



INF8085: Cybersécurité

Sécurité des applications web

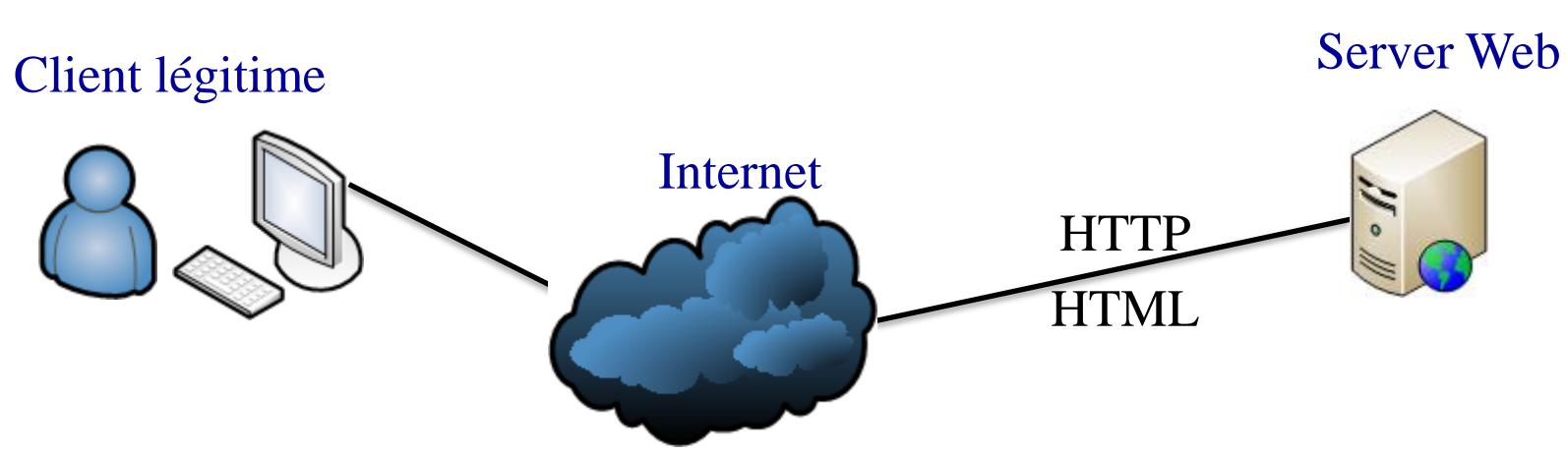


Contenu du cours

- Architecture des applications web
- Authentification
- SQL injection
- Cross site scripting
- Vérification des données usager
- Cross site request forgery
- Phishing (hameçonnage) et moralité de l'histoire



