



Université  
De Boumerdes



Université  
De Limoges

# Les différents types de vulnérabilités

*Réalisé par :* Dr RIAHLA

Docteur de l'université de Limoges (France) Maître de conférences HDR à l'université de Boumerdes

*Réalisé par :* Dr RIAHLA



Université  
De Boumerdes

---

# Vulnérabilités applicatives

# Vulnérabilités applicatives

---

Beaucoup d'applications sont vulnérables dues à de la mauvaise programmation (par manque de temps, de motivation, ...) ou volontairement (aménagement d'un point d'entrée, ...).

- Services réseaux (daemons).
- Les applications téléchargées (applet java, ...).
- Les applications web (scripts cgi, ...).

# Vulnérabilités applicatives

---

Les vulnérabilités peuvent être dues:

**Backdoors:** laissées volontairement ou involontairement sur un service par le programmeur

## **Erreurs de programmation**

- Débordements de tampons (buffer overflow)
- Entrées utilisateurs mal validées
- Les problèmes de concurrence
- Chaînes de format



Université  
De Boumerdes

---

# Buffer Overflow

# Buffer Overflow (Introduction)

---

```
int main (int argc, char **argv)
{
char buf [8] ;
strcpy (buf,argv [1]) ;
}
```



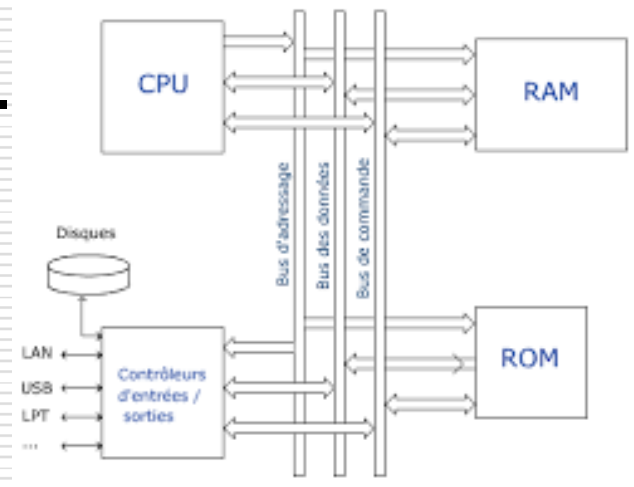
## Exécution:

```
[root@austramine]$ ./demo aaaaaaaaaaaaaaaaaaaaaa
Segmentation fault
```

# Buffer Overflow (Introduction)

---

➤ La fonction principale d'un processeur est de traiter et de déplacer des données.



➤ Lors de ces traitements, le processeur a besoin d'un emplacement afin de sauvegarder rapidement les données traitées.

# Buffer Overflow (Introcution)

---

- La taille des registres ne permet pas à ceux-ci de jouer ce rôle.
- Ces informations sont donc sauvées dans une zone mémoire appelées **pile**.
- Elle est stockée en mémoire à une adresse spécifique

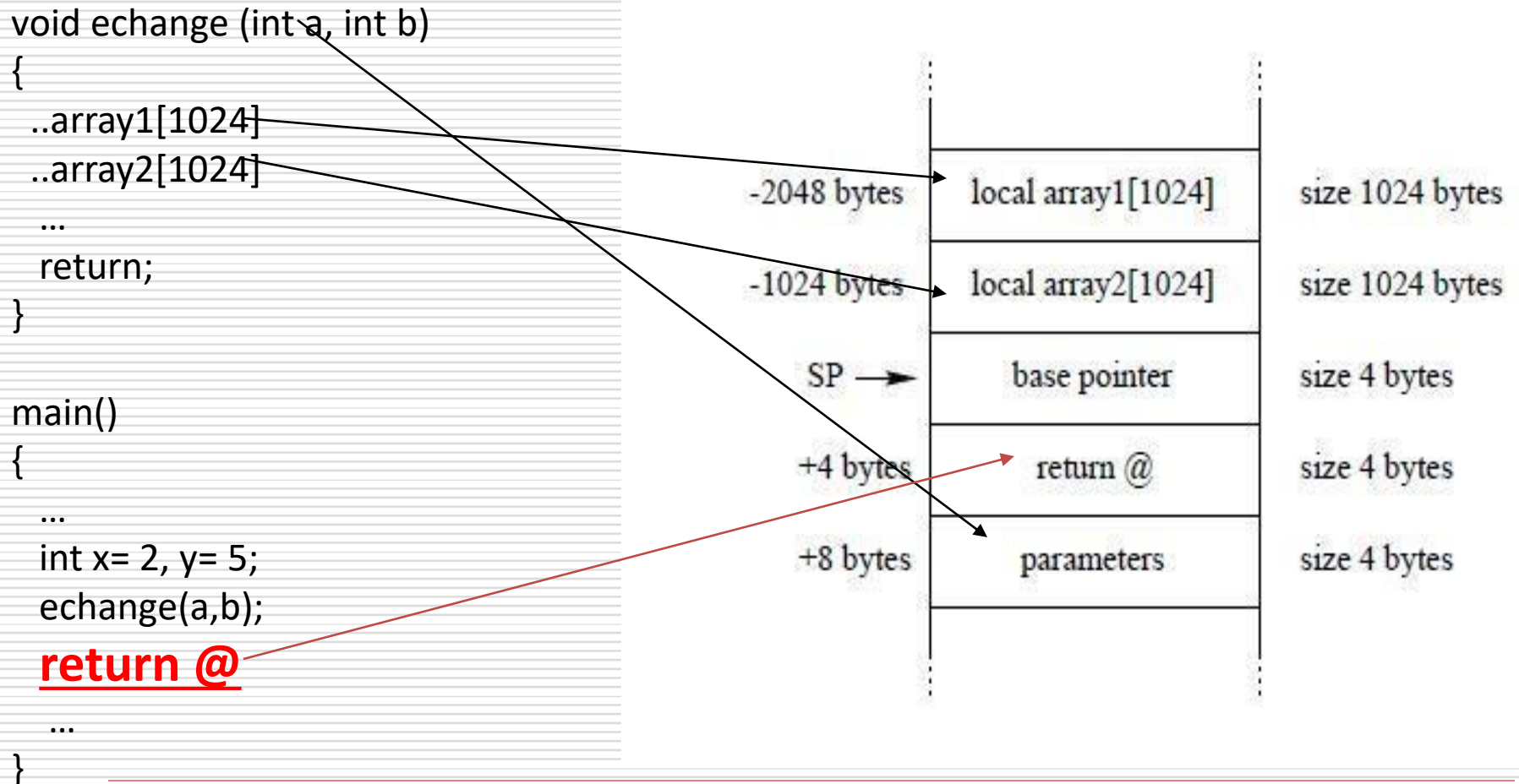


# Buffer Overflow (la pile)

---

Lors de l'exécution d'un programme qui fait **appel a une procédure**, le processeur sauve l'adresse de retour dans la pile, lorsque la procédure se terminera le processeur retournera à l'adresse spécifiée et continuera son travail...

# Buffer Overflow (Structure de la pile)



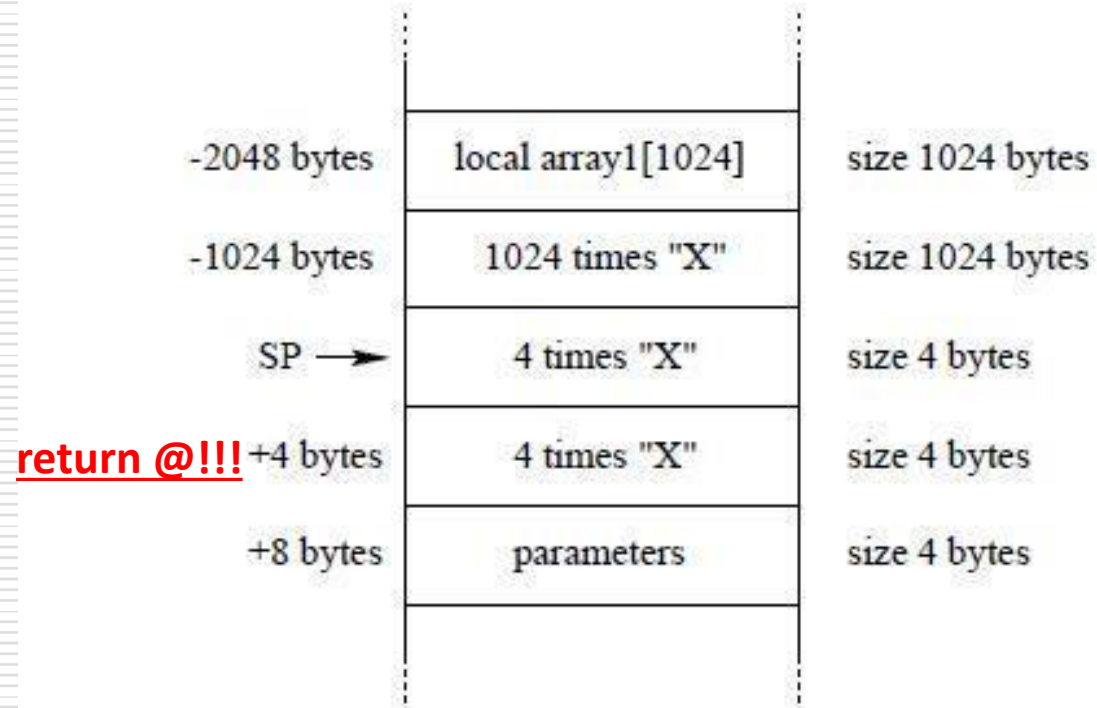
# Buffer Overflow

## (Abuser l'adresse de retour)

---

Si (par hasard !) une procédure écrivait plus d'octets (bytes) dans une variable locale afin que la taille nécessaire à son stockage dans la pile dépasse celle de l'adresse de retour, on appellerait ceci un **Buffer Overflow**.

# Buffer Overflow (Abuser l'adresse de retour)



1032 fois le caractère "X" dans le tableau local "array2"

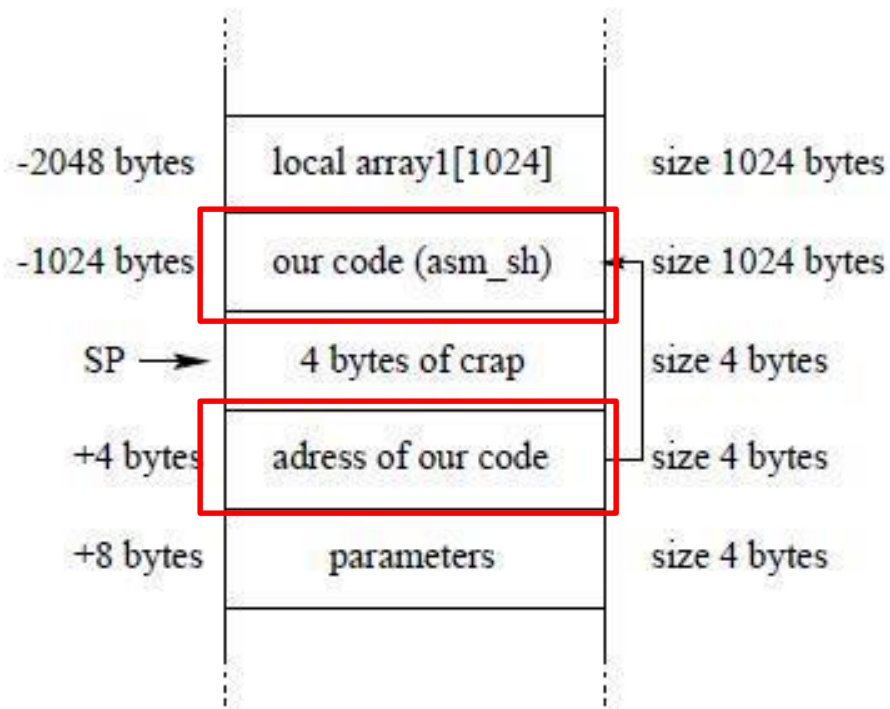
# Buffer Overflow (Exploitation)

---

- Il est possible donc d'indiquer au programme une nouvelle adresse de retour contenant **un code totalement différent (code pirate )**.
- Le programme sautera alors automatiquement à l'adresse spécifiée. . .
- Forcer le programme à sauter vers une adresse où est situé un code assembleur destiné par exemple à exécuter un shell UNIX (**/bin/sh**).

