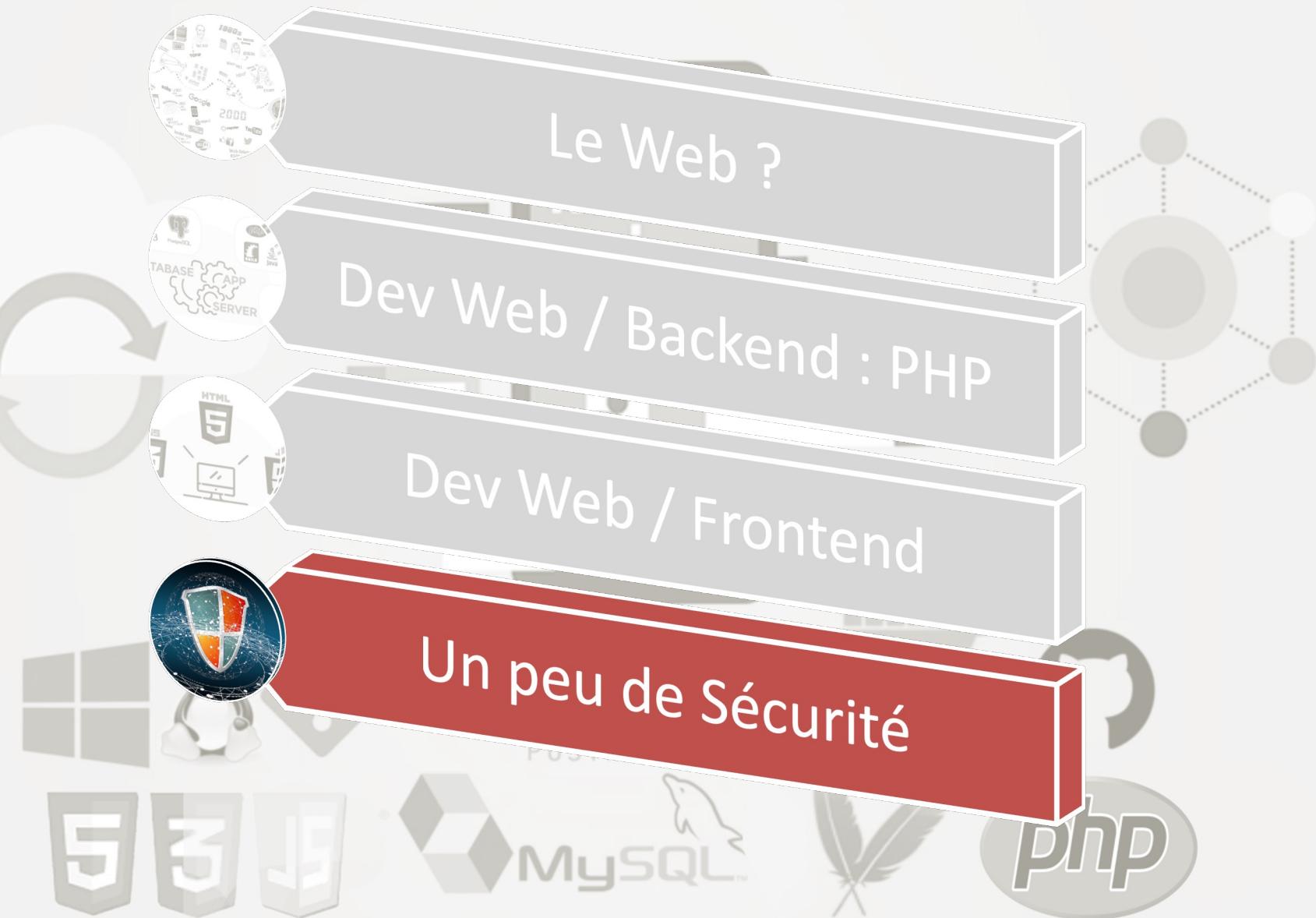


WEB 1 : Initiation à la programmation WEB



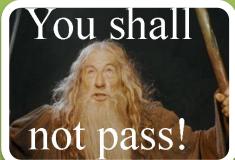


Dev Web / Sécurité des sites Web (qq éléments)

Dev Web / Backend : Plan



La fin de l'insouciance



Accès aux ressources (indiscrétion)



Injection



Man in the Middle (interception)



Intrusion serveur et Vol de Bases



Dev Web / Sécurité : **La fin de l'insouciance**

Dev Web / Sécurité : La fin de l'insouciance



DANS ♡ UN ♡ MONDE ♡ PARFAIT ♡

Les entrées utilisateur sont toujours celles attendues

Personne n'a de mauvaises intentions

Vous avez envisagé et géré toutes les erreurs possibles

Qui s'intéresse aux identifiants et données personnelles d'autres utilisateurs ?



Dev Web / Sécurité : La fin de l'insouciance

Prévoir ...



Des entrées utilisateurs invalides



Des utilisateurs mal intentionnés / incompétents



Que tout peut / va mal se passer



Que l'on cherche à vous avoir
(*Espions, Scripts, CIA* ☺)

Dev Web / Sécurité : La fin de l'insouciance

On va faire un jeu ...



Dev Web / Sécurité : La fin de l'insouciance

Pourquoi faire ?

Voler / Lire des données

Corrompre / Modifier
des données

Corrompre / Planter un site

Endommager un site

Espionner /
Transmettre
des infos

Propager
Virus / Pub

Spoofing
(se faire passer
pour ...)

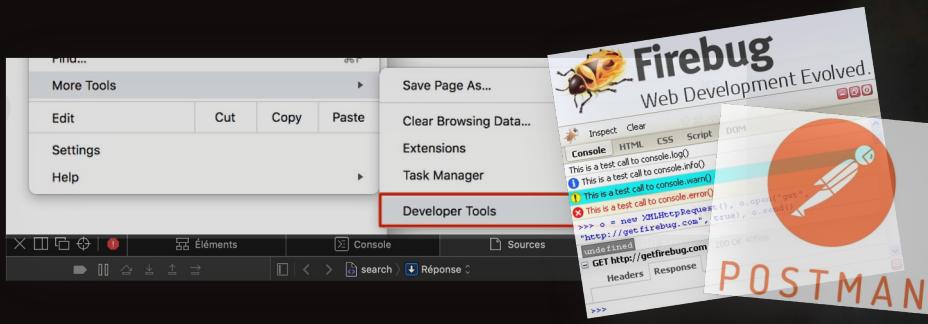


Dev Web / Sécurité : La fin de l'insouciance

Avec quels outils ?