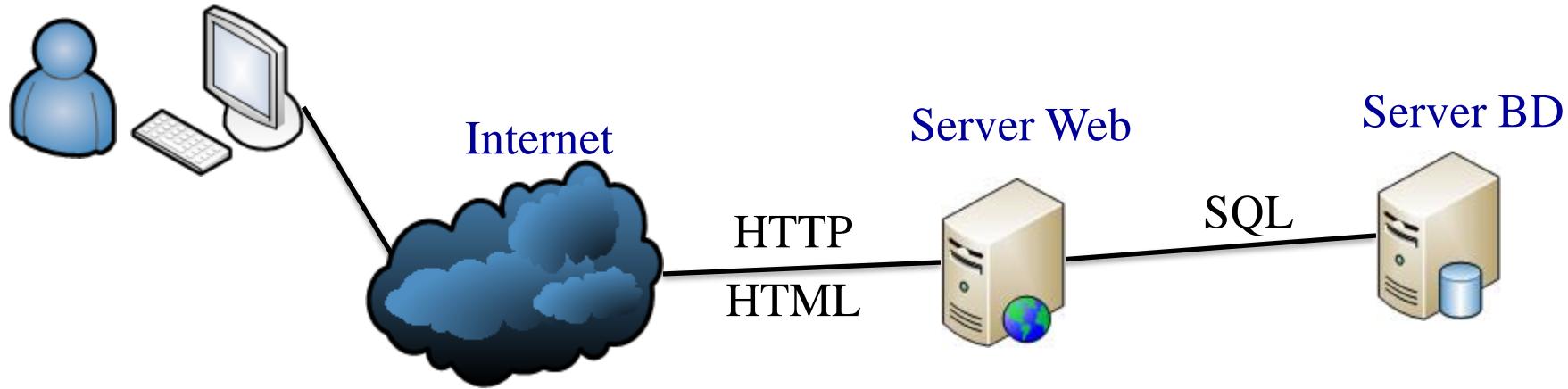
Architecture des applications web





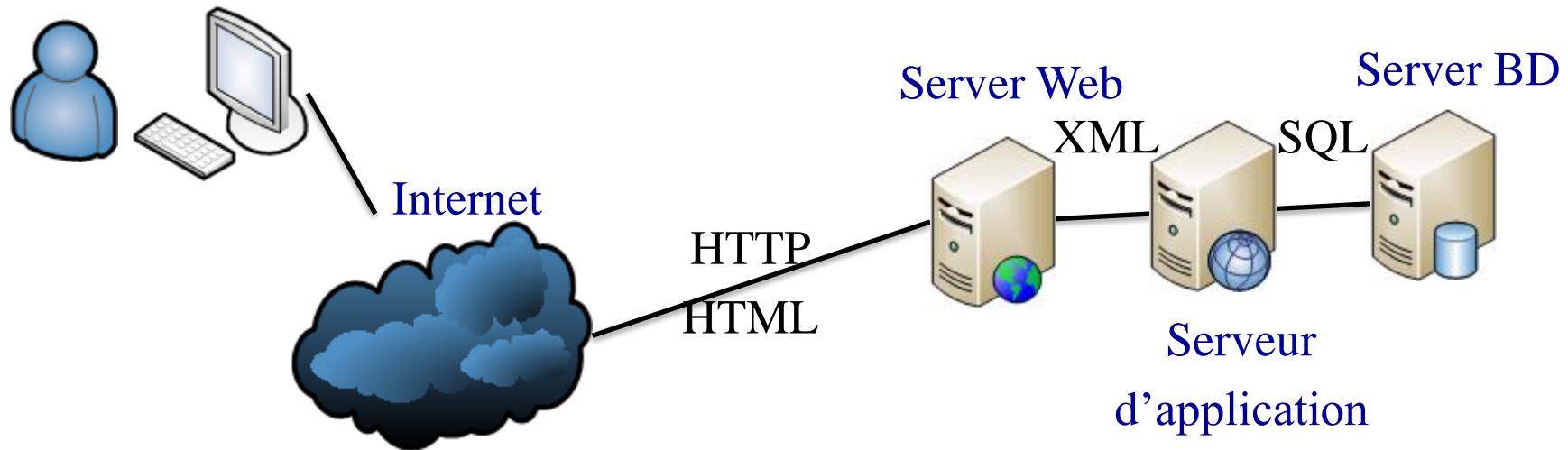
Architecture des applications web

Client légitime



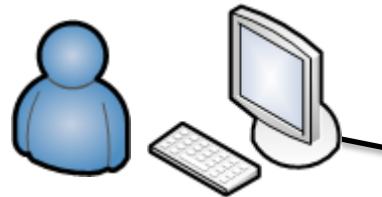
Architecture des applications web

Client légitime



Architecture des applications web

Client légitime



Internet



HTTP
HTML

Server Web



XML



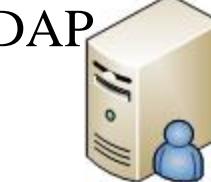
SQL

Server BD



Serveur
d'application

Kerberos

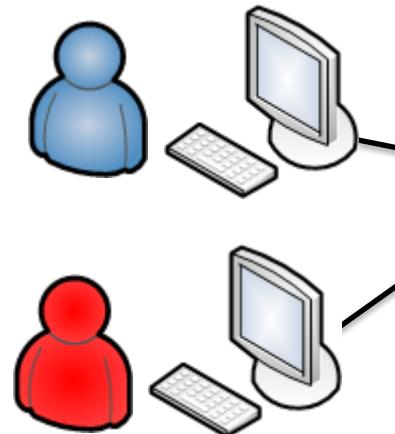


Serveur

d'authentification

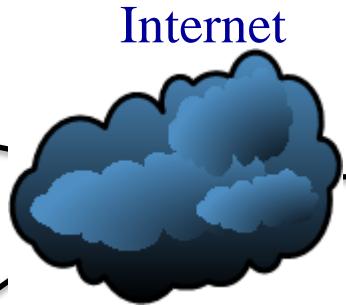
Architecture des applications web

Client légitime



Internet

Client malveillant



HTTP
HTML

Server Web



XML



SQL

Server BD



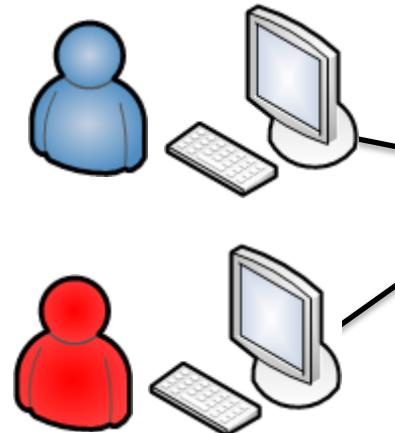
Kerberos
LDAP



Serveur
d'authentification

Architecture des applications web

Client légitime



Internet



HTTP
HTML

Proxy Web
& Pare-feu
Kerberos

Server Web

HTTP
HTML



XML



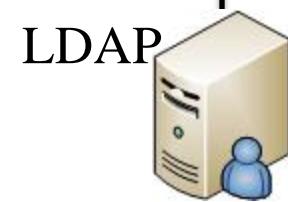
SQL

Server BD



Serveur
d'application

Client malveillant

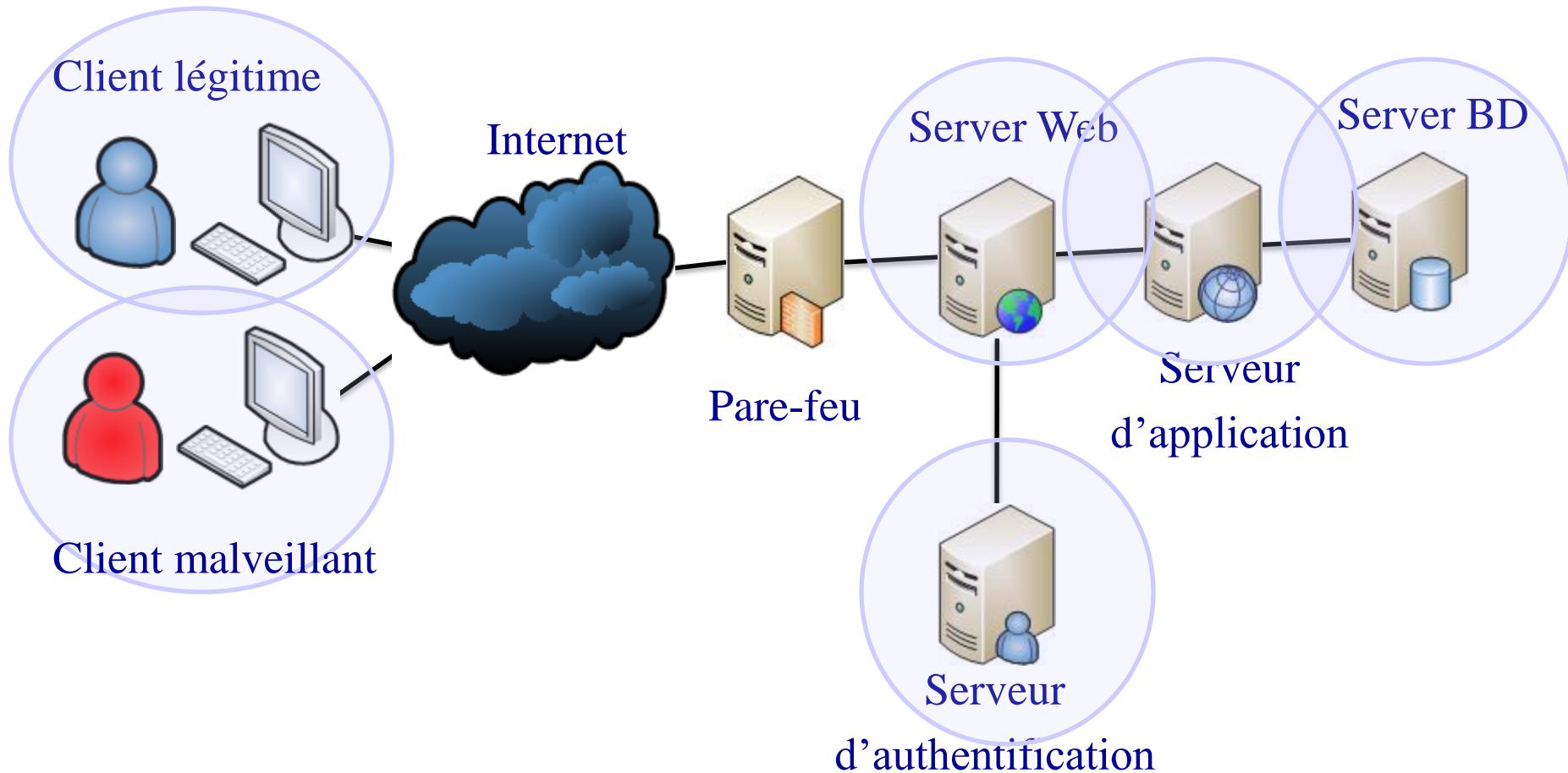


Serveur
d'authentification



Authentification

- Composantes impliquées





Authentification

- Canal de communication sécurisé ([https](https://))
 - Repose sur SSL-TLS (au-dessus de la couche 4)
- Authentifier le serveur
 - Certificat X509
 - Image personnalisée
- Authentifier le client
 - Certificat X509 sur le poste client (optionnel)
 - Nom d'utilisateur + Mot de passe
 - Authentification deux facteurs
- Activation des priviléges
 - En fonction du profil d'authentification
 - Politique d'autorisation (ou de contrôle d'accès)



Authentification

- Authentification du serveur
 - Certificat SSL

The screenshot illustrates two examples of SSL/TLS certificate verification:

Top Window (Google Gmail): A Microsoft Internet Explorer window shows a certificate from VeriSign Class 3 Public Primary CA, identifying the site as www.google.com. The message "This connection to the server is encrypted." is displayed.

Bottom Window (IEEE Xplore): A Firefox window shows a certificate for *.ieee.org issued by DigiCert SHA2 Secure Server CA. The certificate details are as follows:

Émis pour	Détails
Nom commun (CN)	*.ieee.org
Organisation (O)	THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC
Unité d'organisation (OU)	<Ne fait pas partie du certificat>

Bottom Left Window (IEEE Xplore): A Firefox window displaying a document from ieeexplore.ieee.org. It includes a note at the bottom: "IEEE websites place agree to the placement of cookies on your computer."



Authentification

- Authentification du serveur
 - Image personnalisée (par cookie persistant lié au navigateur)

Sign in to Yahoo!

 Are you protected?
Create your sign-in seal.
[\(Why?\)](#)

Yahoo! ID:

Password:

Keep me signed in
for 2 weeks unless I sign out. [New!](#)
[Uncheck if on a shared computer]

[Forget your ID or password?](#) | [Help](#)

Don't have a Yahoo! ID?
Signing up is easy.

[Sign Up](#)

Sign in to Yahoo!



Yahoo! ID:

Password:

Keep me signed in