# Buffer Overflow (Exploitation)

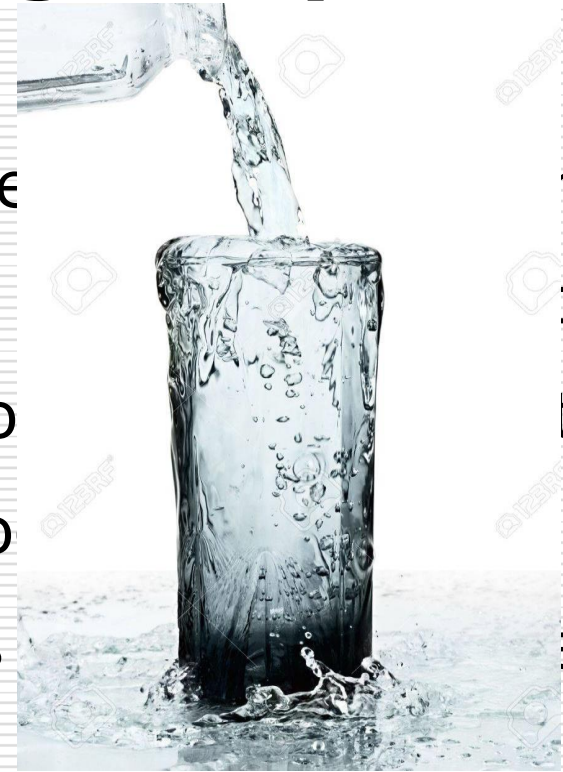
---



# Buffer Overflow (Pourquoi le langage C ?)

---

Beaucoup d'applications écrites en C sont vulnérables car la simplicité et la flexibilité du langage ont prévalu sur les contraintes de sécurité. Ce problème existe également dans les langages de programmation.



# Buffer Overflow

---







Université  
De Boumerdes



Université  
De Limoges

# Attaques sur les applications web



WEB APPLICATION

# Attaques sur les applications web (Sites Web)

---

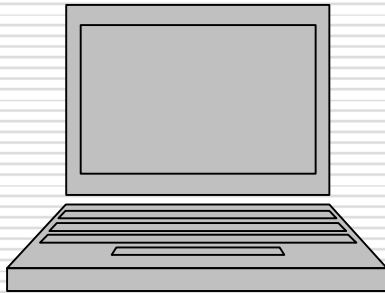
- Le nombre de sites web croît exponentiellement d'année en année.
- On compterait aujourd'hui plus de **1,78 milliard** de sites dans le monde web dans le monde,

**Les sites internet s'exposent naturellement aux attaques qui croient d'année en année.**

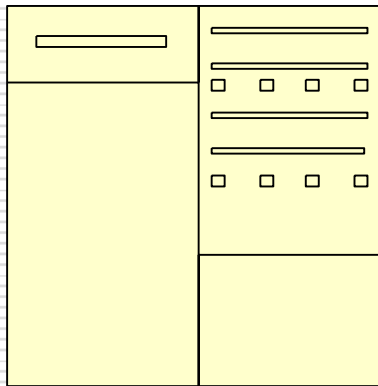


# Attaques sur les applications web (Architecture 3 tiers )

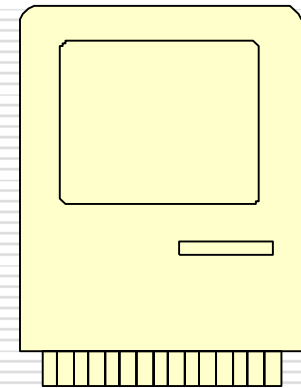
---



Client



Server  
D'application



Serveur de base de  
données

# Attaques sur les applications web (Risques)

---

- Pertes de données par des actes malveillants
- Publication de données confidentielles
- Vol d'identité permettant à un pirate d'avoir un accès administrateur ou sur une zone payante.
- Les abus de ressources qui pourraient ralentir les services du site.
- Détournement de site



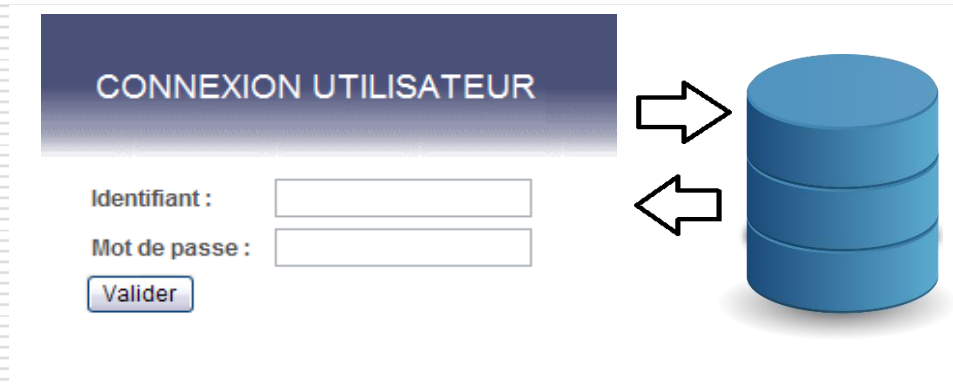
Université  
De Boumerdes

# SQL injection



# SQL injection

- **Définition:** Des méthodes exploitant la faille de sécurité d'une application interagissant avec BDD.



- **But:** injecter dans la requête SQL en cours une portion de requête non prévu par le système qui peut compromettre la sécurité.

---

# Exemple de dégâts



# Exemple de dégâts sql injection

---

**Aout 2013:** Yahoo rapporte le vol de données de plus de 1 milliard de comptes

**Juillet 2012,** Yahoo rapporte le vol de données de plus de 450,000 clients

**Juillet 2010:** des attaques par injection SQL simultanées de Japon et de Chine parviennent à pénétrer la société NeoBeat qui gère des supermarchés sur internet et volent 12 191 données de cartes bancaires.

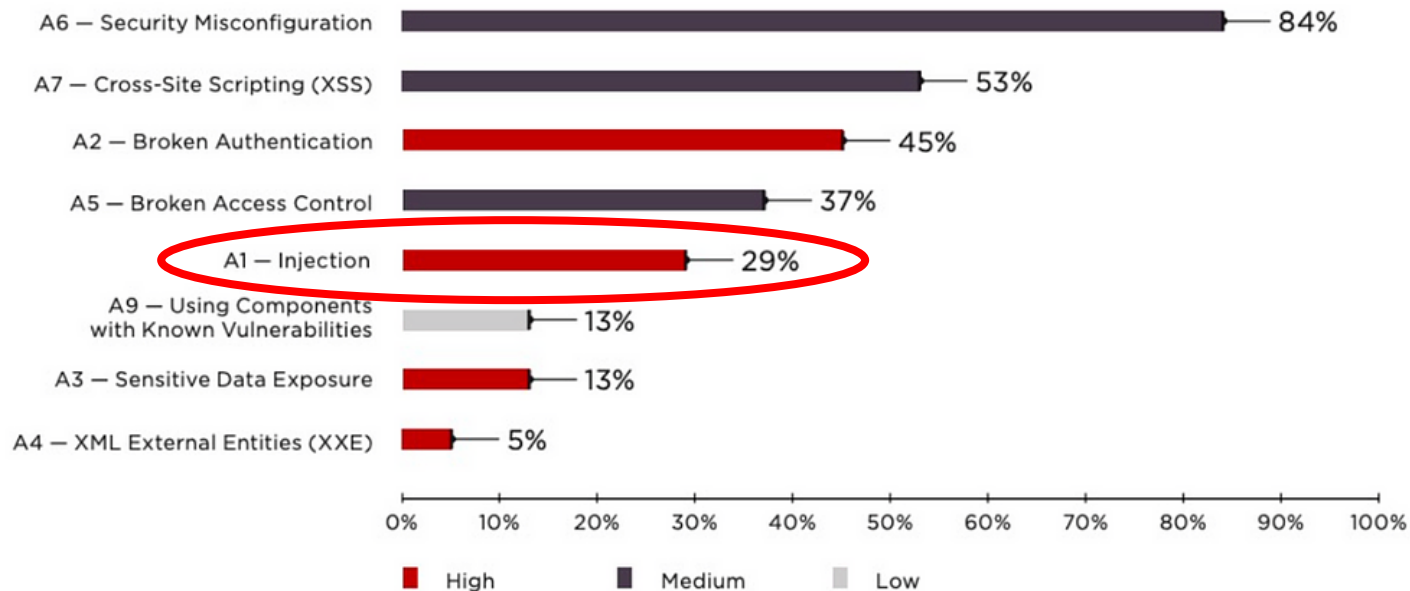
**Avril 2008:** le vol de 10 597 numéros de sécurité sociale américains

...



# SQL injection existe toujours

L'injection SQL est l'une des vulnérabilités les plus anciennes encore présente dans l'OWASP TOP 10.



Selon le rapport (WAAR) d'Imperva publié en Février 2020 : 29% des applications Web, sont encore aujourd'hui vulnérables aux injections SQL

# SQL injection existe toujours

---

## Les cibles actuelles:

- **Les CMS plus populaires:** plus de 5 000 vulnérabilités, dont 4 500 failles d'injection SQL (SQLi) sur plus de 84000 plugins
- une société sud-africaine d'hébergement de sites web, a été victime d'une violation de 40 000 enregistrements de clients.

## Un nouveau type d'injection SQL a fait son apparition en 2019

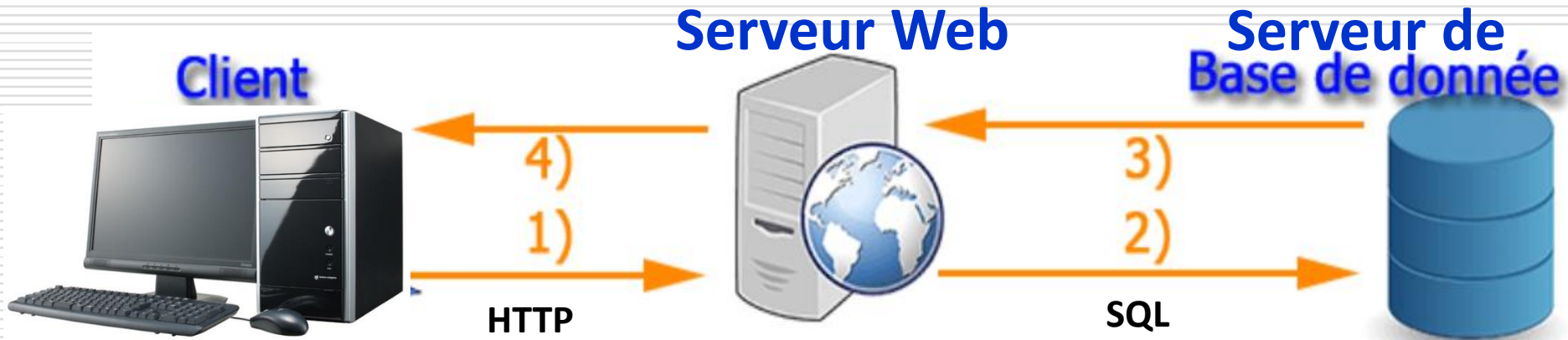
“*Voice-Command SQL Injection*”. basée sur les commandes vocales pour effectuer l'injection.

