

1. Les attaques MITM
2. Le phishing
3. Les attaques CSRF
4. Les modifications du DOM
5. Les injections
6. Le brute force

LES INJECTIONS

Une injection, c'est quoi ?



Faire une injection, c'est faire rentrer du code à l'intérieur d'une application (y compris un site internet).

Faire une injection, c'est faire rentrer du code à l'intérieur d'une application (y compris un site internet).

Comment faire entrer du code à l'intérieur d'une application ?



A LA PLACE D'UN HACKER ...

On utilise tout ce qu'on peut !

- Tous les **inputs** présents sur le site internet
- On essaie même de mettre du code dans les **URL**

laurence@xonatis.com

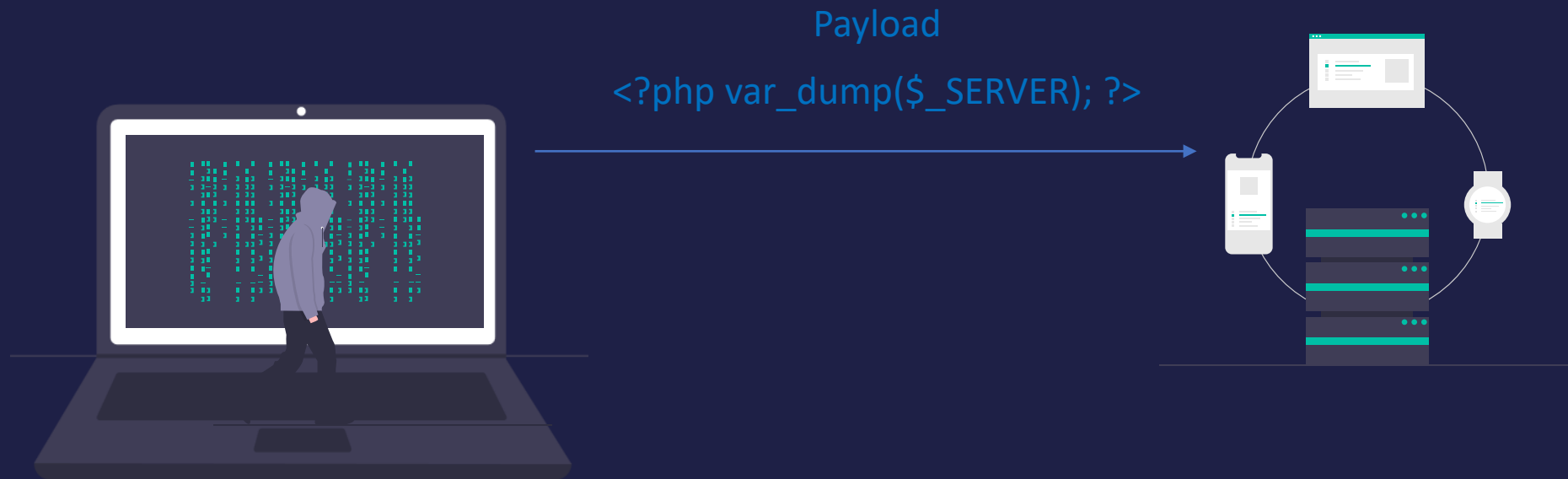
Soumettre

A la place, le hacker va
essayer :

<?php var_dump(\$_SERVER)

Soumettre





Que va-t-il se passer au niveau du
serveur PHP ?



Ceci s'appelle un injection PHP car le hacker essaie d'injecter du code PHP dans le site internet

Presqu'aucun site (sauf une version de Wordpress) est vulnérable aux injections PHP

Quel autre langage peut-on
injecter ?



Une injection de langage SQL s'appelle une
injection SQL

laurence@xonatis.com

Soumettre

A la place, le hacker va
essayer :

' OR 1 = 1; --

Soumettre

Pourquoi ?