for 2 weeks unless I sign out. [New!](#)
[Uncheck if on a shared computer]

[Forget your ID or password?](#) | [Help](#)

Don't have a Yahoo! ID?
Signing up is easy.

[Sign Up](#)



Authentification

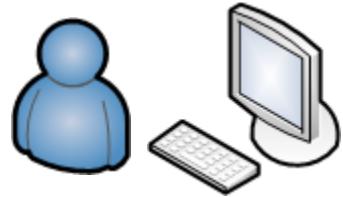
- Comment renforcer l'authentification client-serveur ?
 - Voir cours sur l'authentification
 - Notamment...
- Challenge – response
 - CHAP (Challenge-Handshake Authentication Protocol)
 - Kerberos
- Réauthentification à intervalles réguliers



SQL injection

- Injection SQL (SQL Injection)

Client légitime



Username:

Password:

Remember me on this computer.

Server BD



```
extract($_POST);
```

```
$req = "select mem_code from MEMBRES
       where mem_login = '$login'
       and mem_pwd = '$pass'";
```

```
$result = mysql_query($req) or
die ("Error : the SQL request <br><br>".$req."<br><br> is not valid: ".mysql_error());
list($mem_code) = mysql_fetch_array($result);
if (empty($mem_code))           { //vérifier que la requête a retourné une réponse positive}
```

Server Web

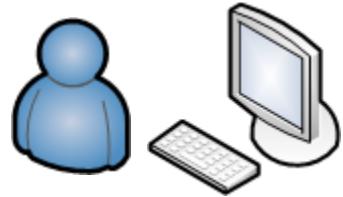




SQL injection

- Injection SQL (SQL Injection)

Client légitime



Username: **daniel**
Password: **Xa4!dfga**
 Remember me on this computer.
Sign in

Server BD



```
select mem_code from MEMBRES  
where mem_login = 'daniel'  
and mem_pwd = 'Xa4!dfga'
```

```
extract($_POST);
```

```
$req = "select mem_code from MEMBRES  
       where mem_login = '$login'  
       and mem_pwd = '$pass'";
```

```
$result = mysql_query($req) or  
die ("Error : the SQL request <br><br>".$req."<br><br> is not valid: ".mysql_error());  
list($mem_code) = mysql_fetch_array($result);  
if (empty($mem_code)) { // vérifier que la requête a retourné une réponse positive}
```

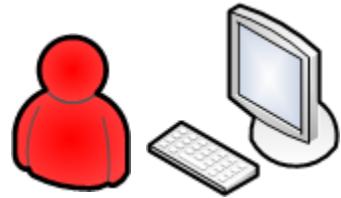
Server Web





SQL injection

- Injection SQL (SQL Injection)



Client malveillant

Username: **daniel**
Password: **' or '1'='1**
 Remember me on this computer.
Sign in

```
extract($_POST);

$req = "select mem_code from MEMBRES
        where mem_login = '$login'
        and mem_pwd = '$pass'";

$result = mysql_query($req) or
die ("Error : the SQL request <br><br>".$req."<br><br> is not valid: ".mysql_error());
list($mem_code) = mysql_fetch_array($result);
if (empty($mem_code))          { // vérifier que la requête a retourné une réponse positive}
```

Server BD



```
select mem_code from MEMBRES
where mem_login = 'daniel'
and mem_pwd = '' or '1'='1'
```

Server Web





SQL injection

- Injection SQL (SQL Injection)



```
x'; INSERT INTO members
('email','passwd','login_id','full_name') VALUES
('steve@unixwiz.net','hello','steve','Steve Friedl');--
```



SQL injection

- Injection SQL (SQL Injection)



`x' ; exec(char(0x73687574646f776e)) ' ;`



`x' ; shutdown ;`

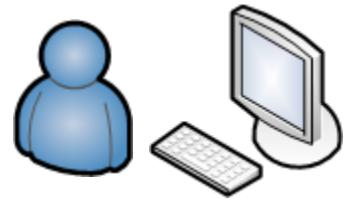
Remarque importante :

- La plupart des SGBD permettent d'exécuter des commandes shell depuis le SGBD
- Exemple Microsoft SQL Server)
- On peut en faire autant avec une SQL injection qu'avec un shell code !