Les même que vous ☺

Outils client

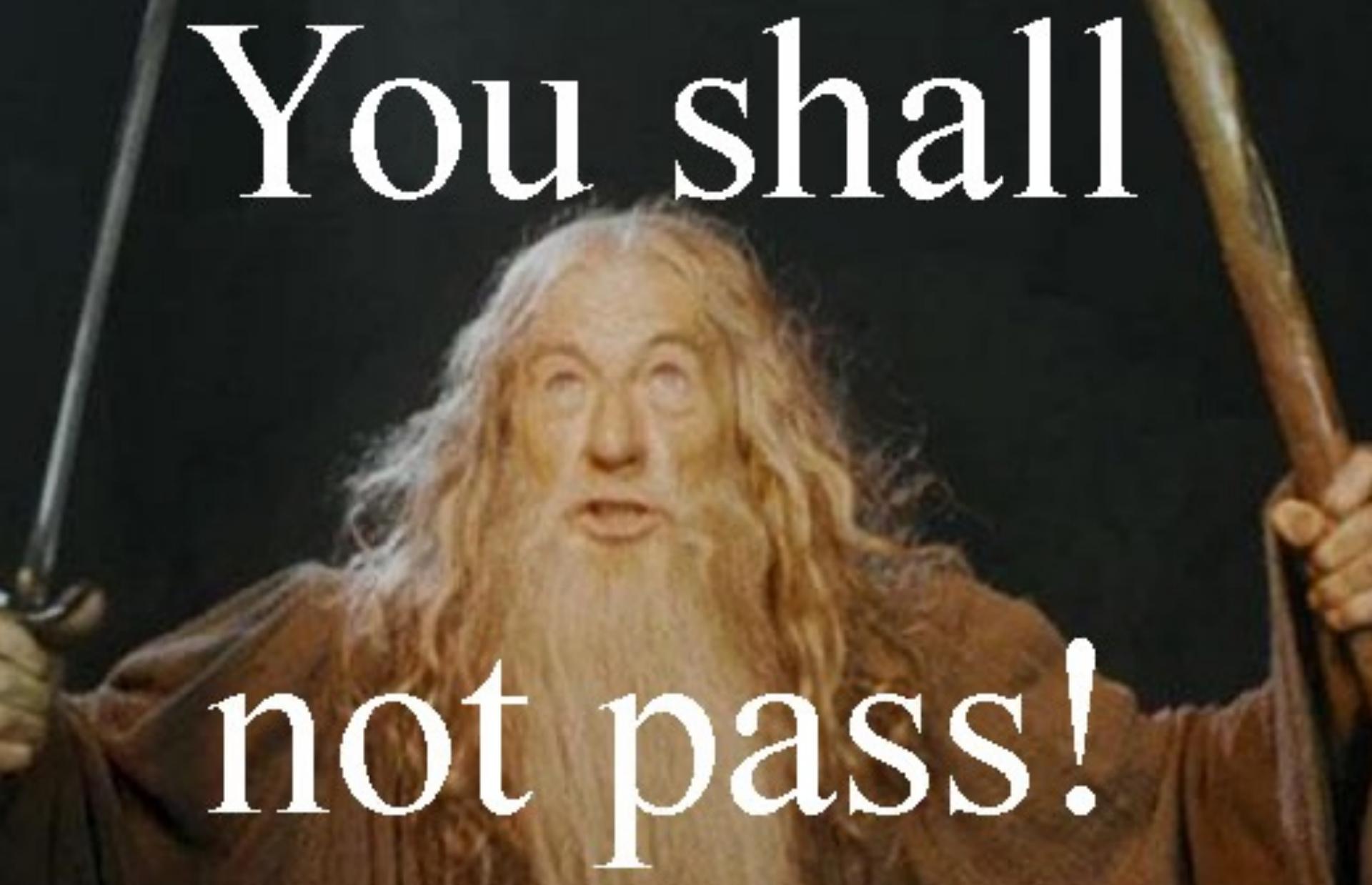


Outils réseau



Renifleurs



A close-up portrait of Gandalf the Grey from the Lord of the Rings movies. He has long, flowing white hair and a long, bushy grey beard. He is wearing a brown robe and holding a long, thin staff in his right hand. His expression is stern and commanding. The background is dark and out of focus.

You shall
not pass!

Dev Web / Sécurité : Accès aux ressources (indiscrétion)



Des utilisateurs mal intentionnés ou curieux

peuvent :

Lire le code source des pages HTML et accéder aux ressources du site

```
<body id="myPage" data-spy="scroll" data-target=".navbar" data-offset="50">
    <nav class="navbar navbar-default navbar-fixed-top">
        <div class="container-fluid" >
            <div class="navbar-header" >
                <table>
                    <tr>
                        <td></td>
                        <td><h1>&ampnbsp&ampnbspGoats</h1></td>
                    </tr>
                </table>
            </div>
        </div>
    </nav>
</body>
```

« Aspirer » le site

Dev Web / Sécurité : Accès aux ressources (indiscrétion)



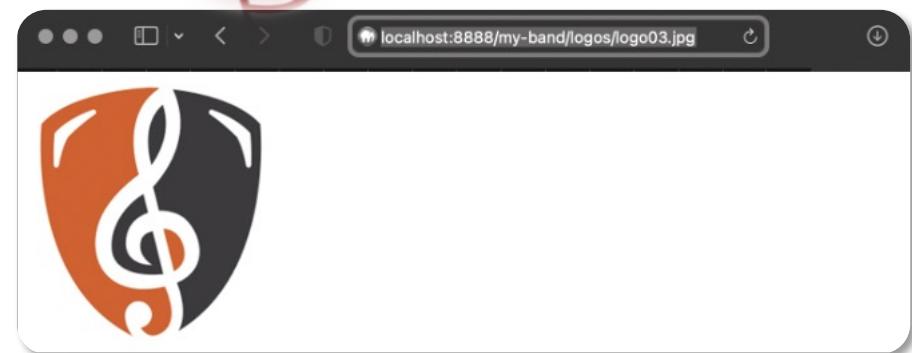
Sur Apache → .htaccess

Va vous permettre de restreindre les accès

Sur les dossiers

: En empêchant le listing

```
EXPLORATEUR ... .htaccess X
MY-BAND logos > .htaccess
logos Options -Indexes
logo01.jpg
logo02.jpg
logo03.jpg
logo04.jpg
```



Dev Web / Sécurité : Accès aux ressources (indiscrétion)



Sur Apache → .htaccess

Va vous permettre de restreindre les accès

Sur tous les fichiers d'un dossier :