Une **action de login** dans un controller a souvent été codée comme ceci dans les sites internet de 2011 (**sites qui existent encore aujourd'hui**)

```
function check()
{
    // On récupère le username et le password du payload
    $username = $_POST['username'];
    $password = $_POST['password'];

    // On crée une connexion vers la base de données (comme mysqli)
    $connexion = new PDO('mysql:host=localhost;dbname=patissor', 'root', '');

    // On exécute la requête SQL pour chercher l'utilisateur avec ce $username et $password
    $requete = "SELECT COUNT(id) AS nbr FROM user WHERE username = '$username' AND password = '$password'";
    $resultat = $connexion->query($requete);

    // On récupère le résultat de la requête SQL dans $row
    $row = $resultat->fetch();

    // On vérifie si nbr > 0 (donc si un utilisateur a été trouvé en base)
    if ($row['nbr'] > 0)
    {
        // On redirige vers l'espace client
        header('LOCATION: /espace-client');
        die();
    }
}
```

Que se passe-t-il si le \$username contient du code SQL ?

```
function check()
{
    // On récupère le username et le password du payload
    $username = $_POST['username'];
    $password = $_POST['password'];

    // On crée une connexion vers la base de données (comme mysqli)
    $connexion = new PDO('mysql:host=localhost;dbname=patissor', 'root', '');

    // On exécute la requête SQL pour chercher l'utilisateur avec ce $username et $password
    $requete = "SELECT COUNT(id) AS nbr FROM user WHERE username = '$username' AND password = '$password'";
    $resultat = $connexion->query($requete);

    // On récupère le résultat de la requête SQL dans $row
    $row = $resultat->fetch();

    // On vérifie si nbr > 0 (donc si un utilisateur a été trouvé en base)
    if ($row['nbr'] > 0)
    {
        // On redirige vers l'espace client
        header('LOCATION: /espace-client');
        die();
    }
}
```

```
SELECT COUNT(id) AS nbr FROM user  
WHERE username = '$username' AND password = '$password'
```

```
SELECT COUNT(id) AS nbr FROM user  
WHERE username = '$username' AND password = '$password'
```

Si le hacker rentre ' OR 1 = 1; -- dans le payload pour \$username et aaaa pour le \$password

```
SELECT COUNT(id) AS nbr FROM user  
WHERE username = '$username' AND password = '$password'
```

Si le hacker rentre ' OR 1 = 1; -- dans le payload pour \$username
et aaaa pour le \$password



```
SELECT COUNT(id) AS nbr FROM user  
WHERE username = ' ' OR 1 = 1; -- ' AND password = 'aaaa'
```

```
SELECT COUNT(id) AS nbr FROM user  
WHERE username = '$username' AND password = '$password'
```

Si le hacker rentre ' OR 1 = 1; -- dans le payload pour \$username
et aaaa pour le \$password



```
SELECT COUNT(id) AS nbr FROM user  
WHERE username = '' OR 1 = 1; -- ' AND password = 'aaaa'
```



```
SELECT COUNT(id) AS nbr FROM user  
WHERE username = '' OR 1 = 1; -- ' AND password = 'aaaa'
```

Le hacker a détourné la requete SQL, nous sommes passés de :

```
SELECT COUNT(id) AS nbr FROM user  
WHERE username = '$username' AND password = '$password'
```

A la requete suivante :

```
SELECT COUNT(id) AS nbr FROM user  
WHERE username = '' OR 1 = 1; -- ' AND password = 'aaaa'
```

Le mot de passe n'est même plus vérifié !
Et le username ne sert plus à rien !

Il est donc possible de s'identifier sans
préciser le bon username et le bon password



Payload
' OR 1 = 1; --





A LA PLACE D'UN HACKER ...

Objectifs :

