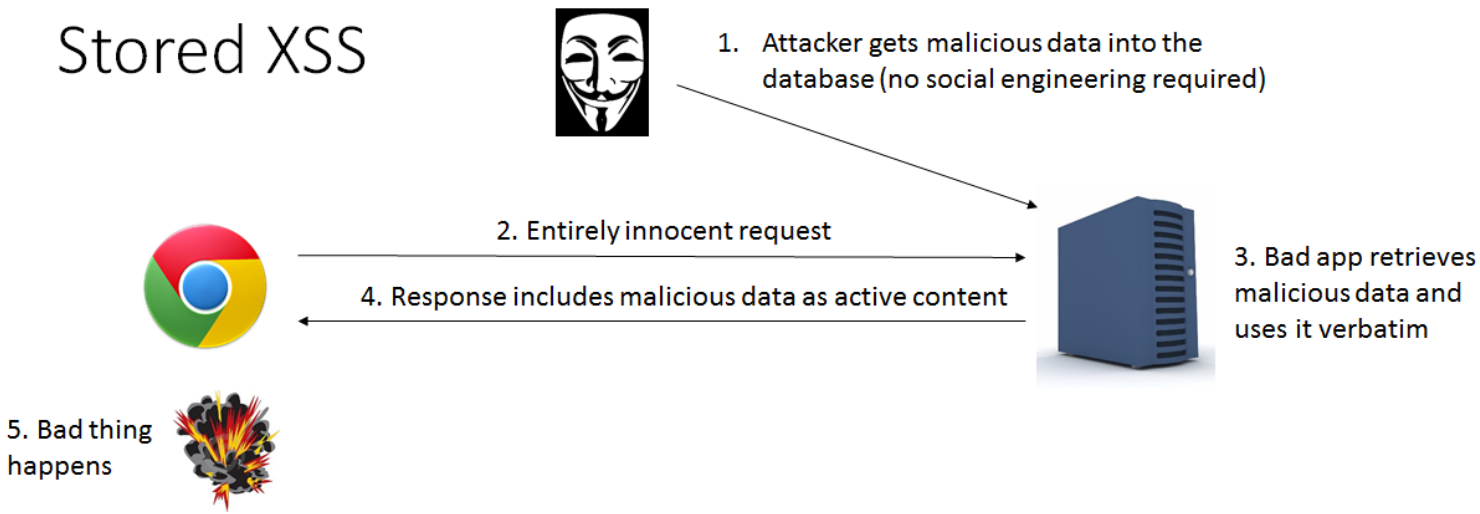
Attaque XSS réfléchi



Source :