Cross site scripting

- Cross site scripting (XSS) non persistant

Client légitime



Server Web



```
extract($_POST);

$req = "select * from POSTS
        where title = '$stitle'
```

Search results for Gagner de l'argent:

- Comment gagner de l'argent facile et des cadeaux sur internet...
- L' objectif du blog est de présenter toutes les idées qui permettent d' économiser ...

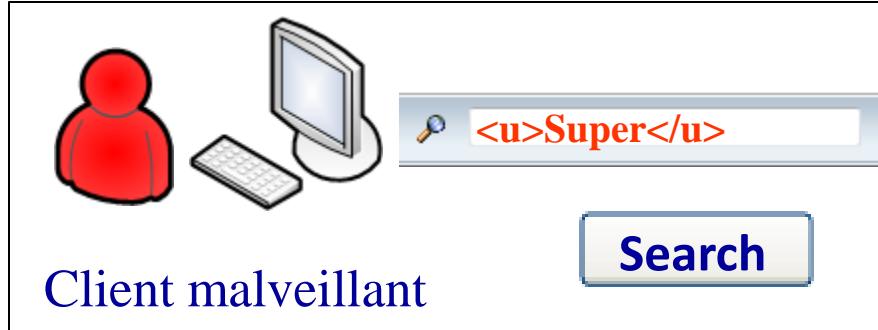
```
<html>
<head></head>
<body>

<h1>Search results for Gagner de l'argent :</h1>
<itemize>
    <item>Comment gagner de l'argent facile et
des cadeaux sur internet...</item>
    <item>L'objectif du blog est de présenter
toutes les idées qui permettent d'économiser ...</item>
</itemize>
</body>
</html>
```



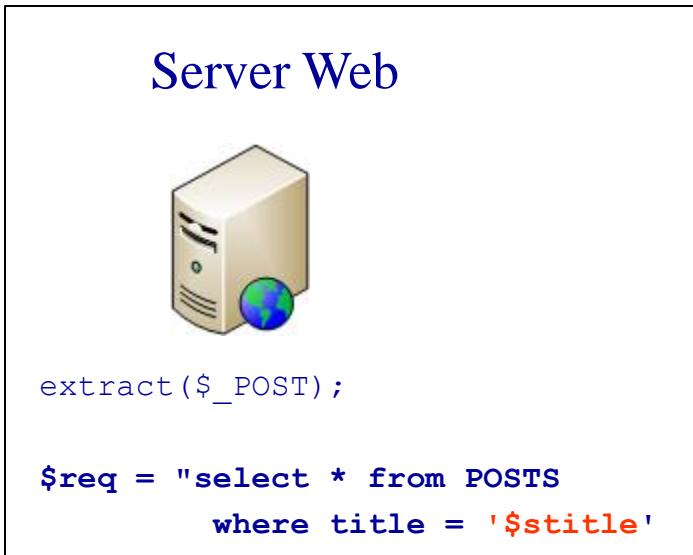
Cross site scripting

- Cross site scripting (XSS) non persistent



Search results for Super:

No results found



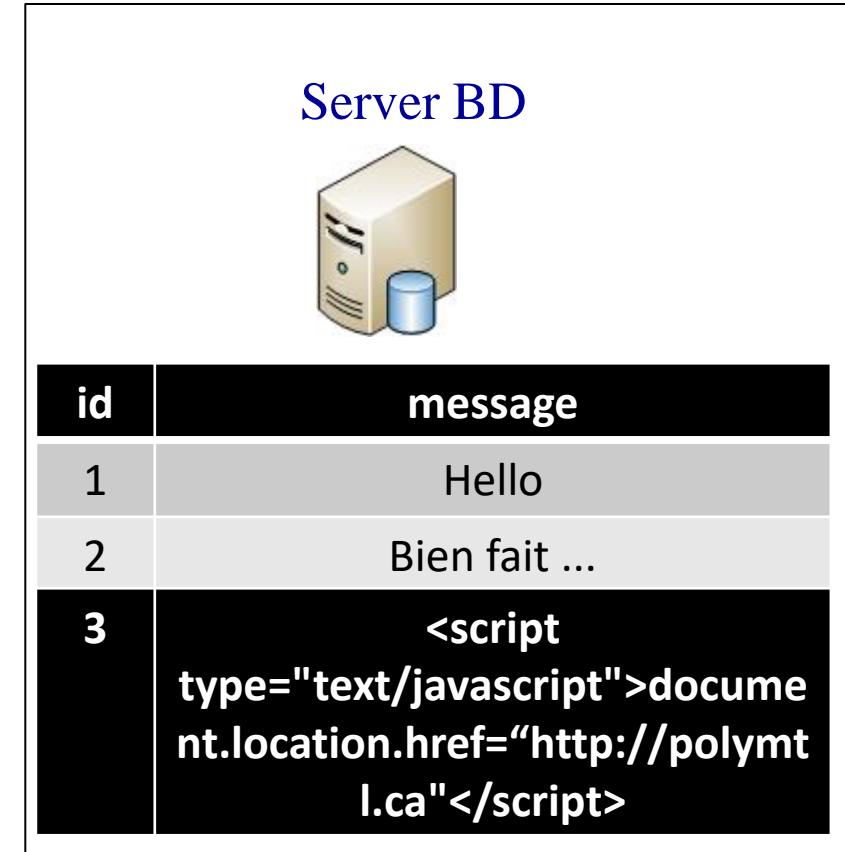
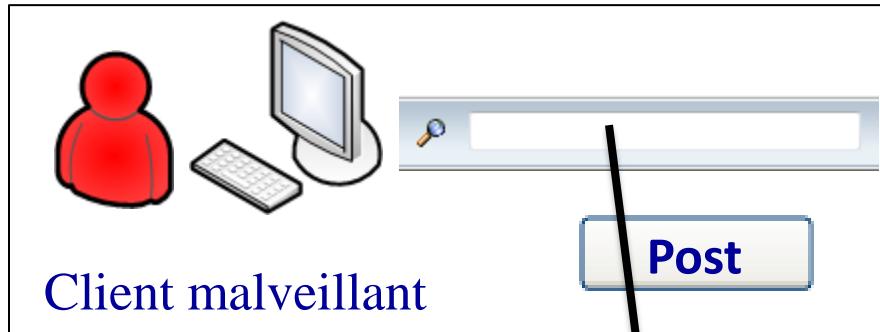
```
<html>
<head></head>
<body>

<h1>Search results for <u>Super</u> :</h1>
No results found
</itemize>
</body>
</html>
```