# SQL injection

- Un pirate peut se connecter à un site en tant qu'administrateur, modifier le prix des produits qu'il achète, tricher à un jeu...etc



- Beaucoup plus courante sur les applications web.



# SQL injection: Simple exemple



*Bienvenue sur Le logiciel de gestion d'un cabinet dentaire*

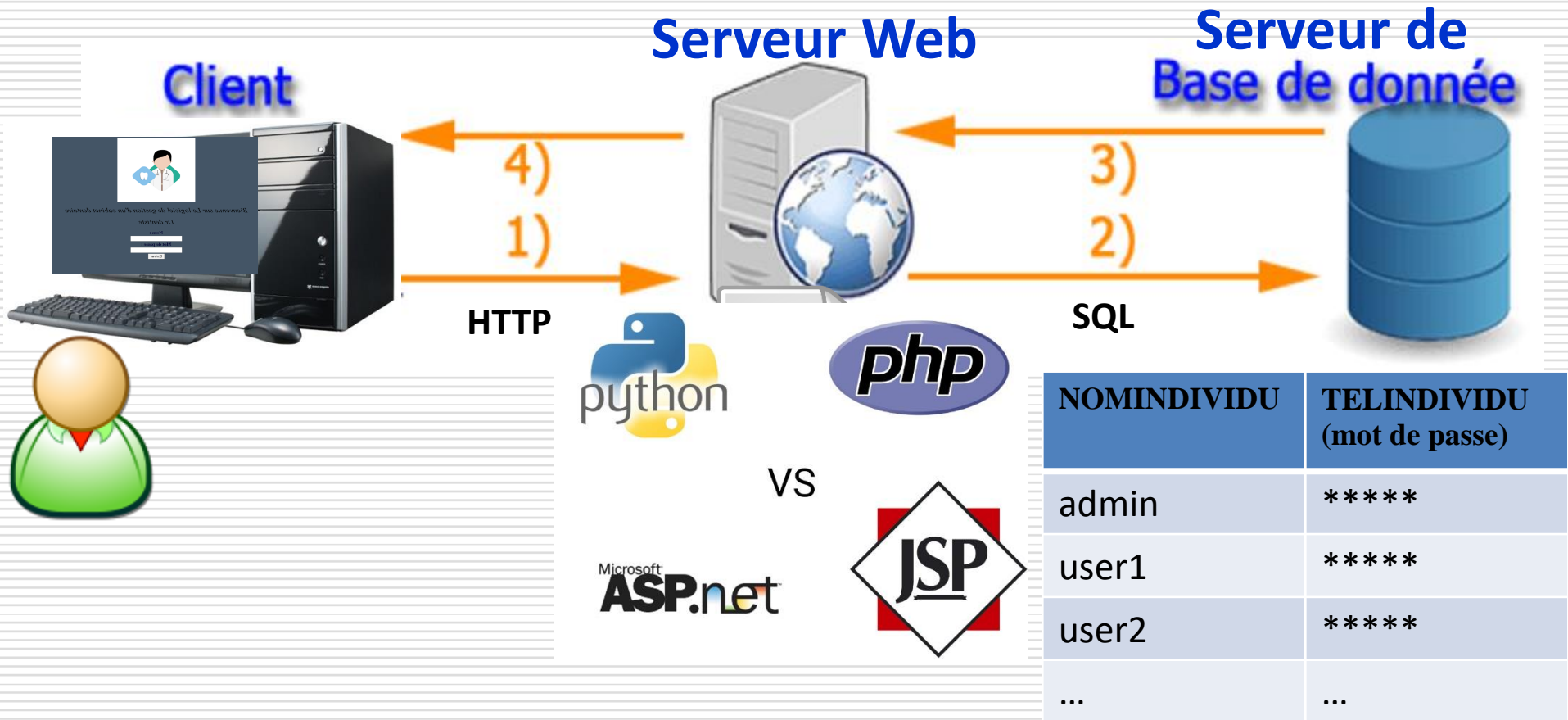
*Dr dentiste*

Nom :

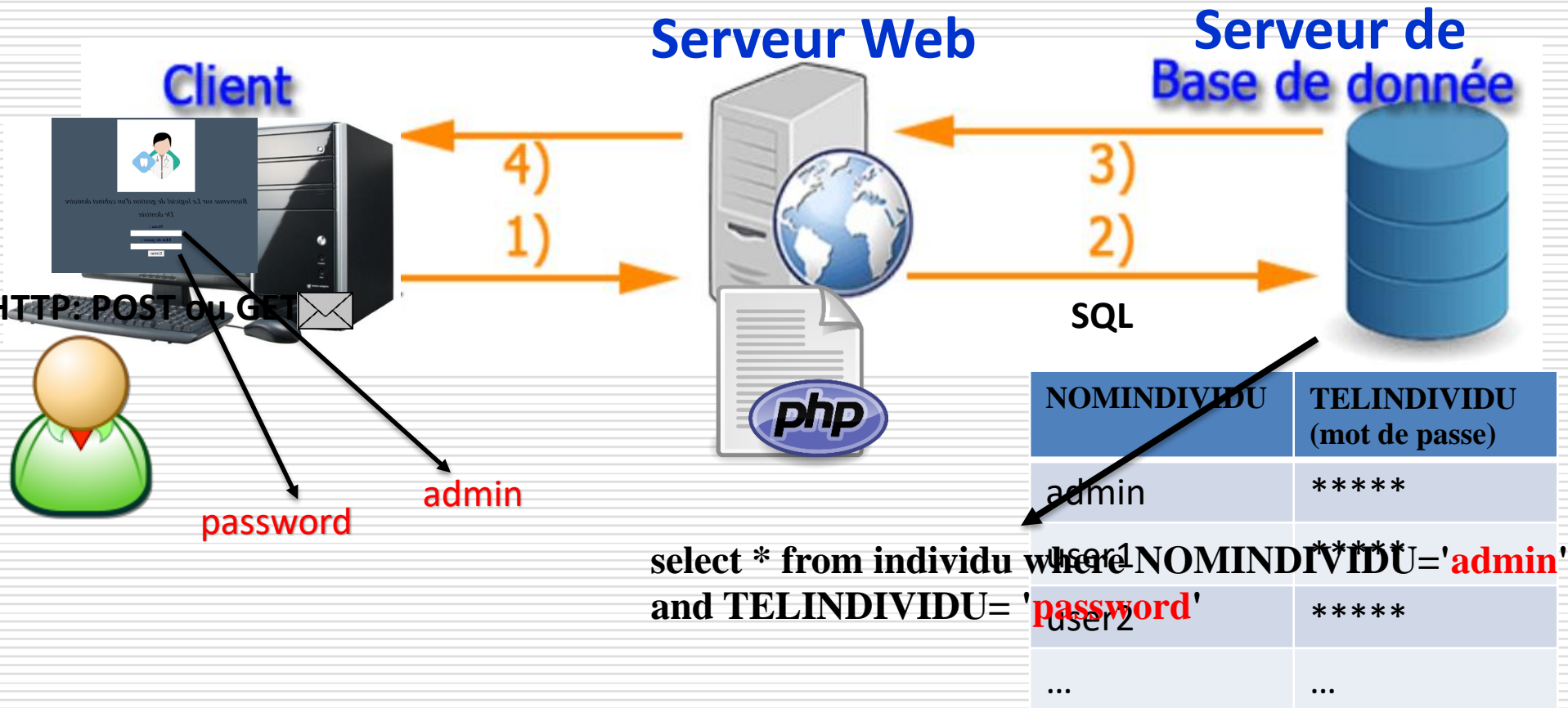
Mot de passe :

Entrer

# SQL injection: Simple exemple



# SQL injection: Simple exemple



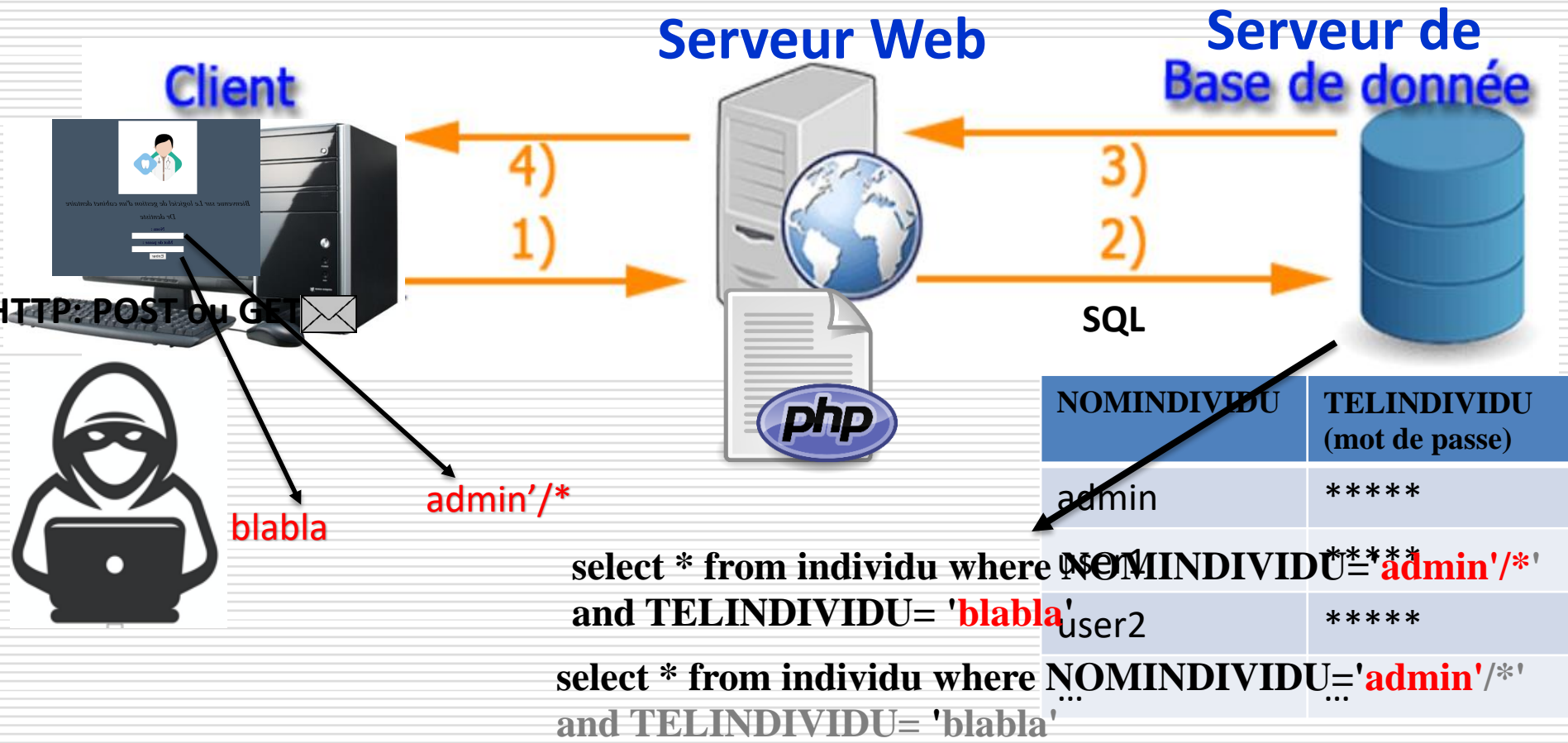
# SQL injection: Simple exemple

---



User: admin  
Mot de passe:??????