`http://blog.scottlogic.com/2016/02/29/Cross-site-scripting.html`

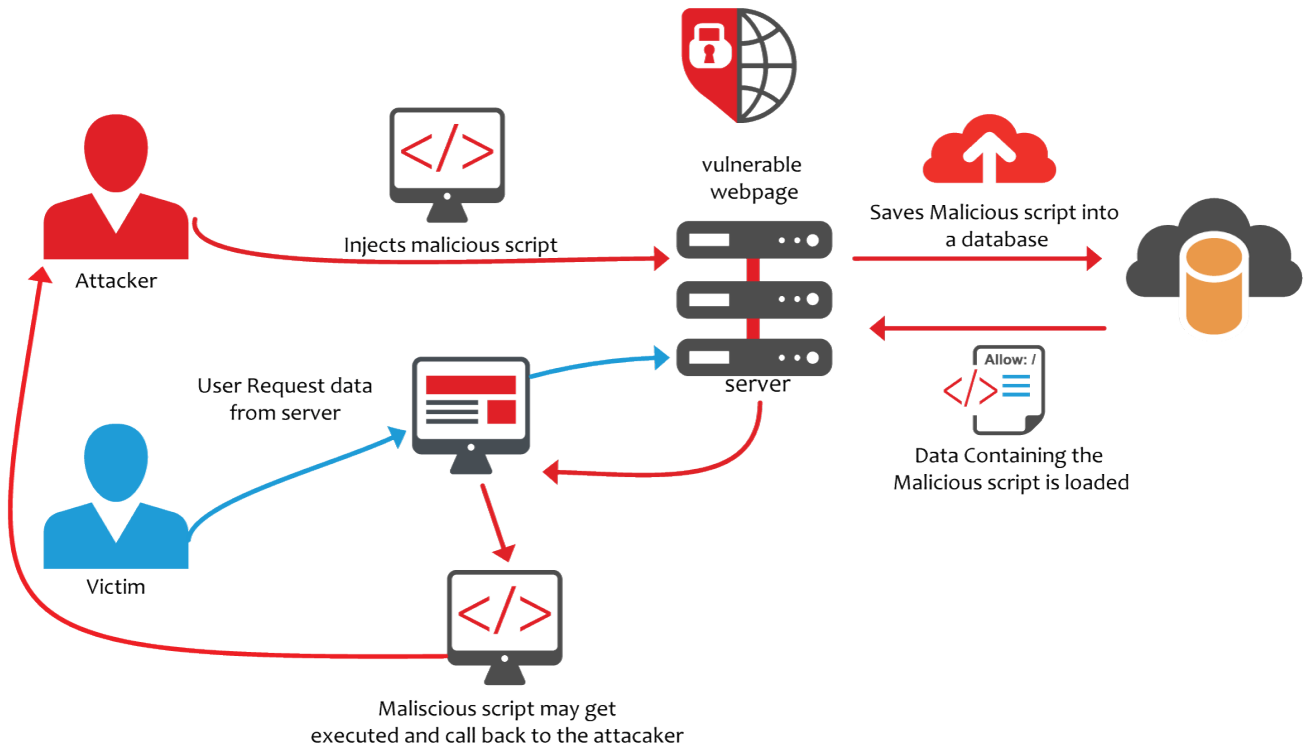
Attaque XSS stocké



Source :

`http://blog.scottlogic.com/2016/02/29/Cross-site-scripting.html`

Attaque XSS (stocké) aveugle



Source : `https://www.acunetix.com/blog/articles/blind-xss/`

Prévenir les XSS

- marquer les chaînes de caractères rentrées par l'utilisateur ;
- filtrer ces chaînes avant le stocker ;
- filtrer ces chaînes avant les des afficher ;
- utiliser un système de filtrage automatique, typiquement, auto-Escape dans un langage de templates.

Exemple de filtre prévenant un XSS :

```
<?php
$comment = filter_var($user_comment,
                      FILTER_SANITIZE_SPECIAL_CHARS);
?>
```

Entités HTML

Le code suivant

```php
<?php
$name = filter_input(INPUT_POST, 'name');
if ($name===null) $name='';
?>
Vous avez saisi <?=$name ?>
<input type="text" name="name" value="<?=$name ?>" >
```

génère le code HTML suivant si `$_POST['name']` contient `<"`

```
Vous avez saisi <"
<input type="text" name="name" value="<"" >
```

Problème : On obtient un code HTML qui n'est pas valide.

Entités HTML

Solution :

```php
<?php
$name = filter_input(INPUT_POST, 'name',
                     FILTER_SANITIZE_SPECIAL_CHARS,
                     ['options' => ['default' => '']]);
?>
Vous avez saisi <?=$name ?>
<input type="text" name="name" value="<?=$name ?>" >
```

On obtient le code suivant si `$_POST['name']` contient `<"`

```
Vous avez saisi &#60;&#34;
<input type="text" name="name" value="&#60;&#34;" >
```

Résume

Fonctions de filtrage de données produites par l'utilisateur :

`filter_input`, `filter_var`, `filter_input_array`, `filter_var_array` ...

Deux types de filtres :

- Filtres de validation :

  `FILTER_VALIDATE_INT`, `FILTER_VALIDATE_EMAIL`, `FILTER_VALIDATE_REGEXPR`, ...

  pour vérifier la conformité des données aux attentes.
- Filtre de nettoyage :

  `FILTER_SANITIZE_SPECIAL_CHARS`,

  ...pour prévenir le XSS, et d'autres taches similaires.

Voir : `http://php.net/manual/fr/book.filter.php`

Plan

Création de la base de données

```
PRAGMA foreign_keys = ON;
DROP TABLE IF EXISTS choices;
DROP TABLE IF EXISTS courses;
DROP TABLE IF EXISTS students;

CREATE TABLE courses(
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        name TEXT
);

CREATE TABLE students (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        firstname TEXT,
        lastname TEXT
);
```

Création de la base de données

```
CREATE TABLE choices (
       student_id INTEGER,
       course_id INTEGER,
       UNIQUE(student_id, course_id),
       FOREIGN KEY(student_id)
                      REFERENCES students(id) ON DELETE CASCADE,
       FOREIGN KEY(course_id)
                      REFERENCES courses(id) ON DELETE CASCADE
);

INSERT INTO courses(name) VALUES ('Java');
INSERT INTO courses(name) VALUES ('PHP');
INSERT INTO courses(name) VALUES ('C');
```

Les requêtes de sélection

Obtenir tous les cours disponibles :

```
SELECT * FROM courses
```

Obtenir les cours choisis par les étudiants :

```
SELECT students.firstname,
               students.lastname,
               group_concat(courses.name, ', ') as courses
FROM students
LEFT JOIN choices ON students.id = choices.student_id
LEFT JOIN courses ON choices.course_id = courses.id
GROUP BY students.id
```

Les requêtes d'insertion

Insérer un nouvel étudiant :

```
INSERT INTO students(firstname, lastname) VALUES (?, ?)
```

Associer un cours à un étudiant :

```
INSERT INTO choices(student_id, course_id) VALUES (?, ?)
```

Base de données en PHP avec PDO

Nous allons utiliser PDO qui :

- est interface pour accéder à une base de données en PHP ;
- gère la connexion, l'envoie de requêtes, etc.
- permet de changer facilement de SGBD ;
- utilise des objets.