1. Exécuter du code dans la base de données du site internet

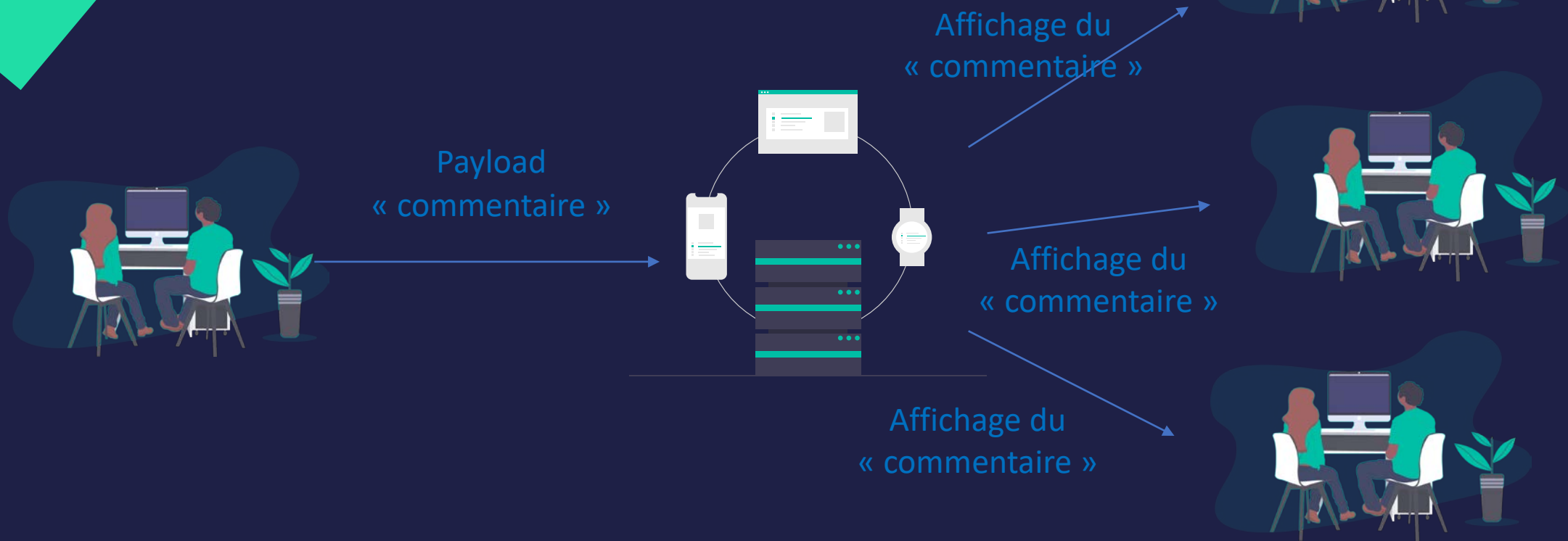
Mise en place :

1. Deviner comment est la forme de la requete SQL au backend
2. Utiliser un input pour écrire un code SQL qui viendra s'insérer dans la forme de la requete SQL pour la détourner

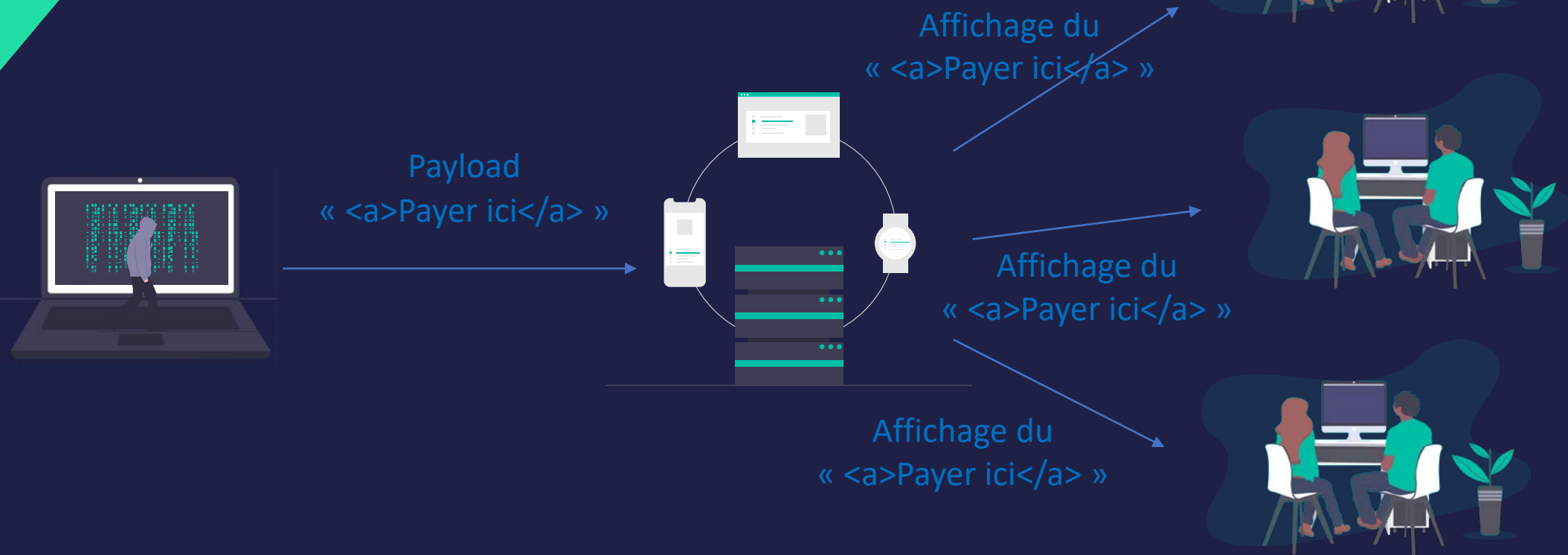
Une injection de langage HTML s'appelle une injection XSS (Cross site scripting)