Cross site scripting

- Cross site scripting (XSS) persistent



```
<script type="text/javascript">document.location.href="http://polymtl.ca"</script>
```

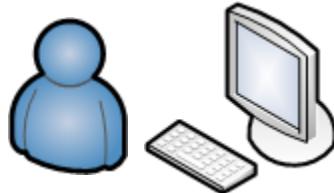
Your message has been posted



Cross site scripting

- Cross site scripting (XSS)

Client légitime



Guestbook messages:

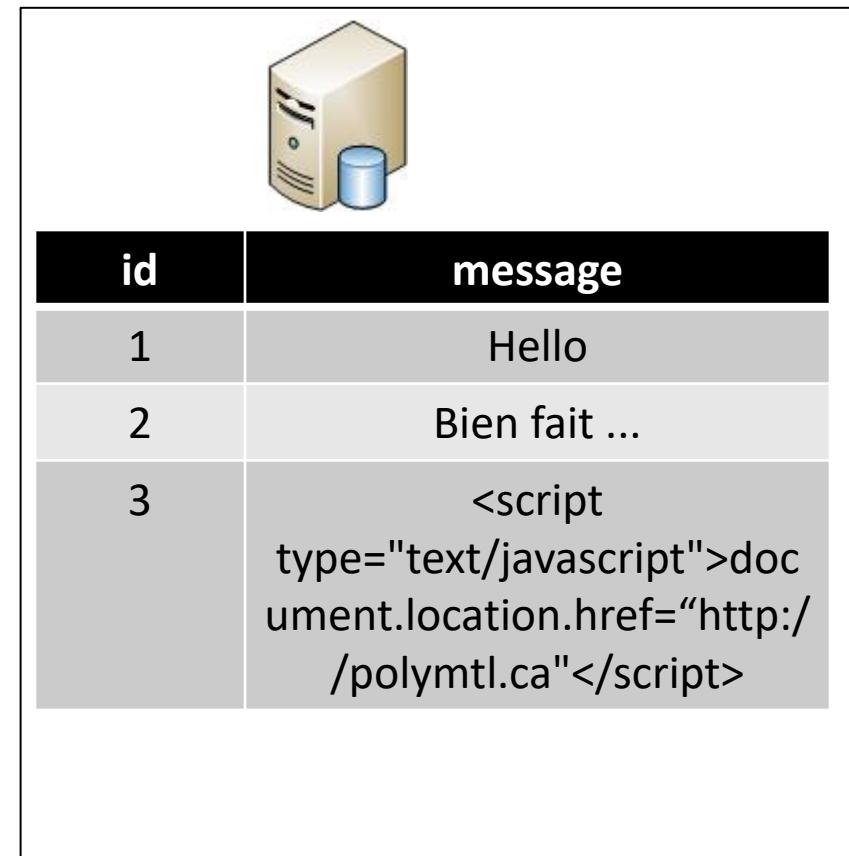
Hello

Bien fait ...



```
<h1>Guestbook messages:</h1>
Hello<br>
Bien fait<br>
<script
type="text/javascript">document.locatio
n.href="http://polymtl.ca"</script><br>
...
...
```