EXPLORATEUR

MY-BAND

logos

.htaccess

logo01.jpg

logo02.jpg

logo03.jpg

htaccess

htaccess

logos > .htaccess

1 Options -Indexes

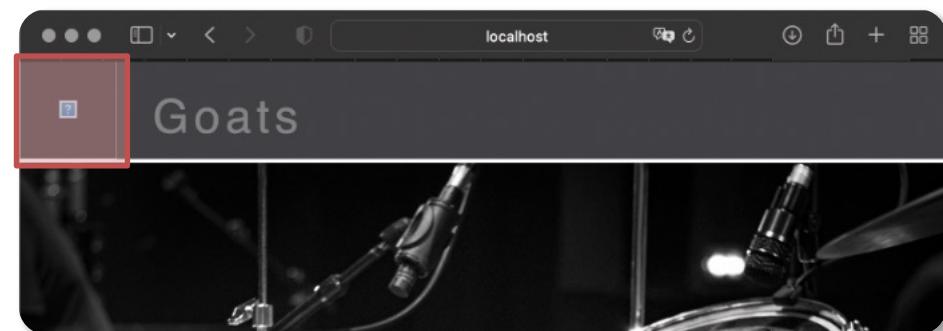
2 deny from all



Oui mais



```
<td></td>
<td>&ampnbsp&ampnbspGoats</td>
<tr>
```



Dev Web / Sécurité : Accès aux ressources (indiscrétion)



Sur Apache → .htaccess

Va vous permettre de restreindre les accès

Sur tous les fichiers d'un dossier :



: Générer l'image côté serveur → « logo.php »

```
EXPLORATEUR ... .htaccess ● logo.php ✎
MY-BAND /+ ⏺ ↻
> logos
> photos
band_generators.php
contact.php
dbconnect.php
footer.php
header.php
index.php
logo.php
```

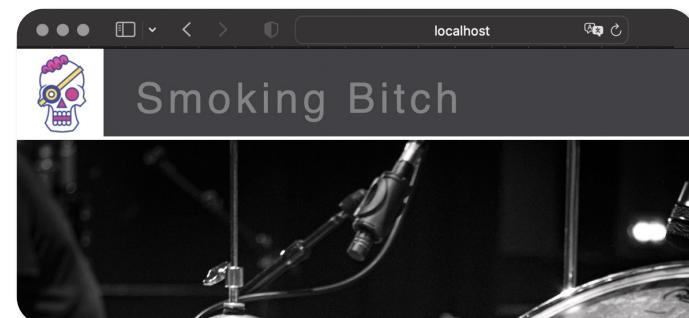
File contents of logo.php:

```
/Applications/MAMP/htdocs/my-band/
logos/.htaccess
1 $LOGOS_DIR = "./logos";
2 $logo_img = $_GET["logo"];
3
4
5 echo file_get_contents($LOGOS_DIR."/".$logo_img);
6 ?>
```

Nom du fichier image en GET :
e.g. logo03.jpeg

file_get_contents : Renvoi le contenu binaire d'une fichier

```
<td></td>
<td><h1>&ampnbsp&ampnbsp<?php echo $_SESSION['bandname'] ;?></h1></td>
</tr>
```





Sur Apache → .htaccess

Va vous permettre de restreindre les accès



Fonctionne avec tout type de ressource
(images, document)



Dev Web / Sécurité : Injection

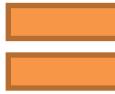
Dev Web / Sécurité : Injection



Des entrées utilisateurs invalides



Des utilisateurs mal intentionnés



Possibilité d'injection :



Def. : Utilisation des entrées utilisateurs pour « injecter » des données et comportements non prévus / malveillants

XSS Injection

(cross site scripting)

XSS Injection (cross site scripting)

SQL Injection



Dev Web / Sécurité : Injection

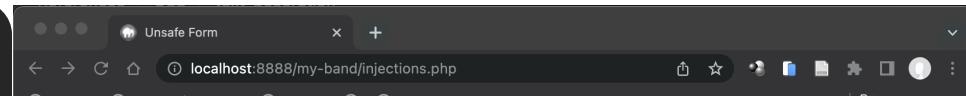
XSS Injection

(cross site scripting)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8"/>
    <title>Unsafe Form</title>
  </head>

  <body>
    <h1>Qui êtes vous ?</h1>
    <form action="injections.php" method="get">
      Tapez votre nom :
      <input type="text" name="name" size="60" />
      <input type="submit" value="Valider" />
    </form>
    <br/>
    <h1>Votre nom est :</h1>
    <?php
      if (isset($_GET["name"])){
        echo $_GET["name"];
      } ?>
    </h1>
  </body>
</html>
```

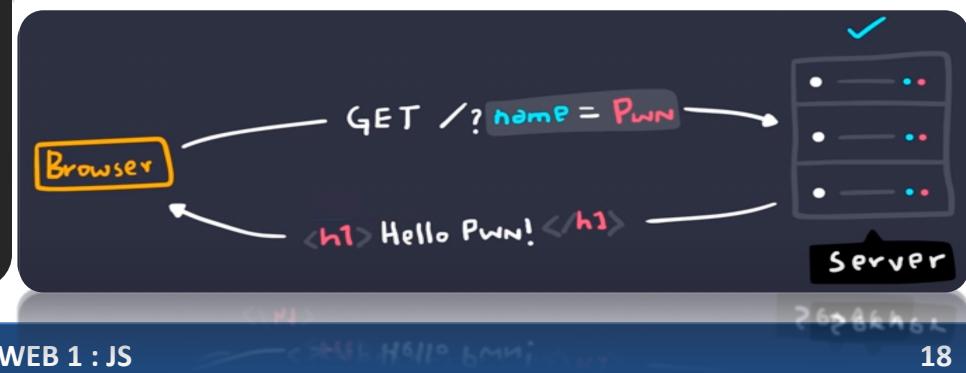
Exemple jouet



Qui êtes vous ?

Tapez votre nom : Valide

Votre nom est :



Dev Web / Sécurité : Injection

XSS Injection

(cross site scripting)

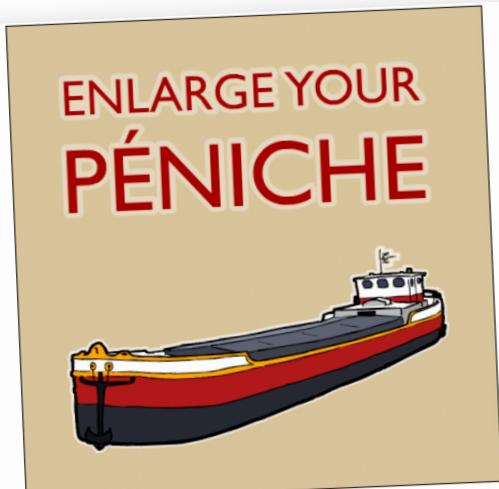
Exemple jouet



I'M SO SCARED!!!

Social Hacking

Phishing



Cliquez ici

Ouais mais M'sieur, ça marche que si l'utilisateur lui-même insère lui-même l'entrée corrompue !

