

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

LES ATTAQUES CSRF

Avez-vous déjà entendu parler
de CSRF ?



Répondons d'abord à une question !

Un patissier ajoute des gateaux sur son site.
Il peut le faire car il est administrateur du site.



Un hacker n'est pas administrateur, mais
peut-il forcer le patissier à créer un gateau
sur son site ?

Oui !



A LA PLACE D'UN HACKER ...

Quelle stratégie pour forcer le
pâtissier ?



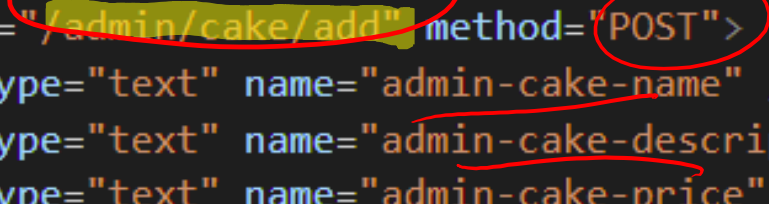
Indice : **tout le monde** (y compris un hacker)
peut écrire un formulaire et changer l'*action*
pour envoyer des données à notre back end



Réponse ...

1. On **construit un faux site** de vente de voitures
2. On construit un **formulaire d'ajout** de gateaux dans notre faux site, mais **dont l'action va vers le vrai site**
3. On **met des valeurs aux inputs** et on **les rend invisibles** en mettant **type="hidden"**
4. On fait de tel sorte que le patissier clique sur le bouton de soumission pour qu'il soumette les données vers le vrai site **en son nom** (son compte sera utilisé car c'est lui qui a cliqué)

Sur le site du patissier www.patissor.com, on voit dans l'inspecteur ou dans le code source de la page dans le navigateur (clic-droit sur la page dans Chrome > Voir la source) :



```
<form action="/admin/cake/add" method="POST">
  <input type="text" name="admin-cake-name" /><br />
  <input type="text" name="admin-cake-description" /><br />
  <input type="text" name="admin-cake-price" /><br />
  <button type="submit">Créer</button>
</form>
```

On en déduit que pour créer un geteau, il faut un formulaire avec cette configuration :

- action : www.patissor.com/admin/cake/add
- method : POST
- names :
 - admin-cake-name
 - admin-cake-description
 - admin-cake-price

1. Ensuite on fait un faux site (www.voiture-vendue.fr) comme un site de vente de voiture
2. On fait un formulaire identique, mais en **cachant les inputs** et en leur donnant des **valeurs par défaut** sur notre faux site
3. On change aussi le **libellé du bouton** pour tromper le visiteur

```
<form action="https://www.patissor.com/admin/cake/add" method="POST">  
  <input type="hidden" name="admin-cake-name" value="Pouet Cake" /><br />  
  <input type="hidden" name="admin-cake-description" value="Cake mauvais" /><br />  
  <input type="hidden" name="admin-cake-price" value="3.54" /><br />  
  <button type="submit">Contactez-nous pour vendre votre voiture</button>  
</form>
```

Que se passe-t-il si le patissier (connecté comme administrateur sur patissor.com) va sur le site voiture-vendu.fr et clique sur **Contactez-nous pour vendre votre voiture** ?

Le patissier, à son insu, va créer un gateau qui va plomber sa réputation !

Il ne reste plus qu'à hacker de lui transmettre le lien par email.

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

C'est une attaque **Cross-Site
Request Forgery** (CSRF)

Cross-Site : sites croisés (faux sites vers site réel)

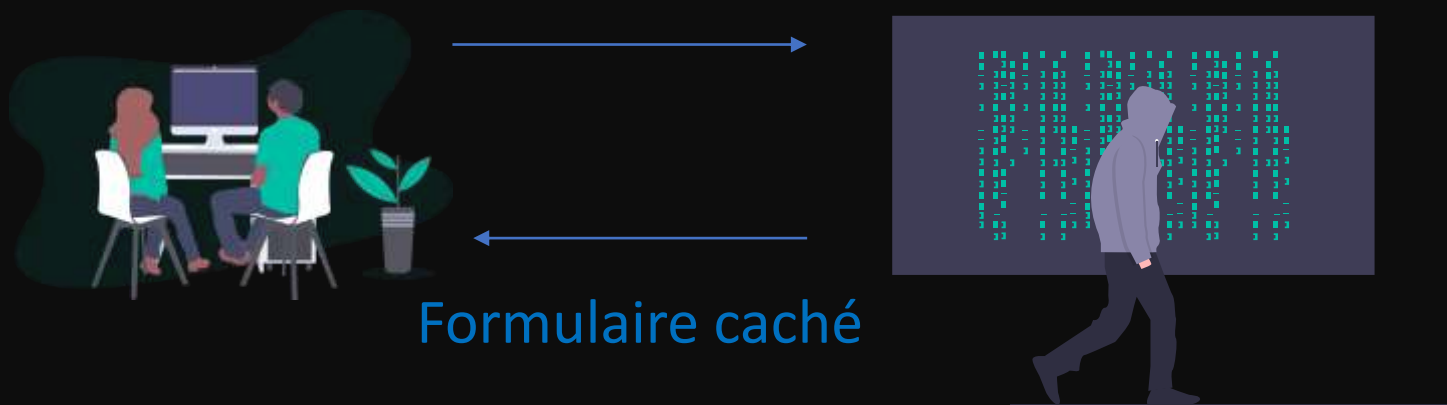
Request : requête

Forgery : falsification

L'idée d'une attaque CSRF : le hacker est capable de forcer un administrateur ou un utilisateur à faire une requête [request] à son insu [forgery] sur un faux site vers un site réel [cross-site]

1) Le visiteur va sur le faux site

Voiture-vendu.fr
(fake)



2) Lorsque le visiteur soumet le formulaire caché



patissor.com/admin/cake/add

Le visiteur s'est fait avoir !



A LA PLACE D'UN HACKER ...

Objectifs :

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

Mise en place :

1. Récupérer la configuration du formulaire sur le vrai site
2. Faire un faux site avec un formulaire caché et en renommant le bouton pour le leurrer