# SQL injection: Simple exemple





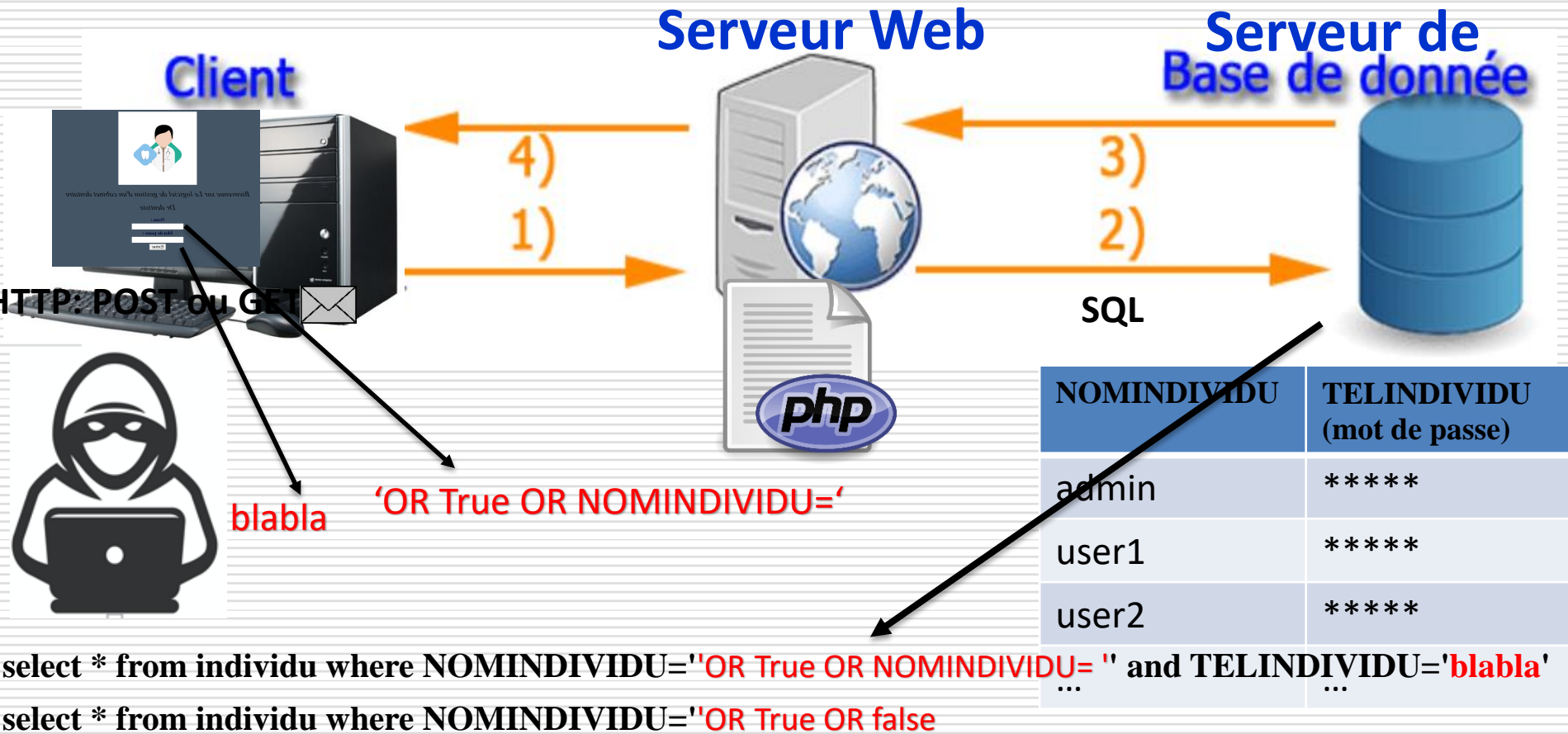
# SQL injection: Simple exemple

---



User: ?????  
Mot de passe:??????

# SQL injection: Simple exemple



# SQL injection: Simple exemple

---



# SQL injection: Simple exemple



*Bienvenue sur Le logiciel de gestion d'un cabinet dentaire*

*Dr dentiste*

Nom :

Mot de passe :

Entrer



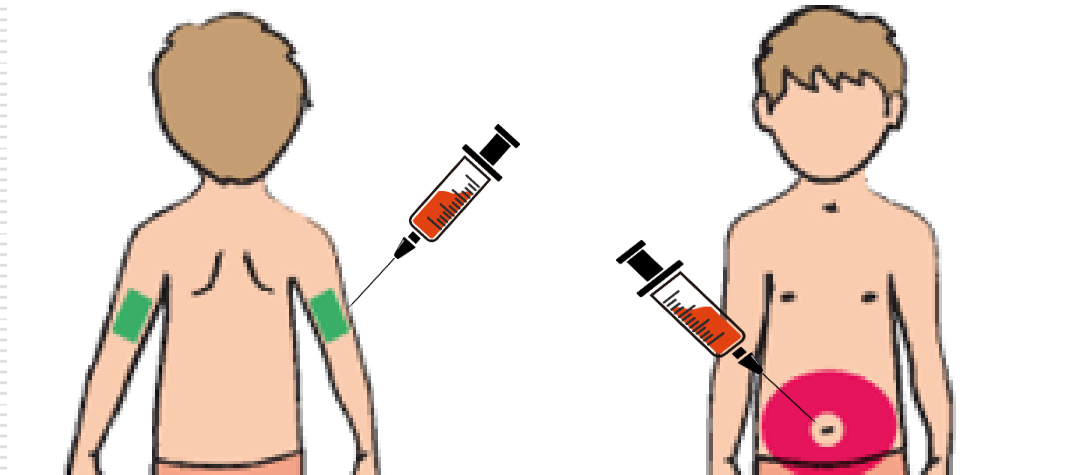
Université  
De Boumerdes

---

# Comment éviter les attaques par injection SQL???

# Petite Précision !

## SQL injection: autres moyens d'injection



# SQL injection: autres moyens d'injection

---



**http://notrecible.com/client.php?id=2**

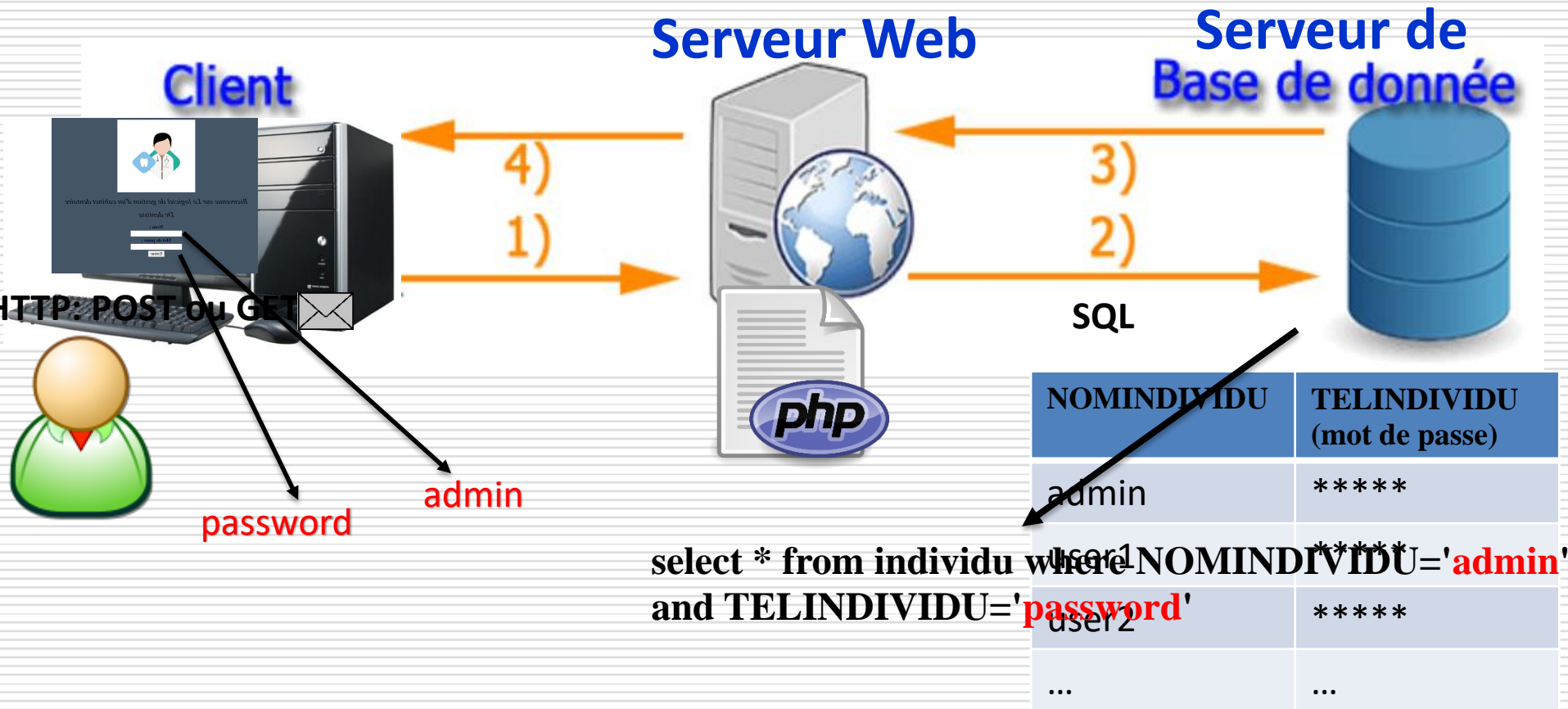
**SELECT nom, mail FROM client WHERE ID = 2**

**http://notrecible.com/client.php?id=2 or 1=1**

**SELECT nom, mail FROM client WHERE ID = 2 or 1=1**

# Remarque importante

## Nom du champ html VS nom du champ BDD





# Outils

---

**SQLMap** est un exemple d'outil permettant d'effectuer des requêtes SQL de manières automatisées dans le but de trouver et d'exploiter ce type de vulnérabilité

# Types d'injections SQL

---

- La méthode *blind based*
- la méthode *time based*
- la méthode *error based*
- la méthode *union based*
- la méthode *Stacked queries*
- blind boolean-based
- out-of-band



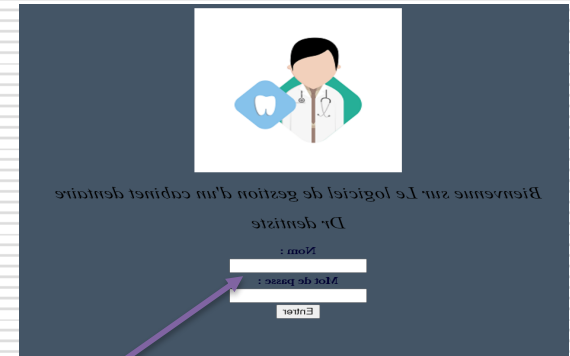
Université  
De Boudjora



Université  
De Limoges

# Comment éviter les attaques par injection SQL???

# SQL injection: Solution



**Il faut Vérifier le format des données saisies.**

# SQL injection: Solution

## Échappement des caractères spéciaux



Utiliser la fonction `mysql_real_escape_string`, qui transformera la chaîne `'/*` en `\/*`

`select * from individu where NOMINDIVIDU='admin\/*' and TELINDIVIDU='blabla'`

L'apostrophe a été dé-spécialisée en la faisant précéder d'un caractère « \ ».