Les injections XSS fonctionnent sur des pages qui **affichent des données** communautaires (comme par exemple **une page de commentaires**)

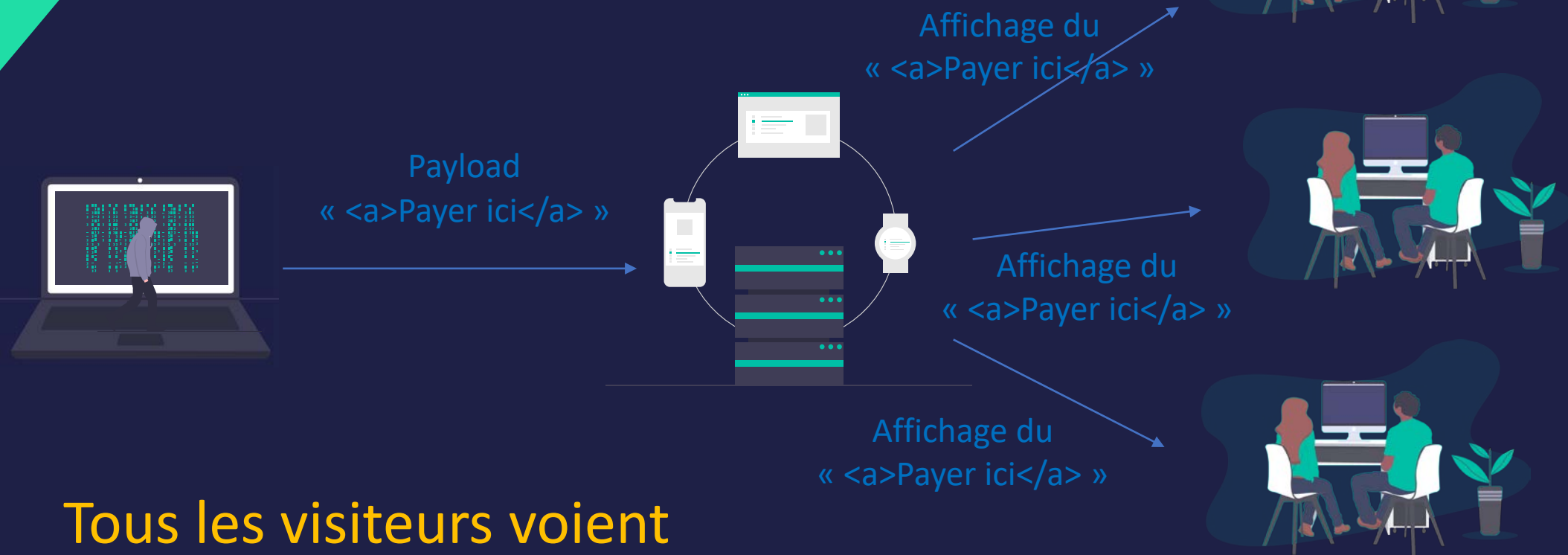
Fonctionnement normal



Fonctionnement hacké



Fonctionnement hacké



Tous les visiteurs voient
le faux lien pour payer !



A LA PLACE D'UN HACKER ...

Objectifs :

1. Exécuter du code dans le navigateur des visiteurs

Mise en place :

1. Chercher une page qui affiche aux visiteurs ce que l'on écrit (peut être une page de commentaires ou autre)
2. Utiliser un input pour écrire un code HTML à la place du commentaire pour qu'il s'affiche chez les visiteurs