```
<body align="center">
    <br/>
    <h1 style="color:red; text-decoration:blink;">
<a href="http://localhost:8888/my-band/injections.php?name=%3Cdiv+s
    &lt;/h1&gt;</pre>
```

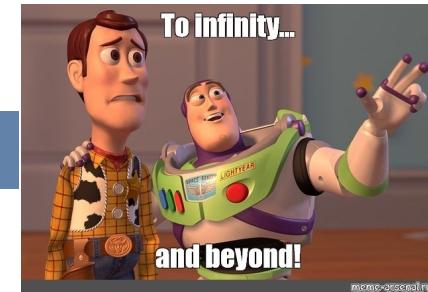


Dev Web / Sécurité : Injection

XSS Injection

(cross site scripting)

Les possibilités sont infinies



- Rediriger les utilisateurs vers un site Web malveillant.
 - Enregistrer les frappes de l'utilisateur sur le clavier.
 - Accéder à l'historique de navigation de l'utilisateur et au contenu des presse-papiers.
 - Exécuter des attaques basées sur un navigateur Web (comme planter le navigateur).
 - Obtenir les informations sur les cookies d'un utilisateur qui est connecté à un site Web.
 - Voler le jeton de session de connexion, permettant à l'attaquant d'interagir avec l'application comme la victime, sans avoir à connaître son mot de passe.
 - Forcer l'utilisateur à envoyer des requêtes à un serveur, contrôlées par l'attaquant.
 - Modifier le contenu de la page.
 - Piéger la victime pour qu'elle divulgue son mot de passe pour accéder à l'application ou d'autres applications.
 - Infecter la victime avec d'autres codes malveillants en utilisant une vulnérabilité du navigateur lui-même, voire prendre le contrôle de l'ordinateur de la victime.

Dev Web / Sécurité : Injection

XSS Injection
(cross site scripting)

3 Types d'attaques XSS :



Attaques XSS reflétées (XSS non persistant) :

Celle qu'on vient de voir. Les attaquants utilisent des liens malveillants, des emails de phishing et d'autres techniques d'ingénierie sociale pour inciter les victimes à effectuer une demande au serveur dans laquelle est « injecté » du code HTML ou JS

Attaques basées sur le DOM:

Assez similaire aux XSS non persistant mais en utilisant des éléments non affichés - cachés dans le Document Object Model.
(ex : l'ancre #)

Ce type d'attaque se produit généralement côté client et ne passe pas par le serveur.

Attaques XSS stockées (XSS persistant) :

Attaque XSS la plus dévastatrice. Elle se produit quand une page Web va stocker puis afficher un contenu envoyé par un pirate. Les points d'entrée sont généralement des messages sur les forums, des commentaires sur des blogs, du texte dans des listes, des profils utilisateurs et des champs de nom d'utilisateur.

Dev Web / Sécurité : Injection

XSS Injection
(cross site scripting)

Exemple de XSS persistant

The screenshot shows a web browser window with a dark-themed interface. At the top, there's a purple bar with the text "Exemple de XSS persistant". Below it, the browser title bar shows "localhost" and various icons. The main content area displays a "Set List" page with a table of songs. The table has columns for "TITLE", "ARTIST(S)", and "STYLE". A search bar at the top left of the table says "Search ..". The table data is as follows:

TITLE	ARTIST(S)	STYLE
A ma place	A. Bauer & Zazie	Pop Music
A Whiter Shade Of Pale	Procol Harum	Love & Slows
Ai se tu e pego	M. Telo	Tour du Monde - Tube été
Allumer le feu	J. Halliday	Rock - Johnny
Beat it	M. Jackson	Rock around the world
Belles, belles, belles	C. François	Sixties - Twist
Blues suede shoes	E. Presley	Sixties - Rock'n roll

At the bottom of the browser window, the developer tools' "Console" tab is open, showing the message "Console effacée à 13:14:31". To the left of the browser window, there's a small image of a man in a suit pointing his finger, and some handwritten text in French about XSS attacks.

Attaques XSS stockées
Attaque XSS la plus
Web va stocker puis
points d'entrée sont
commentaires sur
utilisateurs et des ch

26/09/2024 WEB 1 : JS 22

Dev Web / Sécurité : Injection

```
[...appendTo:"parent",axis:false,connectWith:false,connectOnBlur:false,handle:false,helper:"original",items:">[...]
```

XSS Injection
(cross site scripting)



Comment s'en protéger ?

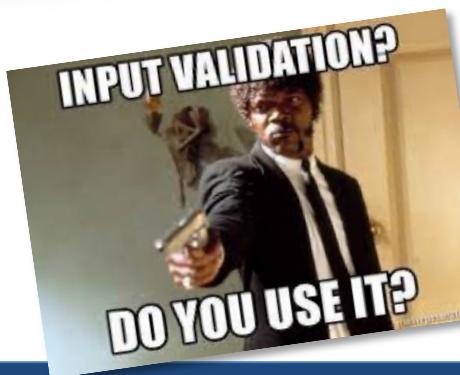


Utiliser un navigateur récent

```
System: Enter password:  
Me: ScoobyDoo  
System: sorry password must  
contain a special character  
Me: ScoobydooFeaturingBatman
```

Empêcher, Supprimer ou Échapper les caractères spéciaux
(surtout < et >)

```
<?php  
htmlspecialchars($_GET["name"]);  
?>
```



Valider les entrées des formulaires :

- Côté Client
- Côté Serveur
- Les deux

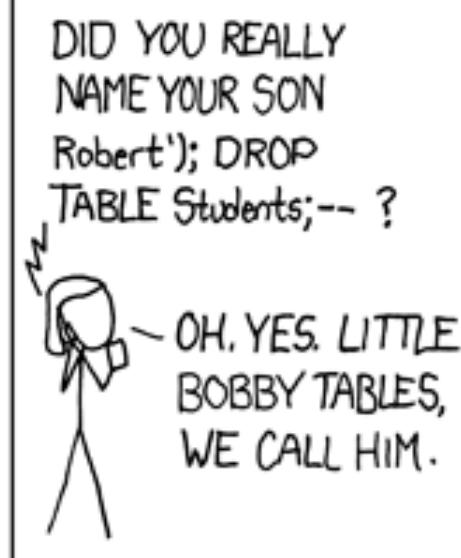
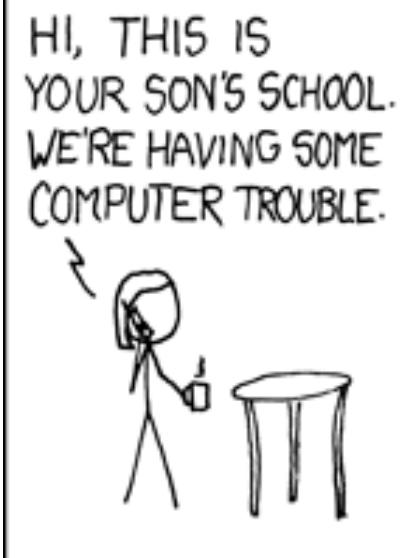
Dev Web / Sécurité : Injection



Def. : Utilisation des entrées utilisateurs pour « injecter » des termes dans vos requêtes SQL et en modifier fondamentalement le comportement



Cette faille existe souvent quand une page récupère les entrées de l'utilisateur et les insèrent telles quelles dans la requête.



Dev Web / Sécurité : Injection



Quels types de SQL que nous pouvons injecter dans la requête?
Pourquoi est-ce mauvais?