# SQL injection: Solution

## Échappement des caractères spéciaux



**Et si les variables sont numérique!?**

Exemple : `SELECT * FROM DB WHERE num_badge = 0299090902 AND code_4_chiffre = 1098`

## Solutions

Utiliser la fonction `ctype_digit` pour vérifier les variables numériques des requêtes

Forcer la transformation de la variable en nombre en la faisant précéder d'un *transtypreur*,

Exemple (int)

# SQL injection: Solution

## Échappement des caractères spéciaux

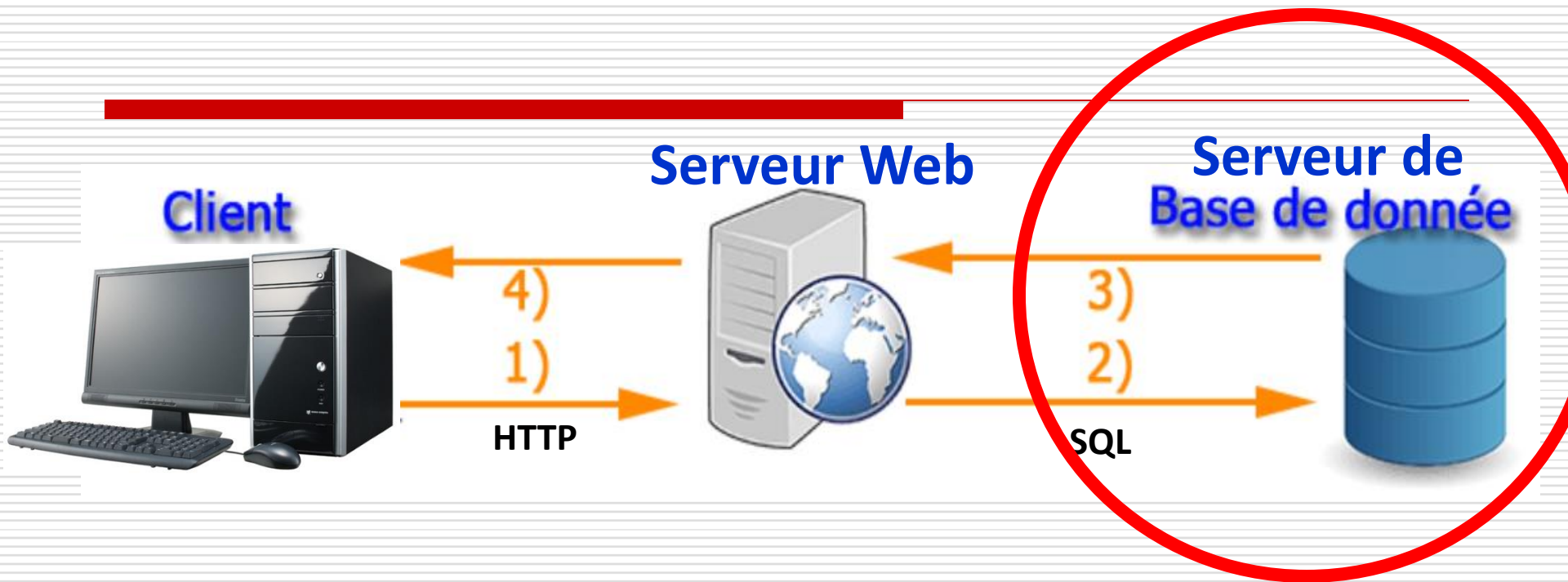
---



- Les « **magic quotes** » étaient utilisées par défaut dans la configuration de PHP.
- Placer un caractère d'échappement automatiquement devant les apostrophes et guillemets dangereux.

Ces options comportent certaines failles donc remplacer par d'autres : la fonction **mysqli\_real\_escape\_string**, les classes PHP Data Objects, etc.

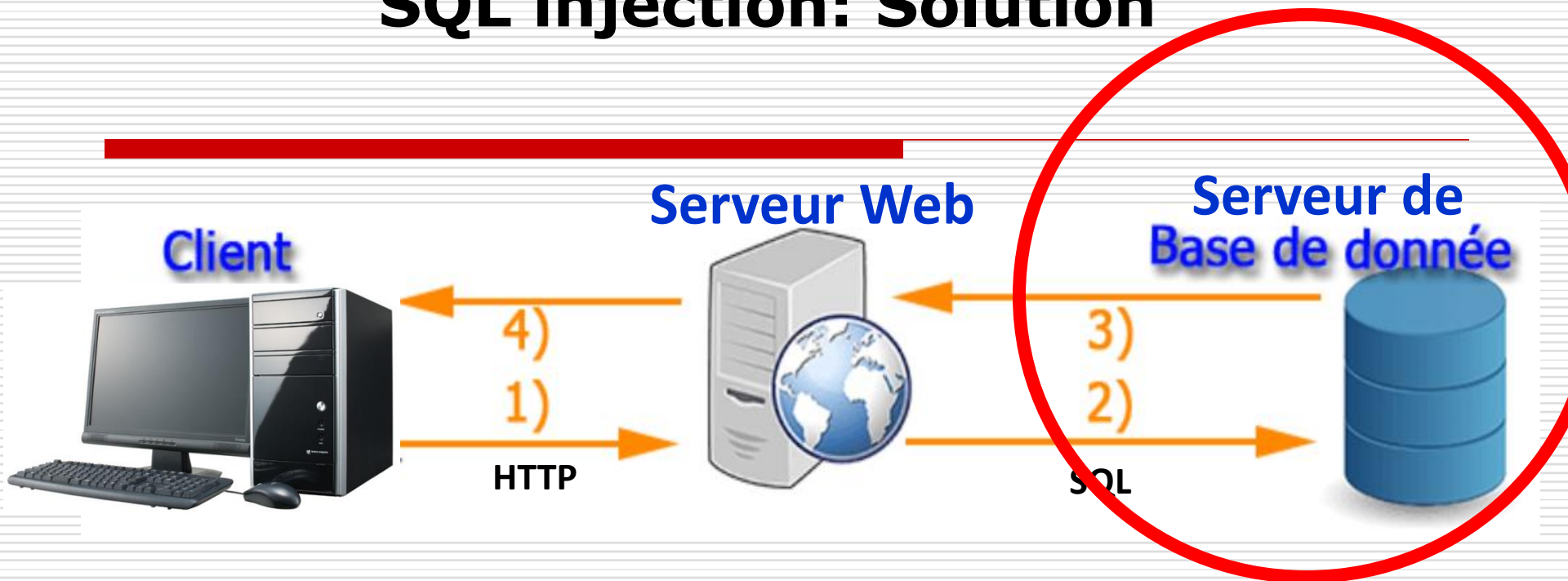
# SQL injection: Solution



- Ne pas afficher la requête ou une partie de la requête.
- Supprimer les comptes users non utilisés et :les comptes par défaut
- Eviter les comptes sans mot de passe.
- Restreindre au minimum les privilèges des comptes utilisés
- Utiliser une **expression rationnelle** pour valider qu'une donnée entrée par l'utilisateur est bien de la forme souhaitée



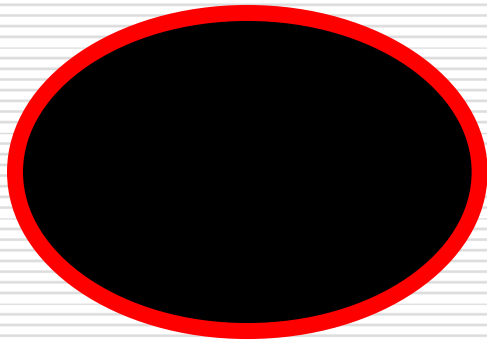
# SQL injection: Solution



- Utiliser des procédures stockées à la place du SQL dynamique (Les données entrées par l'utilisateur sont alors transmises comme paramètres)
- Utiliser des requêtes SQL paramétrées : le SGBD qui se charge d'échapper les caractères selon le type des paramètres.
  - On envoie des requêtes à trous au serveur SQL
  - Puis on lui envoie les paramètres qui boucheront les trous

# SQL injection: Solution requêtes SQL paramétrées

---



**Serveur de Base de données  
SGBD**



Université  
De Boumerdes



Université  
De Limoges

# L'attaque XSS (Cross Site Scripting)



# XSS (Principe)

---

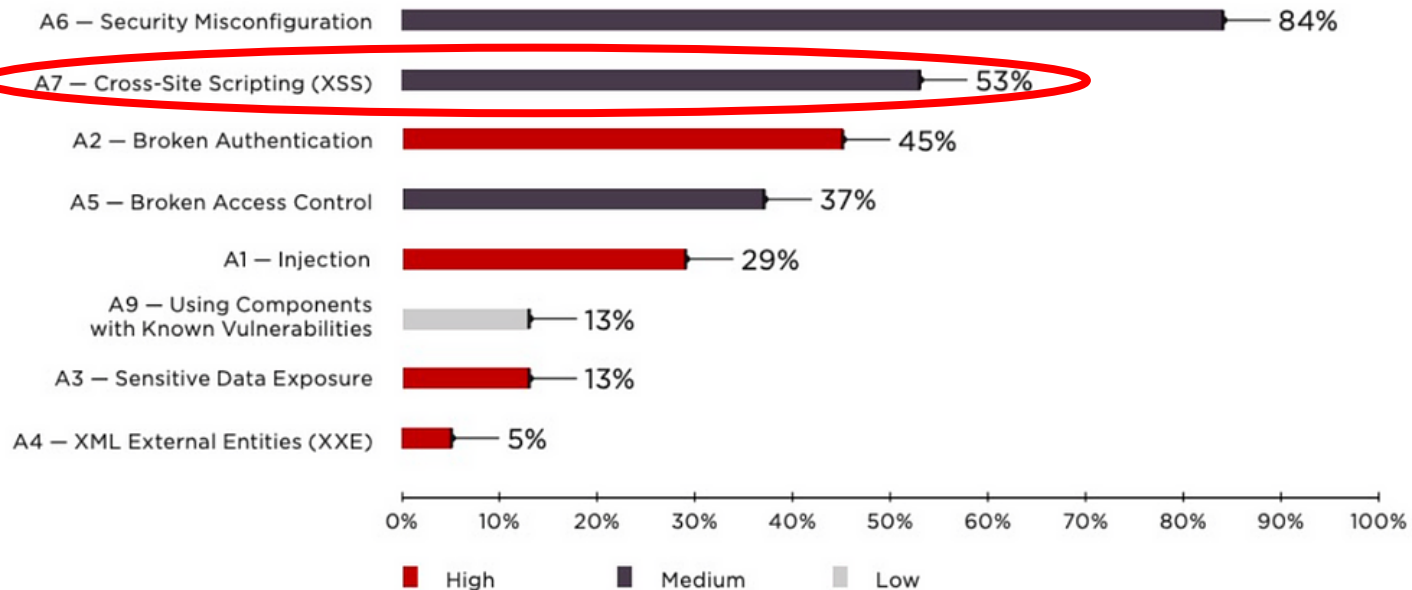
- Utilise des ressources Web pour exécuter des scripts :
  - Dans le navigateur Web de la victime ou
  - Dans une application pouvant être scriptée.



