

1. Les attaques MITM : HTTPS
2. Le phishing : Eduquer les visiteurs
3. Les attaques CSRF : Token CSRF
4. Les modifications du DOM : Faire les vérifications au backend
5. Les injections : Figer la forme de la requête SQL et désactiver les balises HTML
6. Le brute force : Mettre une limite sur le nombre d'essais



Qui peut rappeler le principe  
d'une attaque CSRF ?



# RAPPEL ATTAQUES CSRF



A LA PLACE D'UN HACKER ...

## Objectifs :

1. Forcer le visiteur à faire une action à son insu

## Mise en place :

1. Le hacker va regarder soit l'inspecteur soit le code source de la page dans son navigateur pour identifier la configuration du formulaire sur le vrai site à répliquer (action, method et names)
2. Le hacker va ensuite faire un faux site avec un formulaire caché identique et en renommant le bouton pour leurrer l'utilisateur

## Mise en place :

1. Le hacker va regarder soit l'inspecteur soit le code source de la page dans son navigateur pour identifier la configuration du formulaire sur le vrai site à répliquer (action, method et names)
2. Le hacker va ensuite faire un faux site avec un formulaire caché identique et en renommant le bouton pour leurrer l'utilisateur



# Le hacker regarde donc la source dans son navigateur

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta http-equiv="X-UA-Compatible" content="IE=edge">
6     <meta name="viewport" content="width=device-width, initial-scale=1.0">
7     <title>Patissor - Administration</title>
8 </head>
9 <body>
10     <form action="/admin/cake/add" method="POST">
11         <input type="text" name="addmin-cake-name" /><br />
12         <input type="text" name="addmin-cake-description" /><br />
13         <input type="text" name="addmin-cake-price" /><br />
14         <button type="submit">Créer</button>
15     </form>
16 </body>
17 </html>
```

Il en déduit que pour créer un gâteau, il faut un formulaire avec cette configuration :

- action : [www.patissor.com/admin/cake/add](http://www.patissor.com/admin/cake/add)
- method : POST
- names :
  - admin-cake-name
  - admin-cake-description
  - admin-cake-price

## Mise en place :

1. Le hacker va regarder soit l'inspecteur soit le code source de la page dans son navigateur pour identifier la configuration du formulaire sur le vrai site à répliquer (action, method et names)
2. Le hacker va ensuite faire un faux site avec un formulaire caché identique et en renommant le bouton pour leurrer l'utilisateur

# SELLING CAR

LA TRANSPARENCE DE LA TRANSACTION AUTOMOBILE

ACCUEIL

ACHETEZ VOTRE VOITURE

VENDEZ VOTRE VOITURE

CHANGEZ DE VOITURE

SPÉCIAL DEUX-ROUES

SERVICE CARTE GRISE



Contactez-nous



Rejoignez-nous



# SELLING CAR

LA TRANSPARENCE DE LA TRANSACTION AUTOMOBILE

ACCUEIL

ACHETEZ VOTRE VOITURE

VENDEZ VOTRE VOITURE

CHANGEZ DE VOITURE

SPÉCIAL DEUX-ROUES

SERVICE CARTE GRISE

Le hacker va cacher un formulaire  
identique ici



Contactez-nous



Rejoignez-nous



Voici le code que le hacker a mis pour faire le bouton Contactez-nous

```
✓ <form action="https://www.patissor/admin/cake/add" method="POST">  
  <input type="hidden" name="addmin-cake-name" value="Cake nul" /><br />  
  <input type="hidden" name="addmin-cake-description" value="Patissor est mauvais" /><br />  
  <input type="hidden" name="addmin-cake-price" value="3.4" /><br />  
✓ <button type="submit" class="bouton-gris">  
  <i class="icon icon-enveloppe"></i>  
  Contactez-nous  
</button>  
</form>
```

Ce bouton Contactez-nous est un formulaire caché pour créer un gateau sur patissor !

Il ne reste plus qu'à hacker à transmettre le lien par email à notre pâtissier.

Le pâtissier va cliquer dessus, se retrouver sur un site de vente de voiture, cliquer sur le bouton Contactez-nous et bam ! Il a créé en gâteau en tant qu'administrateur car il est connecté en parallèle sur patissor !

Le hacker a donc forcé le  
pâtissier a créé un gâteau à son  
insu !





# PREVENIR LES ATTAQUES CSRF

# Comment faire pour empêcher les attaques CSRF ?



Il faut rendre le formulaire  
incopiable !

# Comment rendre un formulaire incopiable ?



## Rendre un formulaire incopiable :

Rendre un formulaire incopiable est rendre le formulaire unique à chaque utilisateur.

En effet, s'il est unique à chaque utilisateur, le hacker qui regarde dans le code source de la page ne verra que **son formulaire à lui** et s'il le copie tel quel, **son formulaire à lui ne fonctionnera pas pour le patissier.**

Comment rendre un formulaire  
unique à chaque utilisateur ?



## Mise en place :

1. Il faut mettre automatiquement dans le formulaire **une donnée qui est propre à chaque utilisateur**. On peut par exemple ajouter automatiquement l'id de l'utilisateur dans le formulaire.
2. A chaque soumission du formulaire, **dans la fonction du controller, on vérifiera si la donnée est en concordance avec l'utilisateur connecté**



# Le hacker regarde donc la source dans son navigateur

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Patissor - Administration</title>
</head>
<body>
  <form action="/admin/cake/add" method="POST">
    <input type="hidden" name="admin-user-id" value="39" />
    <input type="text" name="addmin-cake-name" /><br />
    <input type="text" name="addmin-cake-description" /><br />
    <input type="text" name="addmin-cake-price" /><br />
    <button type="submit">Créer</button>
  </form>
</body>
</html>
```

# Le hacker regarde donc la source dans son navigateur

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Patissor - Administration</title>
</head>
<body>
  <form action="/admin/cake/add" method="POST">
    <input type="hidden" name="admin-user-id" value="39" />
    <input type="text" name="addmin-cake-name" /><br />
    <input type="text" name="addmin-cake-description" /><br />
    <input type="text" name="addmin-cake-price" /><br />
    <button type="submit">Créer</button>
  </form>
</body>
</html>
```

39 est l'id du hacker

Si le hacker copie ce formulaire dans son faux site pour que le patissier clique dessus, **cela ne marchera pas !**

En effet, **au backend, nous vérifierons** que si la donnée est en concordance avec l'utilisateur connecté :

**Comme 39 est l'id du hacker et non pas celui du patissier**, l'opération de créer un gateau sera vu comme une usurpation d'identité et sera invalidé !

Nous pouvons utiliser d'autres données que l'id de l'utilisateur pour **rendre le formulaire unique**.

Son adresse email, son prenom etc. ou encore mieux (et c'est ce qu'il se fait en réalité), **une référence utilisateur générée aléatoirement à chaque formulaire**.

Cette donnée unique à chaque utilisateur utilisée dans les formulaires est appelée un **token anti-CSRF** ou un **token CSRF**.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Patissor - Administration</title>
</head>
<body>
  <form action="/admin/cake/add" method="POST">
    <input type="hidden" name="admin-user-id" value="39" />
    <input type="text" name="addmin-cake-name" /><br />
    <input type="text" name="addmin-cake-description" /><br />
    <input type="text" name="addmin-cake-price" /><br />
    <button type="submit">Créer</button>
  </form>
</body>
</html>
```

39 est le token anti-CSRF