```
$query = "SELECT * FROM users WHERE username =  
'$username' AND password = '$password'" ;
```



password : ' OR '1'='1



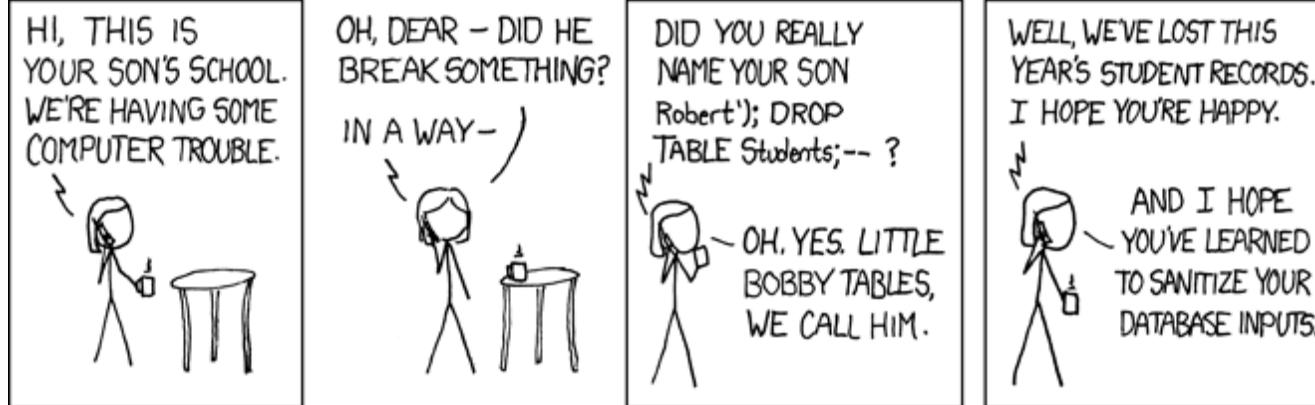
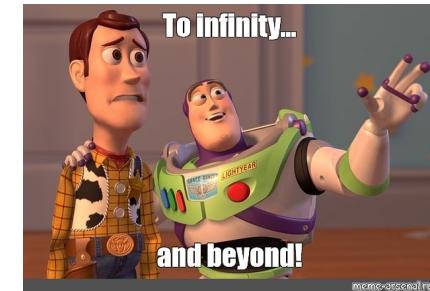
And with delete ??

Dev Web / Sécurité : Injection



SQL Injection

Les possibilités sont infinies



- *Modifier la requête pour renvoyer des données appartenant à d'autres (révélant des informations privées)*
- *Insérer une requête pour modifier des données existantes (augmentation solde d'un compte bancaire...)*
- *Supprimer les données existantes*
- *"Compliquer" la requête pour ralentir le serveur (JOIN a JOIN b JOIN c ...)*

Dev Web / Sécurité : Injection



Comment s'en protéger ?



Empêcher, Supprimer ou Échapper les caractères spéciaux
(notamment les délimiteurs de chaîne : ', ", ...)



Utilisez les requêtes « préparées » avec PDO