XSS  
Cross Site Scripting

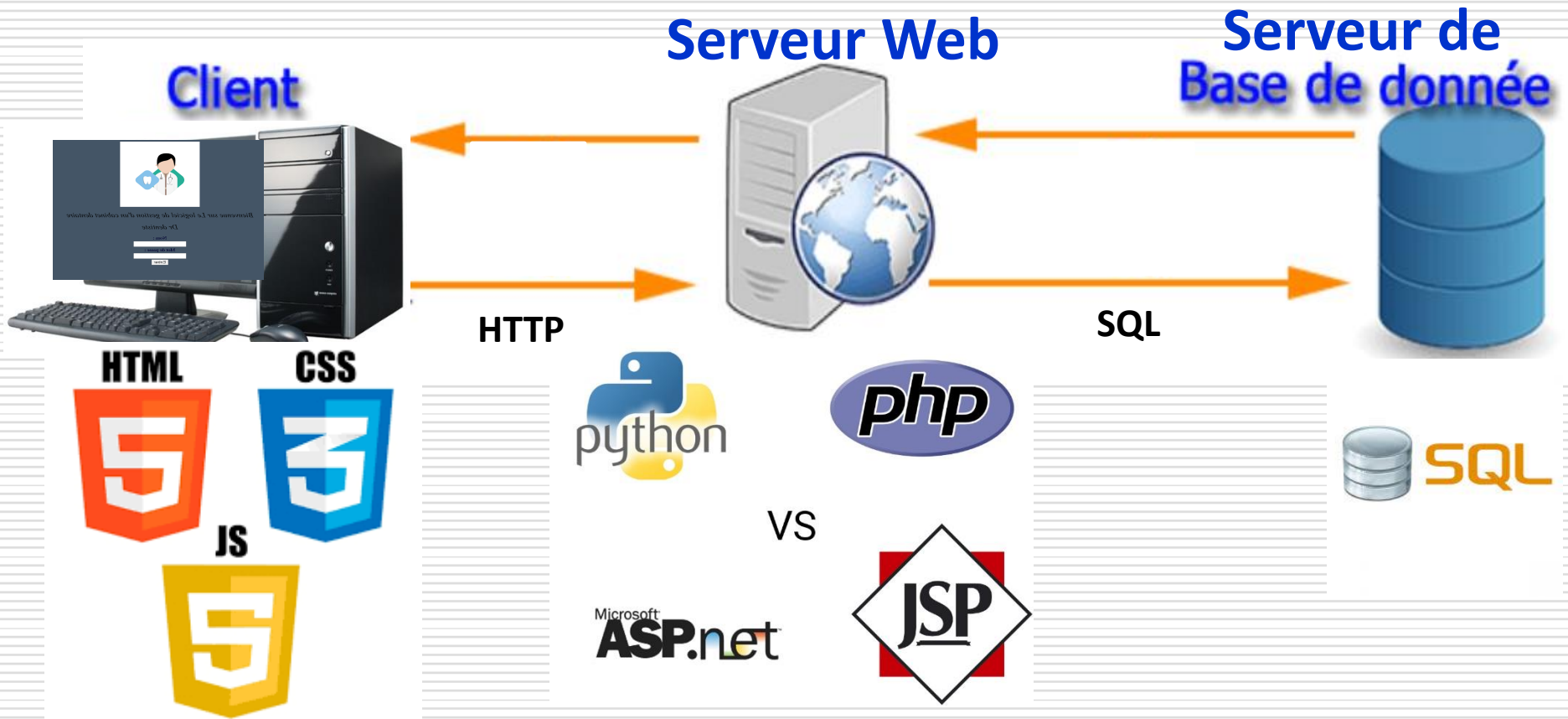
# XSS: Etat actuel

XSS est présente dans l'OWASP TOP 10.



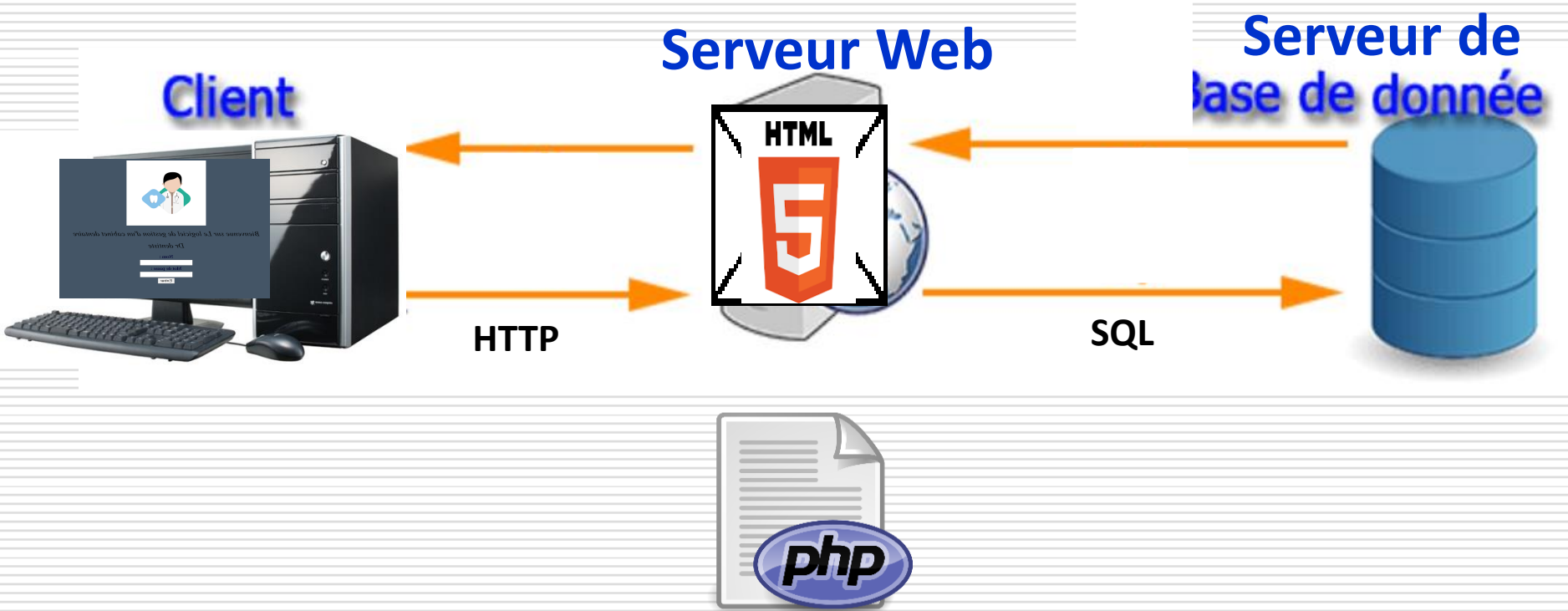
Selon le rapport (WAFAR) développé et publié en Février 2020, 53% des applications Web, sont en période d'urgence vulnérables aux XSS

# Rappel: Les langages de programmation web



# Rappel: Les langages de programmation web

---



# Rappel: Les langages de programmation web

```
<html>
<head>
<script language="javascript">
  alert("Hi there, and welcome.")
</script>
</head>
<body>
</body>
</html>
```

HTML

JavaScript



Hi there, and welcome.

OK

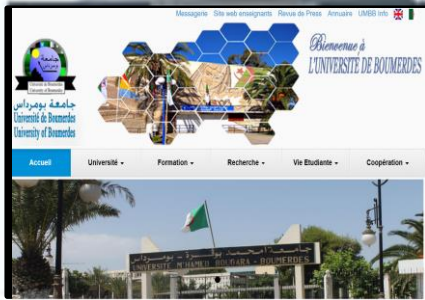


# Pourquoi des langages coté client?



# Rappel: Les langages de programmation web

Client



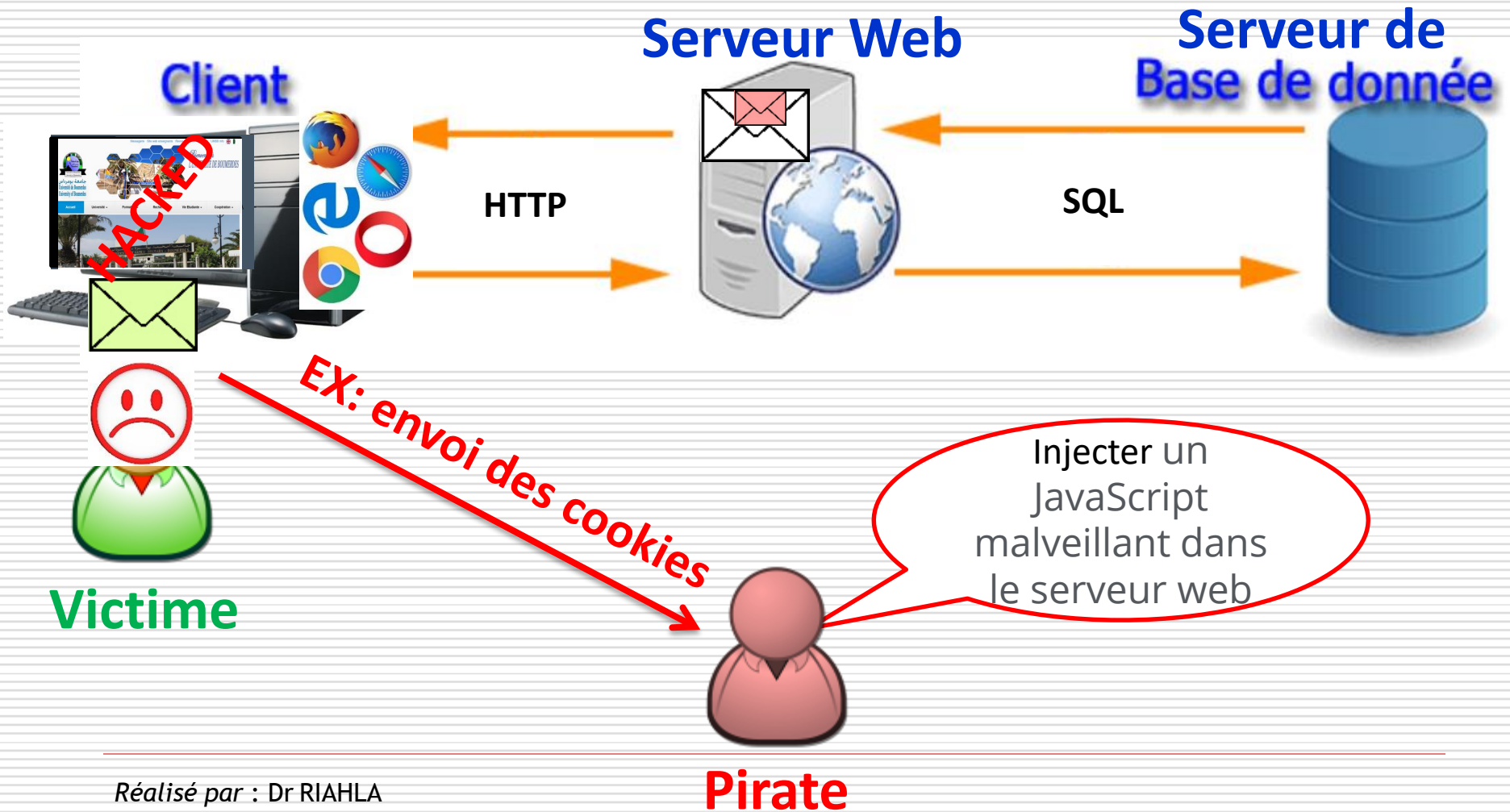
```
<html>
<head>

<script language="javascript">
alert("Hi there, and welcome.")
</script>

</head>
<body>
</body>
</html>
```