Server BD

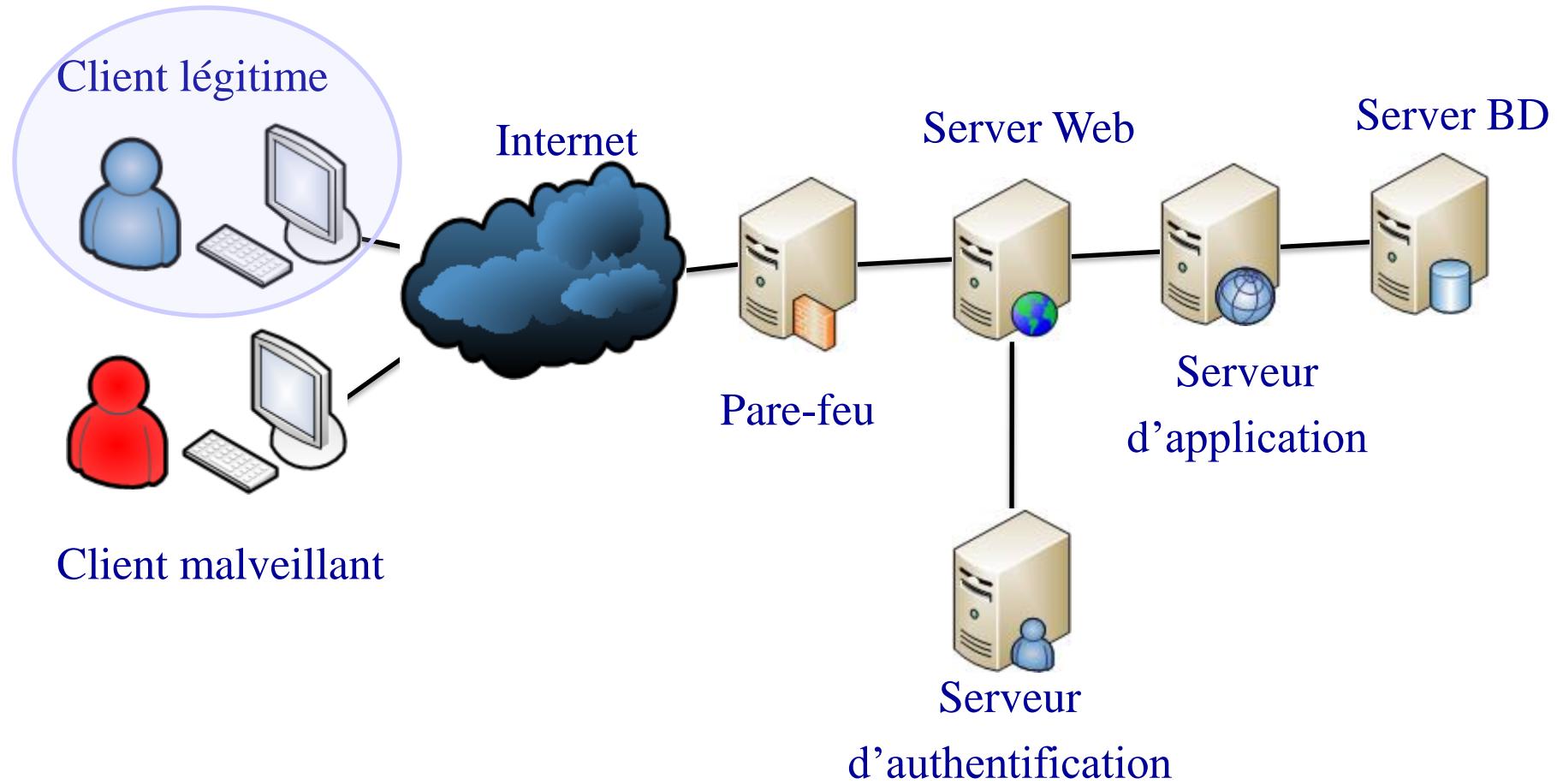




- Quel est le but de l'attaquant dans une attaque XSS ?
 - Rediriger le trafic du client vers le site de l'attaquant
- Pourquoi ?
 - Améliorer le référencement du site de l'attaquant
 - Faire de l'argent
 - L'attaquant se fait de l'argent en faisant cliquer le client sur son site
 - Infecter le site client
 - Exploitation d'une vulnérabilité du navigateur du client
 - Souvent, attaque par buffer overflow
- Remarque
 - Le XSS n'a pas de réel impact sur le serveur attaqué
 - Le serveur sert seulement à relayer le client vers le site de l'attaquant

Vérification des données usager (Input validation)

- Ce qu'on fait





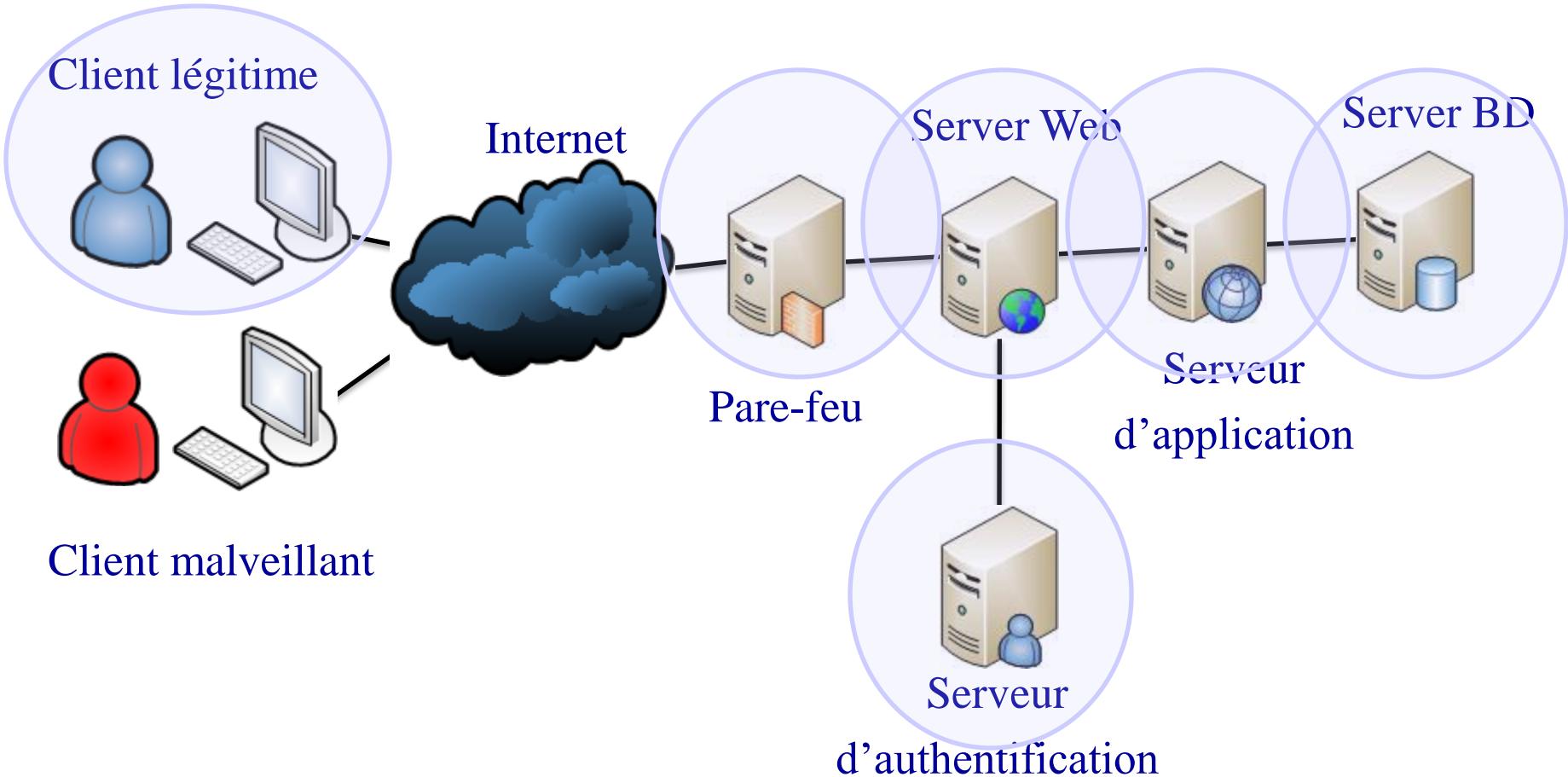
Vérification des données usager (Input validation)

- Code source html

```
<form action="mailto:yourname@yourdomain.com" method="post" onsubmit="return  
checkform(this);">  
  
<script language="JavaScript" type="text/javascript">  
<!--  
function checkform ( form )  
{  
    // see http://www.thesitewizard.com/archive/validation.shtml  
    // for an explanation of this script and how to use it on your  
    // own website  
  
    // ** START **  
    if (form.email.value == "") {  
        alert( "Please enter your email address." );  
        form.email.focus();  
        return false ;  
    }  
    // ** END **  
    return true ;  
}  
//-->  
</script>
```

Vérification des données usager (Input validation)

- Ce qu'on devrait faire



Vérification des données usager (Input validation)

- Valider les données de l'usager
- Où ?
 - sur le serveur Web
 - et/ou sur le serveur d'applications
- Comment ?
 - Exact Match (exemple : seulement « true » et « false » permis)
 - Whitelisting (exemple : seulement (a-zA-Z)+ permis)
 - Blacklisting (exemple: « SELECT » « JOINT » pas permis)
 - Encoding (exemple : mysqli_real_escape_string)
- Quoi d'autre ?
 - Limiter la taille de l'entrée



Vérification des données usager (Input validation)

- Utiliser les SQL Stored Procedures
- Gérer les permissions sur la base de données
 - usagers, rôles, permissions
- Messages d'erreur
- Pare-feu applicatif
 - Software
 - ModSecurity
 - Appliance
 - Cisco, Fortinet, Checkpoint, etc.