```
$stmt = $conn->prepare("INSERT INTO Users (firstname, lastname, email) VALUES (:fn, :ln, :mail)");
$stmt->bind_param(":fn", $firstname); $stmt->bind_param(":ln", $lastname);
$stmt->bind_param(":mail", $mail); $stmt->execute();
```



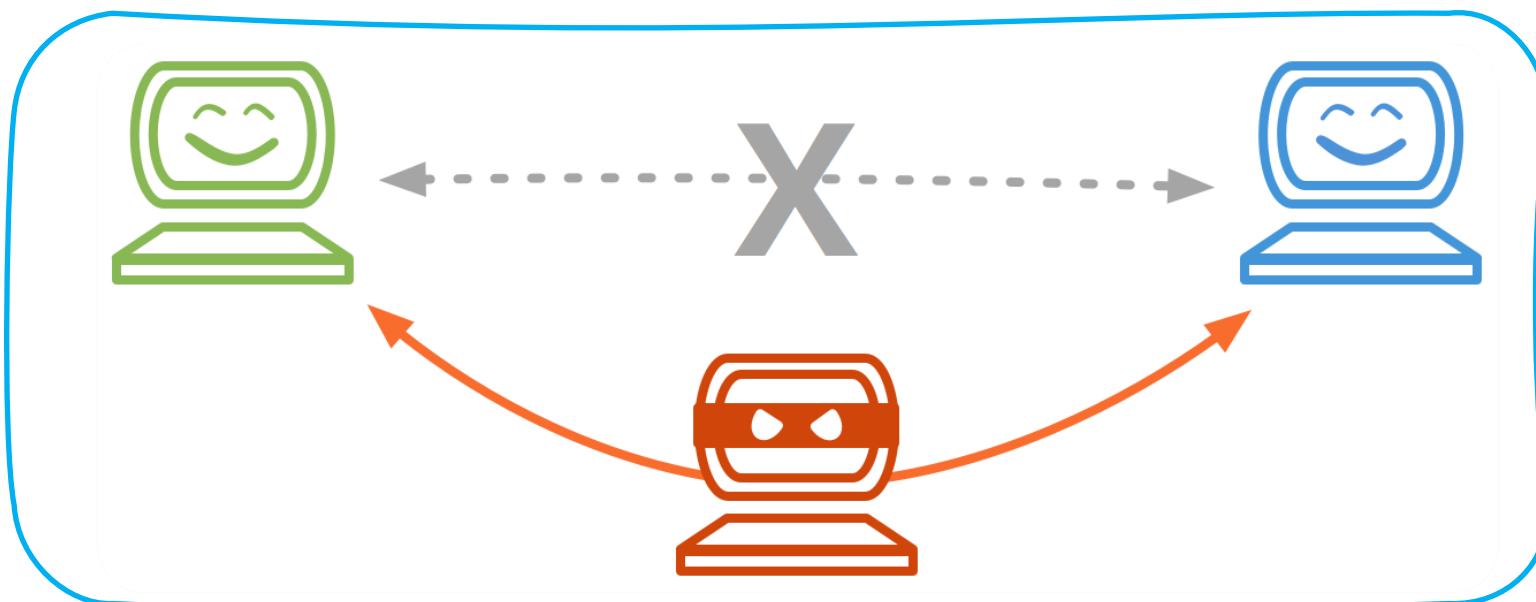
Dev Web / Sécurité : **Man in the Middle** (interception)

Dev Web / Sécurité : Man in the Middle (interception)



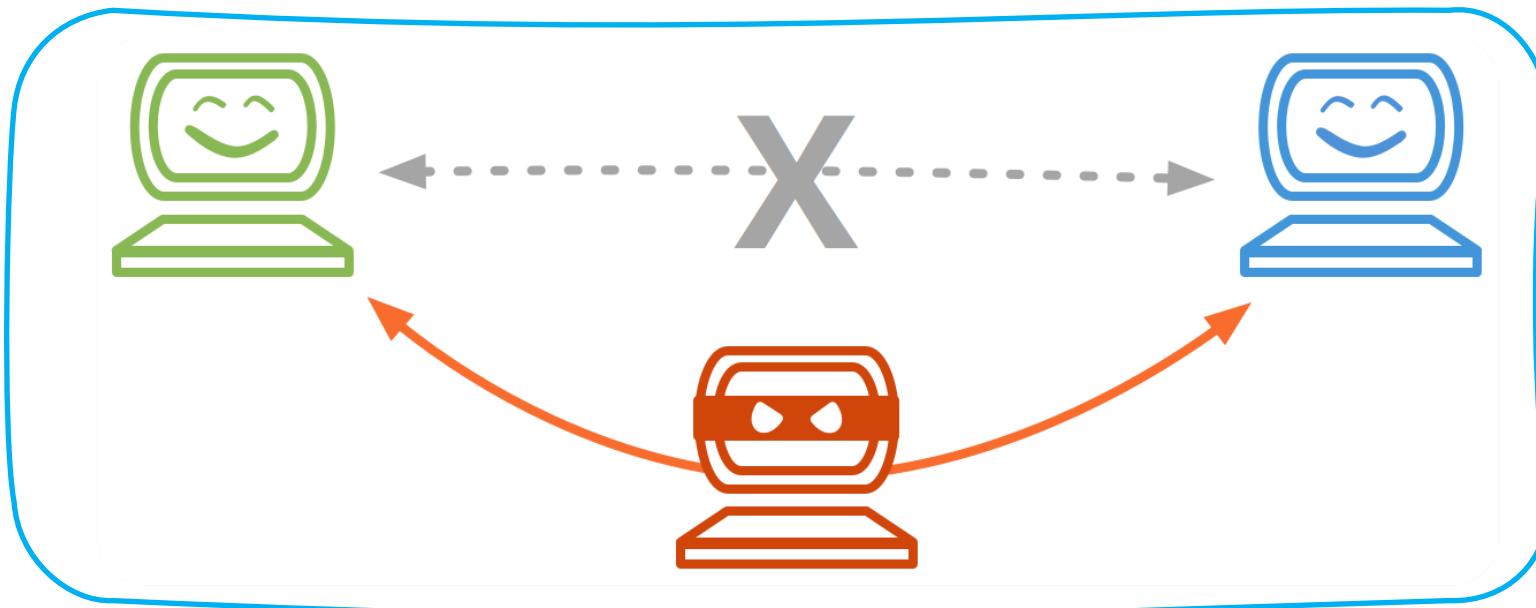
Des utilisateurs mal intentionnés / incompétents

Que l'on cherche à vous avoir
(*Espions, Scripts, CIA ☺*)



Quand un "attaquant" écoute votre réseau et lit ou modifie les données

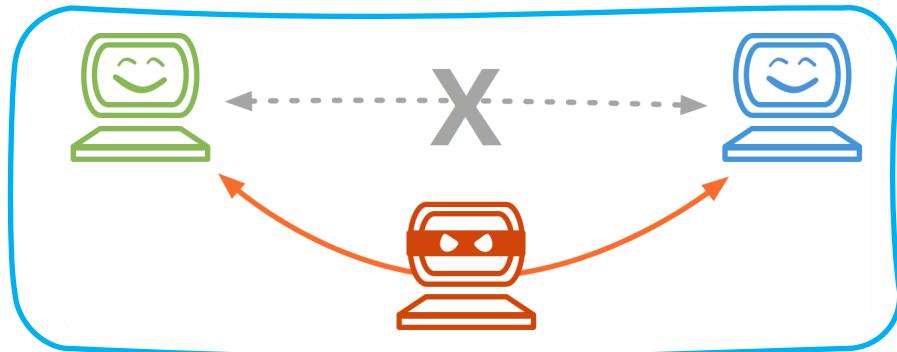
Dev Web / Sécurité : Man in the Middle (interception)



- Fonctionne si l'attaquant peut accéder et compromettre un serveur ou un routeur entre vous et le serveur réel ou si vous êtes sur le même réseau local.
- Souvent, l'attaquant transfère quand même les informations entre vous et le serveur réel, mais il les enregistre ou modifie certaines d'entre elles au passage.
e.g. Il attend que vous envoyiez un couple login/password /, les informations de votre carte de crédit...



Dev Web / Sécurité : Man in the Middle (interception)

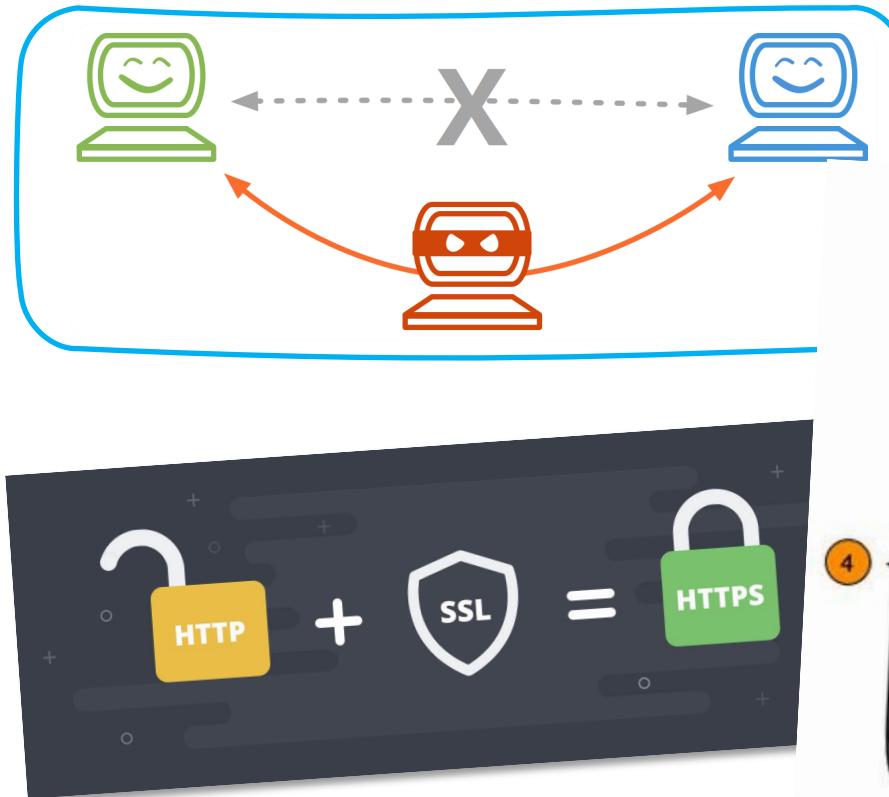


Comment s'en protéger ?

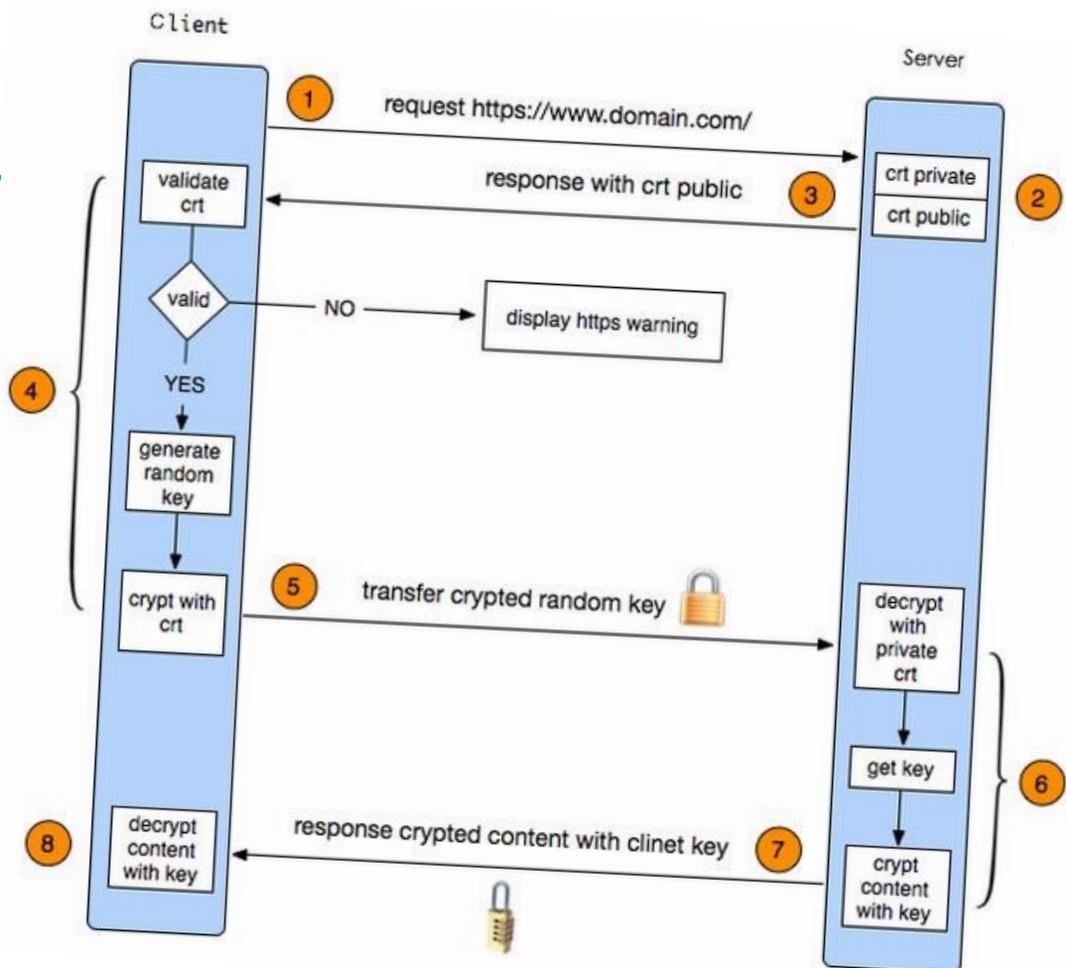


makeameme.or
m9k69mewm9.o

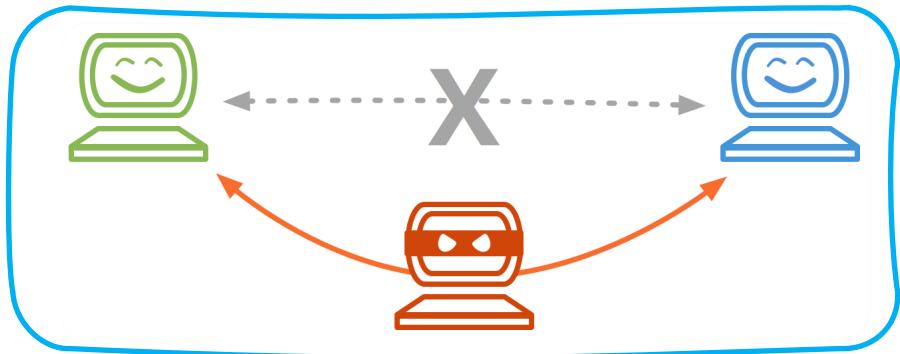
Dev Web / Sécurité : Man in the Middle (interception)



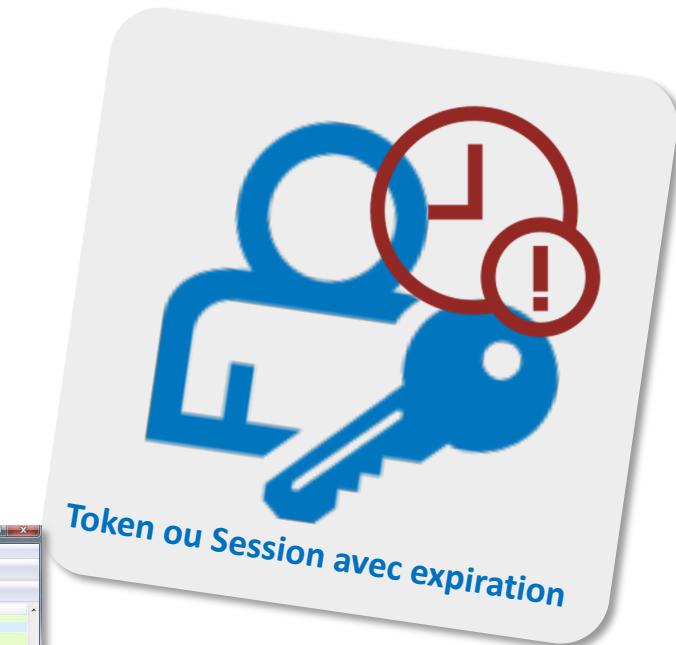
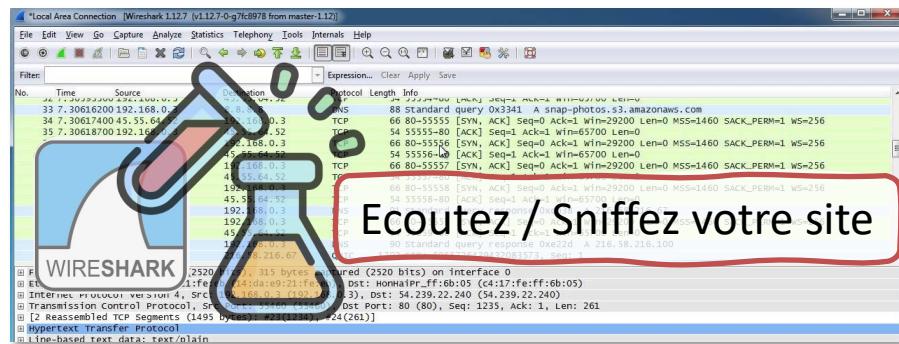
Comment s'en protéger ?



Dev Web / Sécurité : **Man in the Middle (interception)**



Comment s'en protéger ?





Dev Web / Sécurité : **Intrusion serveur & Vol de Base**

Dev Web / Sécurité : Intrusion serveur & Vol de Base