En PHP, l'instanciation d'un objet est similaire à Java :

```php
<?php
$dbHost = $_SERVER['dbHost'];
$dbBd = $_SERVER['dbBd'];
$dbPass = $_SERVER['dbPass'];
$dbLogin = $_SERVER['dbLogin'];
$url = "mysql:host=$dbHost;dbname=$dbBd";
$database = new PDO($url, $dbLogin, $dbPass);
?>
```

Exécution d'une requête et consultation des résultats

En PHP, on utilise l'opérateur -> pour accéder aux membres de l'objet :

```php
<?php
$statement = $database->query("SELECT * FROM courses");
$rows = $statement->fetchall();
?>

<?php foreach ($rows as $row): ?>
    <input type="checkbox"
           name="courses[]"
           value="<?=$row['id']?>">
    <?=$row['name']?>
    <br>
<?php endforeach; ?>
```

Les requêtes de modification

Insertion d'un nouvel étudiant dans la base :

```php
<?php
$statement = $database->prepare(
    'INSERT INTO students(firstname, lastname)
   VALUES (:firstname, :lastname)');
$statement->execute([
    'firstname'=>$firstname
    'lastname' =>$lastname
]);
?>
```

Consultation du dernier identifiant généré :

```php
<?php
$studentId = $database->lastInsertId();
?>
```

Préparer des requêtes de sélection

Consultation des cours associés à un étudiant :

```php
<?php
$query = 'SELECT course_id FROM choices WHERE student_id = ?'
$statement = $database->prepare($query)
$statement->execute([$student_id]);
$rows = $statement->fetchall();
foreach($rows as $row) echo $row['course_id'];
?>
```

Il est également possible de “demander” une ligne à la fois :

```php
<?php
for (;;) {
    $row = $statement->fetch();
    if ($row===false) break;
    array_push($rows, $row);
}
?>
```

Sécurité et injection SQL

- Une injection SQL est l'exploitation d'une faille de sécurité qui consiste à injecter une requête SQL non prévue :

```php
<?php
$query = "SELECT id FROM user WHERE "
        ."username='$username' AND "
        ."password='$password'";
$statement = $database->query($query);
$authenticated = $statement->rowCount()===1;
?>
```

- Si `$username` est égal à `"admin'; -"`, on obtient la requête :

```
SELECT id FROM user
WHERE username='admin'; --' AND password='...'
```

- Et le client se retrouve authentifié en tant qu'administrateur...
- Donc utiliser `prepare` (même si cela n'est pas suffisant)

Les transactions

"Agréger" plusieurs requêtes avec une transaction :

```php
<?php
$database = new PDO($url, $dbLogin, $dbPass);
$database->beginTransaction();
$database->exec(
    "UPDATE account SET balance=balance-1 "
   ."WHERE name='a'");
$database->exec(
    "UPDATE account SET balance=balance+1 "
   ."WHERE name='b'");
if ($success)
    $database->commit();
else
    $database->rollback();
?>
```

- `beginTransaction()` : débuter la transaction
- `commit()` : valider la transaction
- `rollback()` : annuler la transaction

Plan

Structure du projet

Source :
http://pageperso.lif.univ-mrs.fr/~luigi.santocanale/teaching/DW/code/integrationDB.zip

```
.
|-- Makefile
|-- code
|   `-- tools.php
|-- config
|   `-- pages.inc.php
|-- css
|   `-- bootstrap.css
|-- db
|   |-- create.sql
|   `-- database.sqlite
|-- index.php
|-- pages
|   |-- registration_create.php
|   |-- registration_new.php
|   `-- registrations.php
|-- templates
|   |-- error.php
|   |-- footer.html.php
|   |-- header.html.php
|   |-- menu.html.php
|   |-- registration_form.html.php
|   |-- registration_list.html.php
|   `-- success.php
`-- tree.txt

6 directories, 18 files
```

Le formulaire d'inscription

```php
<?php
// pages/registration_new.php
include __DIR__.'/../code/tools.php';

$courses = courses();
include __DIR__.'/../templates/registration_form.html.php';
```

Méthode pour récupérer les cours (tools.php) :

```php
$db = new PDO('sqlite:db/database.sqlite');

function courses() {
    global $db;

    $statement = $db->query("SELECT * FROM courses");
    return $statement->fetchall();
}
```