Vérification des données usager (Input validation)

- Comment vérifier si un site est vulnérable ?
- Rien fait pour se protéger -> probablement vulnérable
- Développé sans gestion de projet -> probablement vulnérable
- Outils de scan automatique
 - Nikto
 - Acunetix (\$\$\$\$ mais gratuit pour test de XSS)
 - WebScarab
 - Autres (<http://sectools.org/web-scanners.html>)



Vérification des données usager (Input validation)

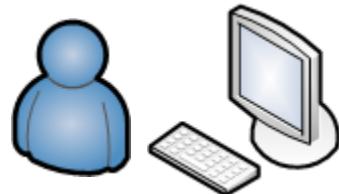
- Autres types d'attaques contre les applications Web
 - Cross Site Request Forgery (XSRT)
 - Remote File Inclusion
 - Variable tampering
 - Interface redressing (clickjacking)



Cross-Site Request Forgery

- Utilisation normale

Client légitime



GET index.php

Server Web



www.exemple.com

Username:
Password:
 Remember me on this computer.

POST login.php

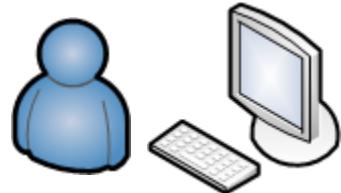
Redirect index.php
Set Cookie: PHP_SESSID=vwae9pa6nw408967a123...



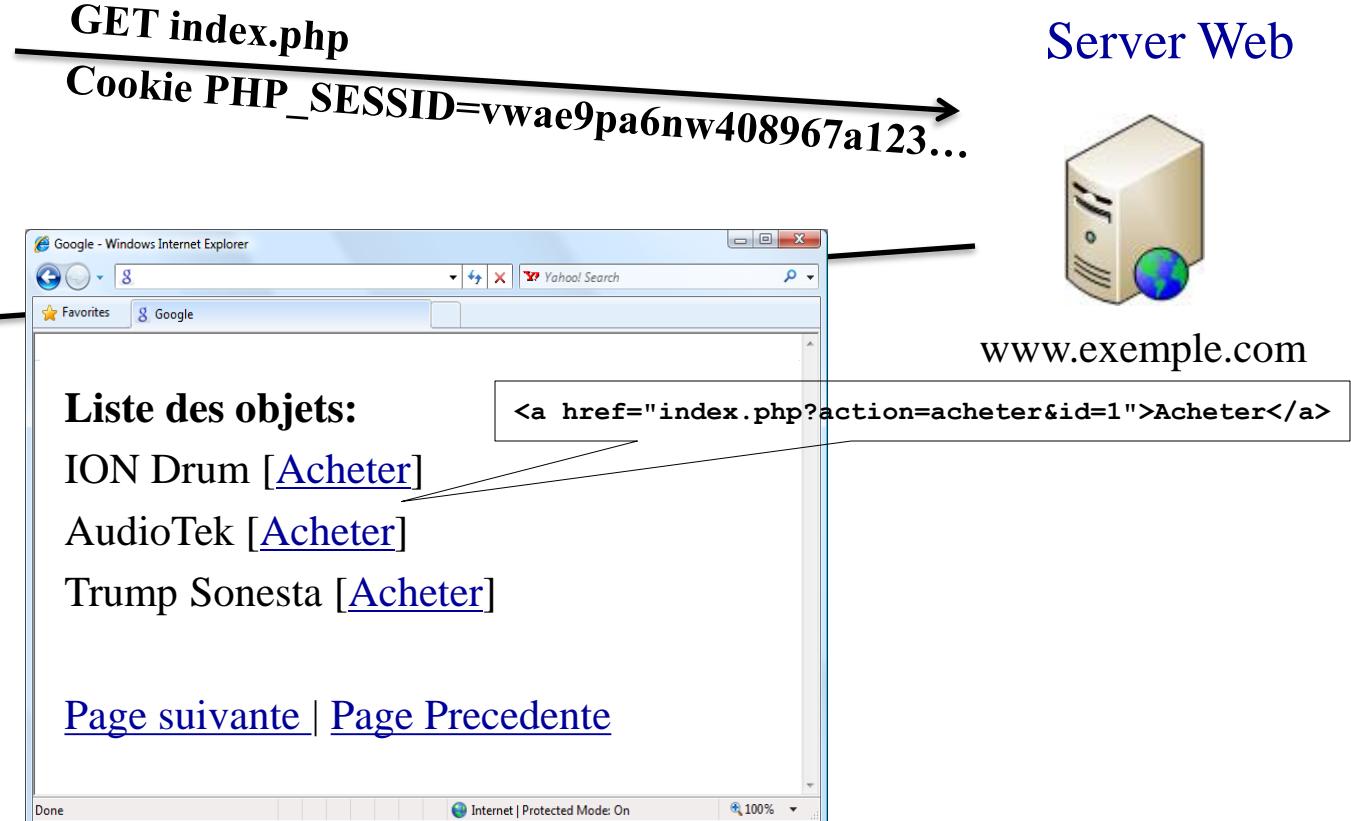
Cross-Site Request Forgery

- Utilisation normale (2/2)

Client légitime



Server Web

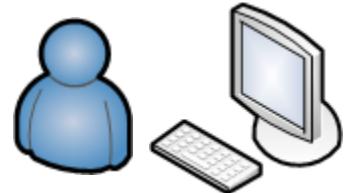


GET =index.php?action=acheter&id=1
Cookie PHP_SESSID=vwae9pa6nw408967a123...