Des utilisateurs mal intentionnés / incompétents

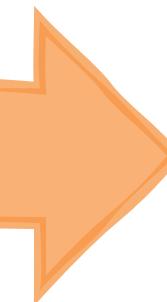
Que l'on cherche à vous avoir
(*Espions, Scripts, CIA ☺*)



Partons du principe que le pire est arrivé :

Votre serveur s'est fait hacké et votre base **Utilisateurs** à été récupérée !

Comment empêcher qu'elle soit exploitée ?
(*Que vos mots de passes soient mis en circulation*)



Dev Web / Sécurité : Intrusion serveur & Vol de Base

1^{er} Problème : Les mots de passes en clair !

1^{ère} Solution : Hash



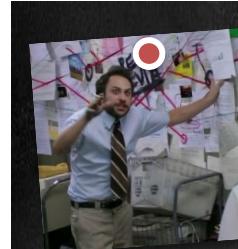
Algorithme de Hachage



Une fonction de hachage prend des données (fichiers, blocs, etc.) et en produit très rapidement une empreinte de taille fixe (assez court, usuellement entre 128 et 512 bits).

Vérifier l'intégrité d'un fichier

Stocker les MDP



3 Propriétés :

- Il est très difficile (très long) de recalculer le contenu d'un message partir de son empreinte ("Résistance à la pré-image").
- Il est très difficile de générer un nouveau message qui donne la même empreinte ("résistance à la seconde préimage").
- Il est très difficile de trouver deux messages qui donnent la même empreinte ("la résistance aux collisions").

Dev Web / Sécurité : Intrusion serveur & Vol de Base

1^{er} Problème : **Les mots de passes en clair !**

1^{ère} Solution :

Hash

Création d'un compte :

Envoi des identifiants
au serveur

Username : **m.yoda**
Password : **mayThe4thWu**

Le Mdp est stocké
haché dans la BDD

Username : **m.yoda**
Password :
9d4a96d76af9a5a22fe1906418da8c8e7b021b5cb
0672830e434aa099e592bb3

Connexion :

Envoi des identifiants
au serveur

Username : **m.yoda**
Password : **mayThe4thWu**

Le Mdp est hashé et comparé au
Mdp hashé dans la BDD