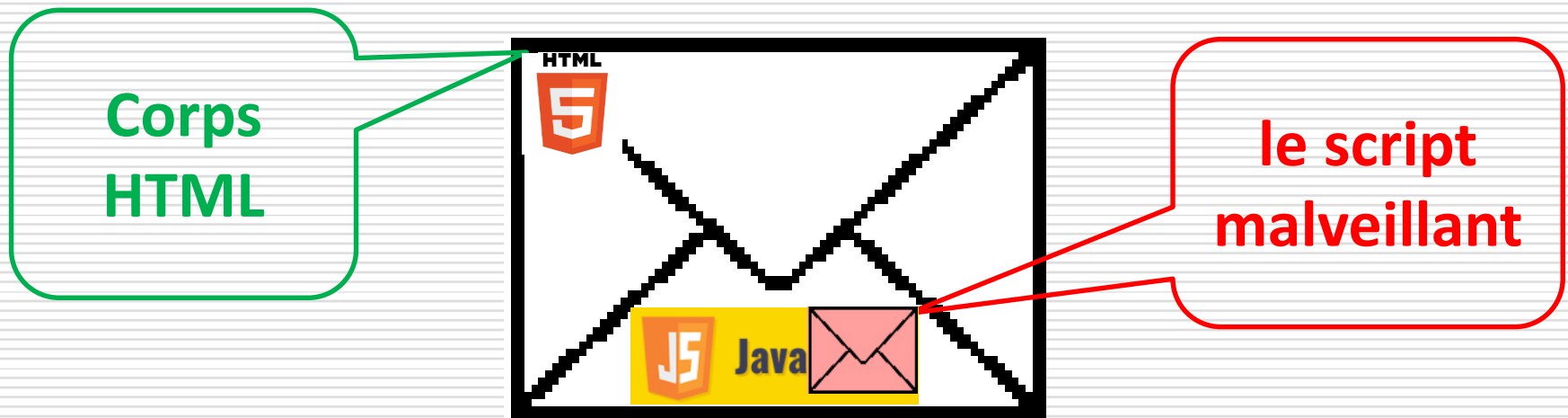
HTML

# XSS: Scénario



# XSS (Emplacement de l'injection)

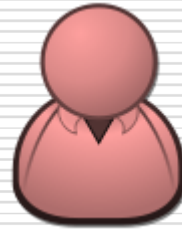
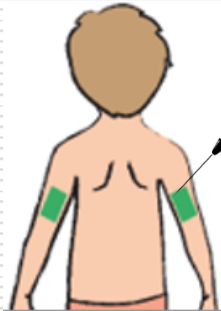
---



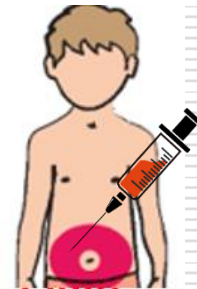
XSS peut être exploité avec **VBScript**, **ActiveX** et **Flash**, mais **JavaScript** est le plus largement touché, en raison de son omniprésence sur le Web.

# XSS (Méthodes d'injection)

L'injection se fait par exemple en déposant un message dans un forum (dans un champ de formulaire) ou par des paramètres d'URL



**Pirate**



Nom
Mail
Message

# XSS (Etapas )

---

1. L'attaquant injecte un JavaScript malveillant dans la BDD d'un site Web.
2. Lorsque la victime demande une page du site Web, le site Web transmet la page à son navigateur avec le script malveillant intégré au corps HTML.
3. Le navigateur de la victime exécute ce script,

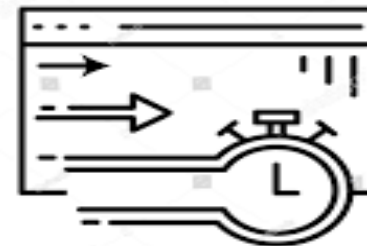
**Exemple du contenu du script:** envoie par exemple le cookie de la victime au serveur de l'attaquant, qui l'extraie et l'utilise pour détourner la session

# XSS (Risque)

- L'attaquant utilise tous les langages pris en charge par le navigateur
- De nouvelles possibilités sont découvertes avec l'arrivée de **HTML5**
- Vol d'informations, par exemple **sessions** et **cookies**.



WEBSITE  
COOKIES



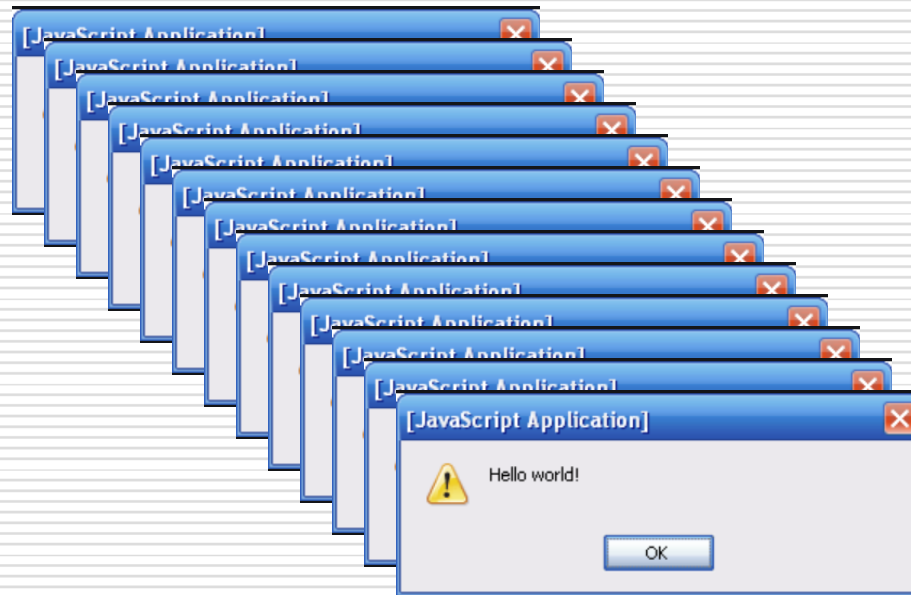
WEBSITE  
SESSION

# XSS (Risque)

---

- Compliquer la lecture d'une page

(**Ex:** boucle infinie d'alertes).





# XSS (Risque)

- Redirection de l'utilisateur (souvent pour un **Phishing**)



# XSS (Risque)

---

- Actions sur le site faillible, à l'insu de la victime et sous son identité (envoi, suppression de messages, ...)
- Un site défiguré, un malware...

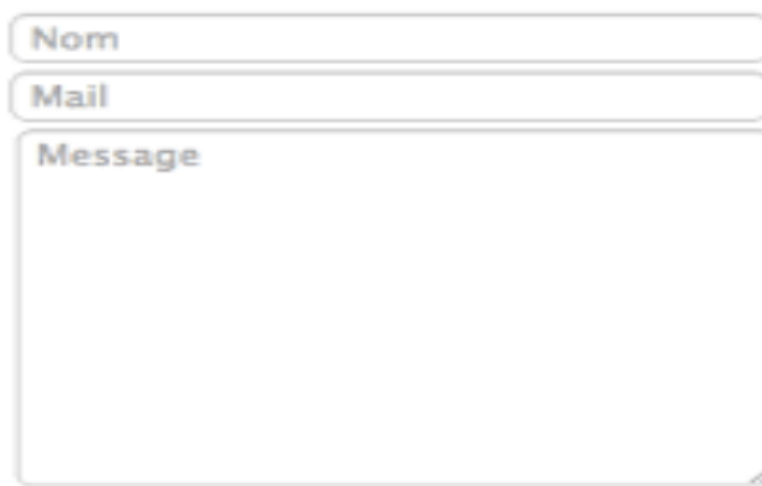
## Les conséquences les plus graves:

- Enregistrer les frappes de touches et des captures d'écran
- Découvrir et de collecter des informations réseau
- Accéder et de contrôler à distance l'ordinateur de la victime.

# Types de failles XSS

---

- Attaques XSS stockées (stored XSS attacks),
  - ✓ Contenu malicieux stocké dans une BDD du serveur
  - ✓ Infection via forum,...etc
  - ✓ Aucune interaction avec la victime n'est nécessaire
  - ✓ **Vise plusieurs victimes**