Le formulaire d'inscription

```
<!-- templates/registration_form.html.php -->
<div class="container">
    <form method="post"
          action="?page=registrationCreate">
        <?php if(isset($error)): ?>
            <div class="error"><?=$error ?></div>
        <?php endif; ?>
        Nom :
        <input type="text" name="firstname" placeholder="Votre nom"
               value="<?=set_value('firstname') ?>">
        Prénom :
        <input type="text" name="lastname" placeholder="Votre prénom"
               value="<?=set_value('lastname') ?>">
        <?php foreach ($courses as $course): ?>
            <input type="checkbox" name="courses[]"
                   value="<?=$course['id'] ?>">
            <?=$course['name'] ?>
        <?php endforeach; ?>
        <button type="submit">Valider</button>
    </form>
</div><!-- container -->
```

Traitement de l'inscription

Traitement de l'inscription : `registration_create.php`

```php
$filters = [
    'firstname' =>
    [
        'filter' => FILTER_VALIDATE_REGEXP,
        'flags' => FILTER_NULL_ON_FAILURE,
        'options' => ['regexp'=>'/^[A-Za-z\' -]{1,20}$/'],
        'message' => 'Le prénom doit etre alphaNum'
    ],
    'lastname' =>

    'courses' => [
        'filter'    => FILTER_VALIDATE_INT,
        'flags'     => FILTER_REQUIRE_ARRAY,
        'options'   => ['min_range' => 1, 'max_range' => 3],
        'message' => 'Vous devez choisir au moins un cours'
    ]
];

$form_data = filter_input_array(INPUT_POST, $filters, true);
```

Traitement de l'inscription

`registration_create.php` : traitement de l'inscription

```
$error = compute_error($form_data, $filters);

if($error!==false) {
    $courses = courses();
    include "$pathToTemplates/registration_form.html.php";
} else {
    $success = register(
        $form_data['firstname'],
        $form_data['lastname'],
        $form_data['courses']);
    include
 ↪   $success?"$pathToTemplates/success.php":"$pathToTemplates/error.ph
}
```

Traitement de l'inscription

Méthode pour réinjecter des valeurs dans le formulaire (`tools.php`) :

```
function set_value($name){
    return filter_input(
        INPUT_POST,
        $name,
        FILTER_SANITIZE_SPECIAL_CHARS,
        ['options' => ['default' => '']]);
}
```

Génération du tableau d'erreurs (`tools.php`) :

```
function compute_error($form_data,$filters){
    foreach($filters as $key => $filter){
        $data = &$form_data[$key];
        if (
            $data===null ||
            (is_array($data)&&in_array(null,$data,true))
        ){
            return $filter['message'];
        }
    }
    return false;
}
```

Traitement de l'inscription

Méthode pour ajouter une inscription (tools.php) :

```php
function register($firstname,$lastname,$courses) {
    global $db;

    $db->beginTransaction();
    $query1='INSERT INTO students(firstname, lastname) VALUES (?, ?)';
    $statement = $db->prepare($query1);
    $success = $statement->execute([$firstname, $lastname]);
    if (!$success) {
        $db->rollback();
        return false;
    }
    $student_id = $db->lastInsertId();

    $query='INSERT INTO choices(student_id, course_id) VALUES (?, ?)';
    $statement = $db->prepare($query);
    foreach ($courses as $course_id) {
        $success = $statement->execute([$student_id, $course_id]);
        if (!$success) {
            $db->rollback();
            return false;
        }
    }
    $db->commit();
    return true;
}
```

Liste des inscrits

```php
<?php
// pages/registrations.php

include __DIR__.'/../code/tools.php';
$registrations = registrations();
include __DIR__.'/../templates/registration_list.html.php';
```

La méthode pour récupérer les inscriptions (`tools.php`) :

```php
function registrations() {
    global $db;
    $query = "SELECT students.firstname,students.lastname,"
           ."group_concat(courses.name, ', ') "
           ."as courses "
           ."FROM students "
           ."LEFT JOIN choices ON students.id = choices.student_id "
           ."LEFT JOIN courses ON choices.course_id = courses.id "
           ."GROUP BY students.id";
    $statement = $db->query($query);

    return $statement->fetchall();
}
```

Liste des inscrits

```
<!-- templates/registration_list.html.php -->
<br>
<div class="container">
    <table class="table table-striped table-bordered">
        <tr>
            <th>Prénom</th>
            <th>Nom</th>
            <th>Cours</th>
        </tr>
        <?php foreach($registrations as $registration):  ?>
            <tr>
                <td><?=$registration['firstname'] ?></td>
                <td><?=$registration['lastname'] ?></td>
                <td><?=$registration['courses'] ?></td>
            </tr>
        <?php endforeach; ?>
    </table>
</div>
```