Cross-Site Request Forgery

- Attaque

Client légitime



GET malicious.asp

Server Web



www.attaque.com

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

</body>
</html>
```

GET =index.php?action=acheter&id=1
Cookie PHP_SESSID=vwaec9pa6nw408967a123...

Server Web



www.exemple.com



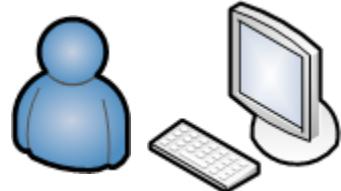
Protection contre le Cross-Site Request Forgery

- Comment se protéger ?
- Token aléatoire
 - Envoyé au moment du login
 - Stocké dans les variables de session du coté serveur
 - Ne pas stocké dans un cookie du coté client, mais
 - Présent dans les liens de toutes les autres pages
 - Vérifié par le serveur pour chaque page

Protection contre le Cross-Site Request Forgery

- Utilisation normale

Client légitime

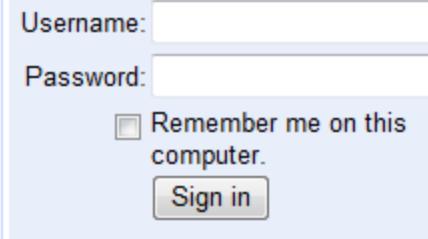


GET index.php

Server Web



www.exemple.com



Username:

Password:

Remember me on this computer.

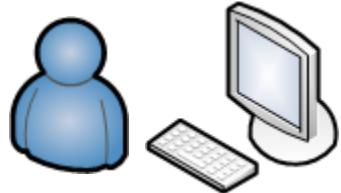
POST login.php

Redirect index.php?secureToken=431fwap8rawddf...
Set Cookie: PHP_SESSID=vxae9pa6nw408967a123...

Protection contre le Cross-Site Request Forgery

- Utilisation normale

Client légitime



Server Web

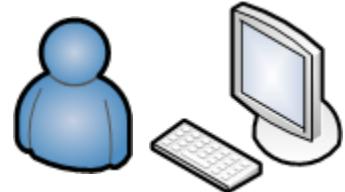


GET =index.php?action=acheter&id=1&secureToken=431fwap8rawddf...
Cookie PHP_SESSID=vxae9pa6nw408967a123...

Protection contre le Cross-Site Request Forgery

- Attaque

Client légitime



GET malicious.asp

Server Web



www.attaque.com

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

</body>
</html>
```

GET =index.php?action=acheter&id=1
Cookie PHP_SESSID=vwaе9pa6nw408967a123...

Server \

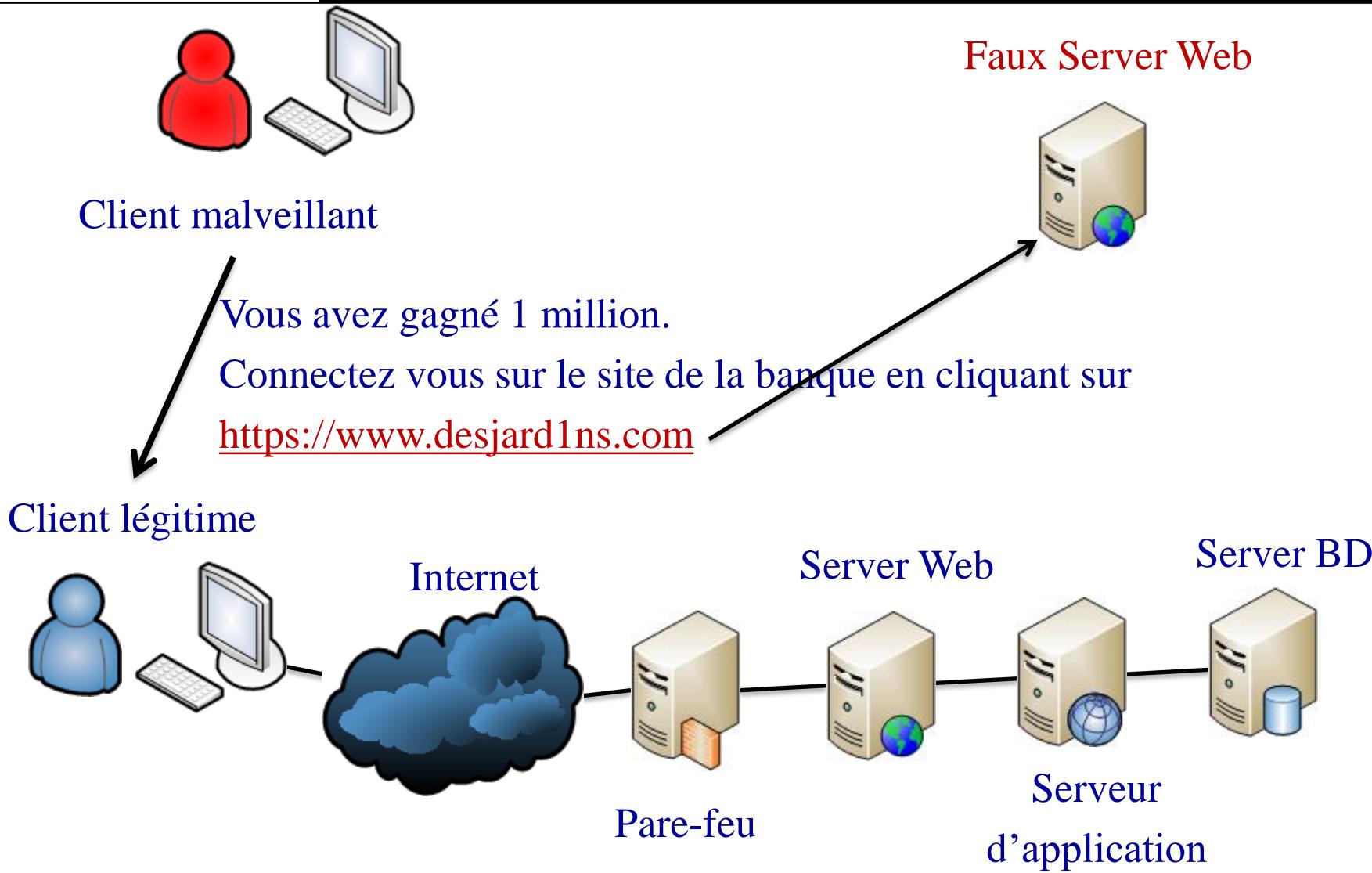


www.exemple.com

Vérification de la variable
secureToken échouée.
Session fermée.



Hameçonnage (Phishing)





Hameçonnage (Phishing)

- Comment se protéger ?
- Filtrer le spam
- Authentification du serveur
- Eduquer les utilisateurs

Moralité de l'histoire

- Chaque attaque est différente
- Exploite la logique de l'application
- Difficile (impossible) à détecter par des outils automatiques
- Code review
- Exemples
 - Faire un don de -100\$
 - Créer un million d'usagers et écrire des messages dans un forum
 - Enlever le câble réseau au milieu d'une partie d'échec

Moralité de l'histoire

- Attaques web très populaires
- Facile de créer une application vulnérable
- Validation des données usager
- Éducation des usagers
- Principe de sécurité de l'oignon (layered security)
- OWASP (Open Web Application Security Project) Top 10
 - http://www.owasp.org/index.php/Top_10_2007
- Clickjacking
 - <http://ha.ckers.org/blog/20081007/clickjacking-details/>
- SQL Injection Cheat Sheet
 - <http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/>
- XSS Cheat Sheet
 - <http://ha.ckers.org/xss.html>