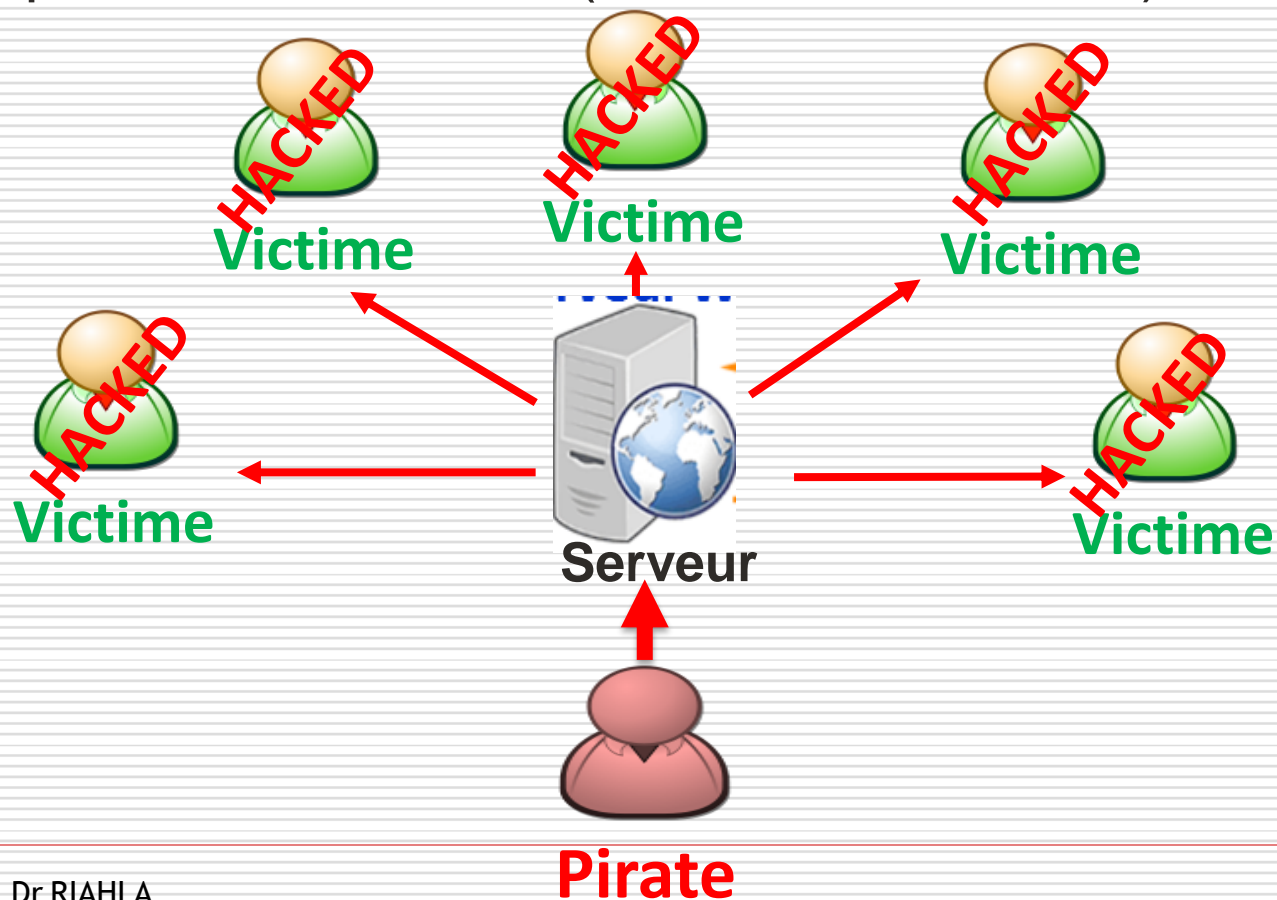
Nom

Mail

Message

# Types de failles XSS

- Attaques XSS stockées (stored XSS attacks),



# Types de failles XSS

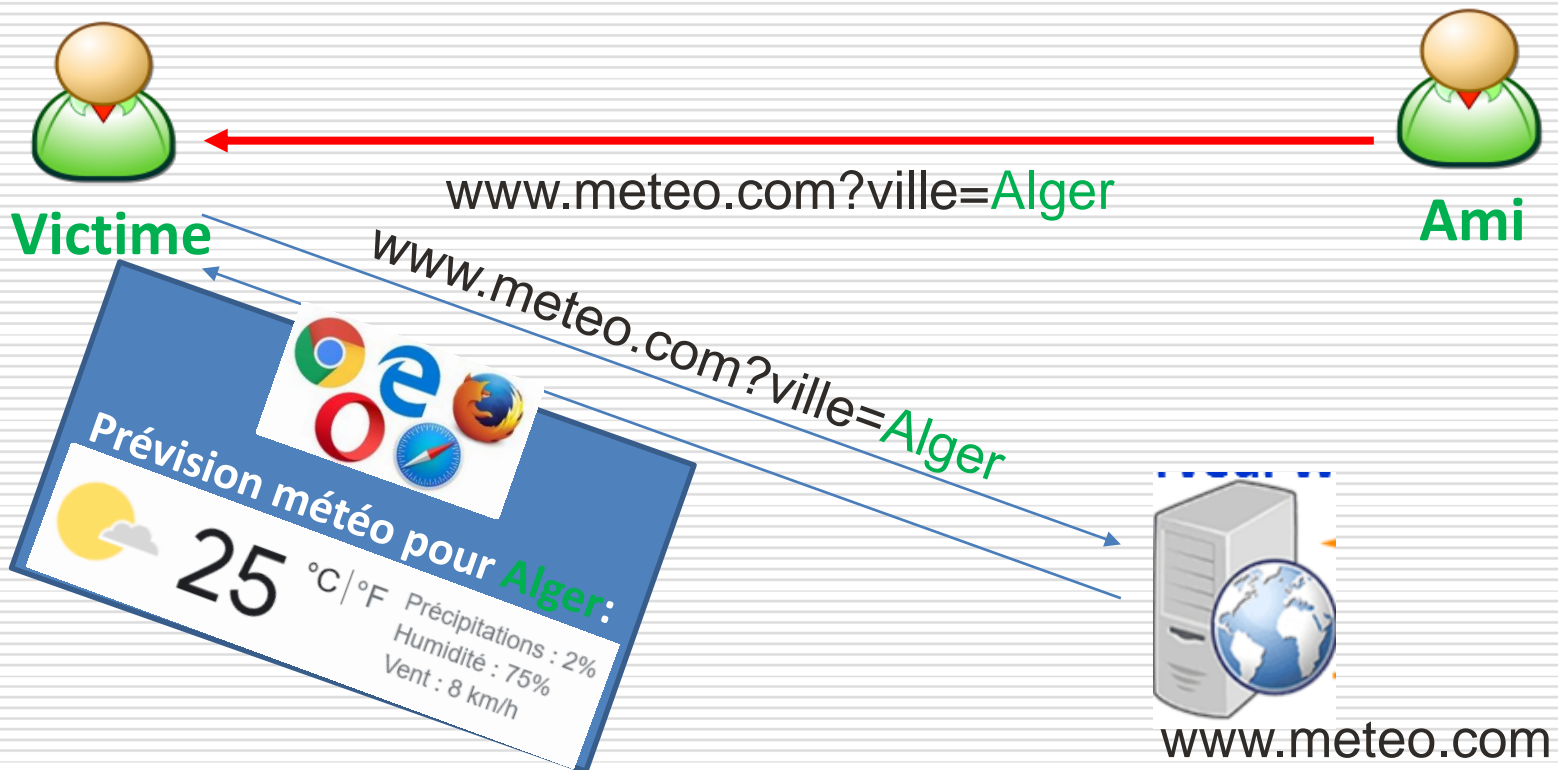
---

- Attaques XSS reflétées (reflected XSS attacks),
  - ✓ Ne stocke pas le contenu malicieux
  - ✓ Le contenu est par exemple livré à la victime via une URL (**exemple**: envoyée par email « **Phishing** »)
  - ✓ **Vise une seule victime**



# Types de failles XSS

- Attaques XSS reflétées (reflected XSS attacks),



# Types de failles XSS

- Attaques XSS reflétées (reflected XSS attacks),



# Types de failles XSS

- Attaques XSS basées sur le DOM (DOM based XSS).
  - DOM est une structure utilisée pour représenter un document dans un navigateur
  - Nécessite 'ou non' une interaction de la part de la victime
  - Elle n'utilise pas le serveur web



**Victime**



**Pirate**



**Serveur Web**



# Types de failles XSS

---

- Attaques XSS basées sur le DOM (DOM based XSS).
  - présente surtout dans les application « web 2.0 »
  - Une bonne quantité de code Javascript est exécutée dans le navigateur de l'utilisateur, dans interaction avec le serveur.

Contrairement aux deux versions précédentes où le contenu était envoyé au serveur avant d'être retourné à la victime, **les attaques DOM XSS se passent directement dans le navigateur de la victime.**

# XSS en pratique

---



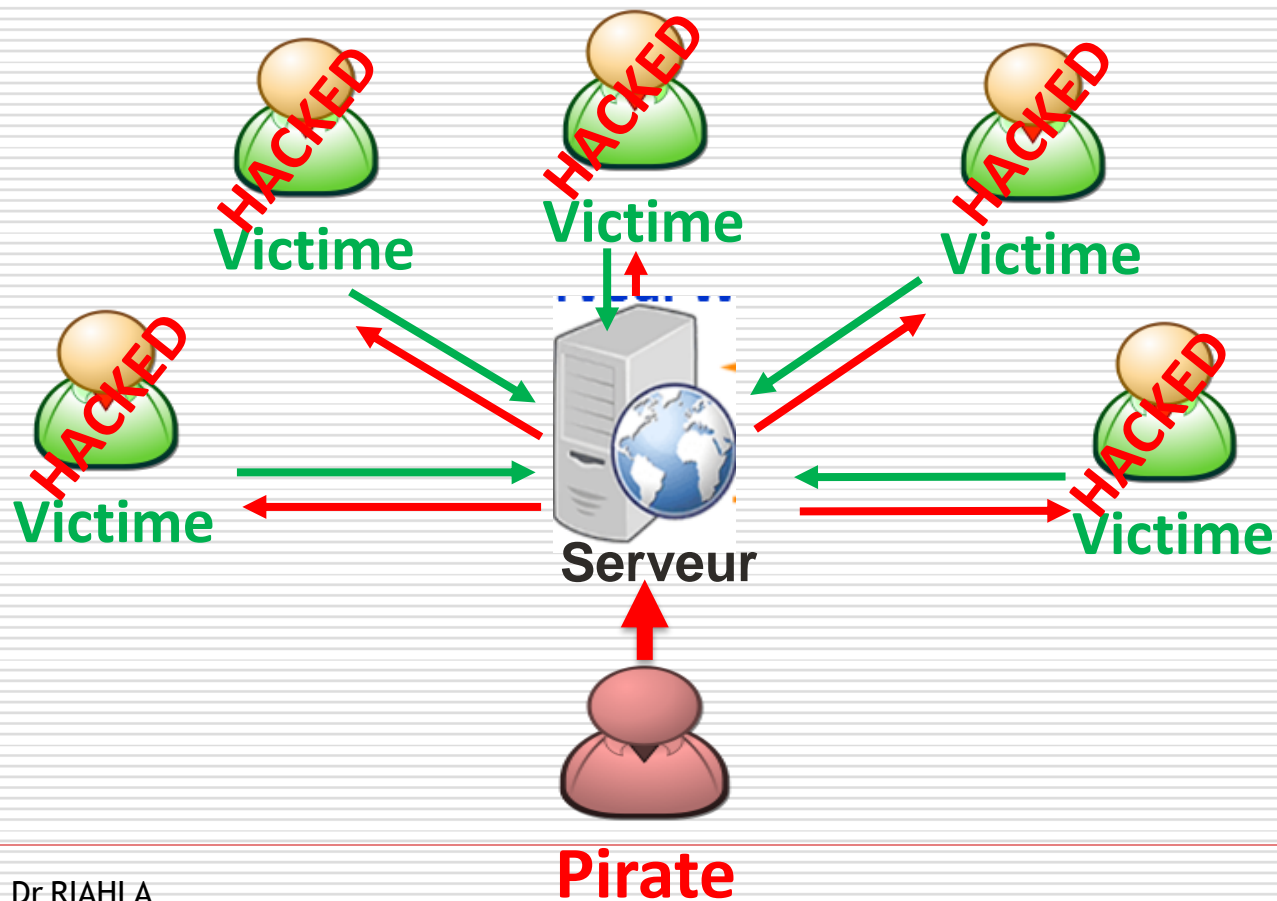
# Attaques XSS stockées (stored XSS attacks)

---

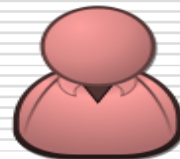
- ✓ Contenu malicieux stocké dans une BDD du serveur
- ✓ Infection via forum,...etc
- ✓ Aucune interaction avec la victime n'est nécessaire
- ✓ Vise plusieurs victimes


# Attaques XSS stockées (stored XSS attacks)

---



# Attaques XSS stockées (stored XSS attacks)



Nom	
Mail	
Message	<div> JavaScript</div>

---

# **XSS (Contre mesure)**

# XSS (Contre mesure)

---

- Nettoyer les données entrées par les utilisateurs dans leurs requêtes HTTP avant de les renvoyer aux autres.
- Convertir les caractères spéciaux tels que ?, &, /, <, > et les espaces en leurs équivalents codés HTML ou URL.
- Offrir aux users la possibilité de désactiver les scripts côté client.



Université  
De Boumerdes

---

# Autres Attaques Web



# Autres attaques

---

- ❑ Les fichiers générés en cache
- ❑ Publication de code source
- ❑ Les fichiers logs
- ❑ L'inclusion en PHP
- ❑ le CSRF