```
$query= "SELECT count(*) FROM users  
WHERE login = "m.yoda" and password =  
"9d4a96d76af9a5a22fe1906418da8c8e7b  
021b5cb0672830e434aa099e592bb3"
```

Dev Web / Sécurité : Intrusion serveur & Vol de Base

1^{er} Problème : Les mots de passes en clair !

1^{ère} Solution :

Hash



Le Hacker n'a pas accès à vos Mdp mais à des Hash quasi impossible à inverser et dont il pourra difficilement extraire le texte original

2^{ème} Problème : Les hackers sont pugnaces 😊



Dev Web / Sécurité : Intrusion serveur & Vol de Base

2^{ème} Problème :

Les hackers sont pugnaces 😊



L'attaque par dictionnaire : Il va hacher pleins de "mots" d'un "dictionnaire" de mots de passe couramment utilisé, des mots de la langue ciblée, des noms propres, etc. Les 500 mots de passe les plus courants sont utilisés par plus de 75% des utilisateurs.

L'attaque par force brute, ou brute-force : il va hacher toutes les combinaisons possibles de lettres/chiffres/symboles : a, b, ..., aaa, aab, aac, ..., aba, abb, ... baa... C'est très très long mais comme la plupart des fonctions de hachage sont très rapides, ça fonctionne rapidement sur les mots de passe courts.

Les rainbow tables, ou tables arc-en-ciel : elles contiennent toutes les combinaisons possibles des mots de passe en fonction des paramètres choisis à la génération. Ils disposent de correspondances directes mot de passe <-> haché. En revanche, ces tables sont extrêmement volumineuses, jusqu'à plusieurs Téraoctets ou bien plus en fonction des paramètres : longueur du mot de passe, chiffres, lettres, symboles, etc.

Dev Web / Sécurité : Intrusion serveur & Vol de Base

2^{ème} Problème : Les hackers sont pugnaces ☺

2^{ème} Solution :

Et on ajoute un peu de sel ...



Création d'un compte :

Envoi des identifiants au serveur

Username : m.yoda
Password : mayThe4thWu

Le MdP est stocké haché dans la BDD

Username : m.yoda
« sel » généré : dac6595c04dda81
Password+sel :
**9d4a96d76af9a5a22fe1906418da8c8e7b021b5cb0
672830e434aa099e592bb3**

Connexion :

Envoi des identifiants au serveur

Username : m.yoda
Password : mayThe4thWu

Le MdP est hashé et comparé au MdP hashé+salé dans la BDD

```
$query1 = "SELECT salt FROM users WHERE login = 'm.yoda'"; ... $salt = ...;  
$hash = SHA256('mayThe4thWu' . $salt);  
$query2 = "SELECT count(*) FROM users WHERE login = 'm.yoda' and password = $hash "
```

Dev Web / Sécurité : Intrusion serveur & Vol de Base





Th-Th-Th-That's All Folks

Dev Web / Sécurité : **FIN**