TAMDA Zohir
BTS SIO 2 option SLAM
Bloc 3 - Cybersécurité

Le 8 octobre 2024

Force brute sur un mot de passe

Sommaire

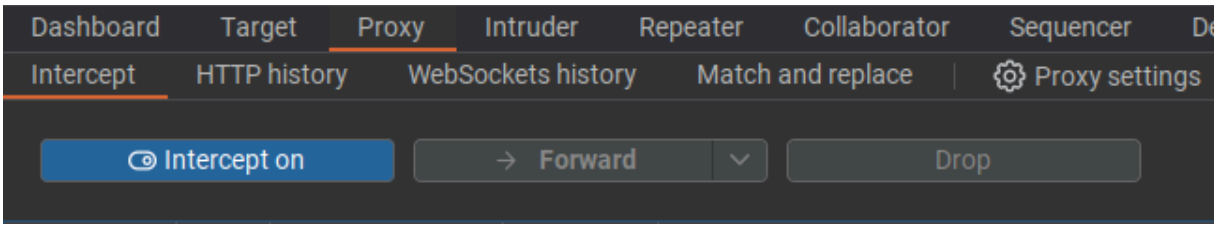
Définition

L’attaque brute force consiste à connaître le login et le mot de passe de la victime en essayant de se connecter à plusieurs reprises.

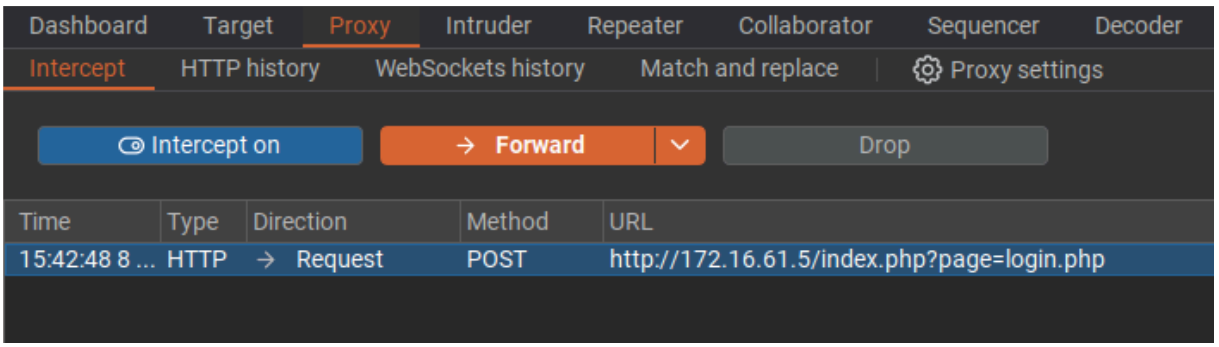
Force brute sur un mot de passe

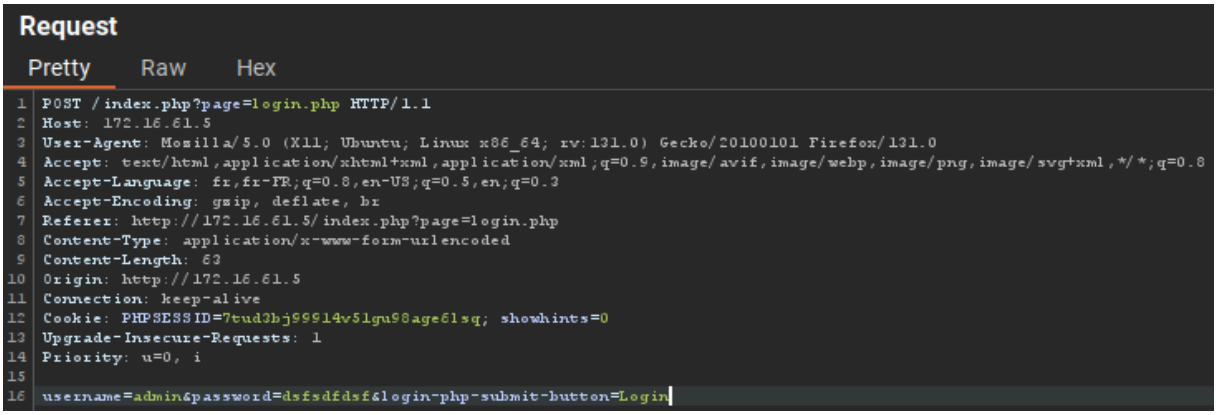
*L'utilisateur **admin** va être la cible pour permettre de se connecter à ce compte. Avec l'attaque brute force, plusieurs combinaisons de mots de passe vont être utilisés séquentiellement.*
Sur le site web "mutillidae", le niveau de sécurité est défini à 0, ce qui permet d'effectuer des attaques.

Côté proxy, l'interception est activée pour permettre de récupérer des informations sur les requêtes ainsi que des réponses.

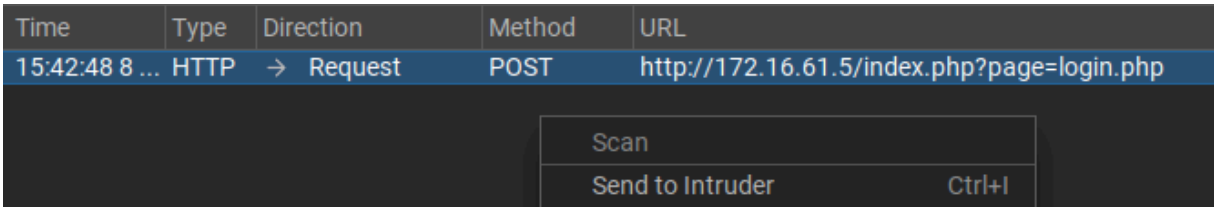


Dans le formulaire, le nom d'utilisateur est **admin** et le mot de passe erroné a été saisi pour intercepter la requête dans BurpSuite.

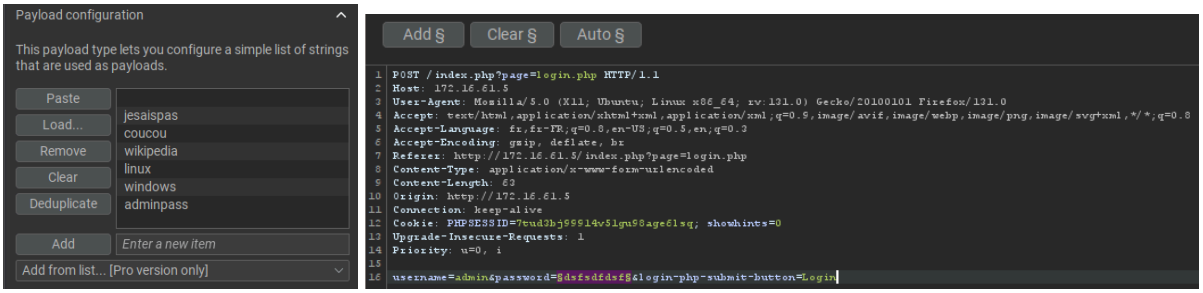




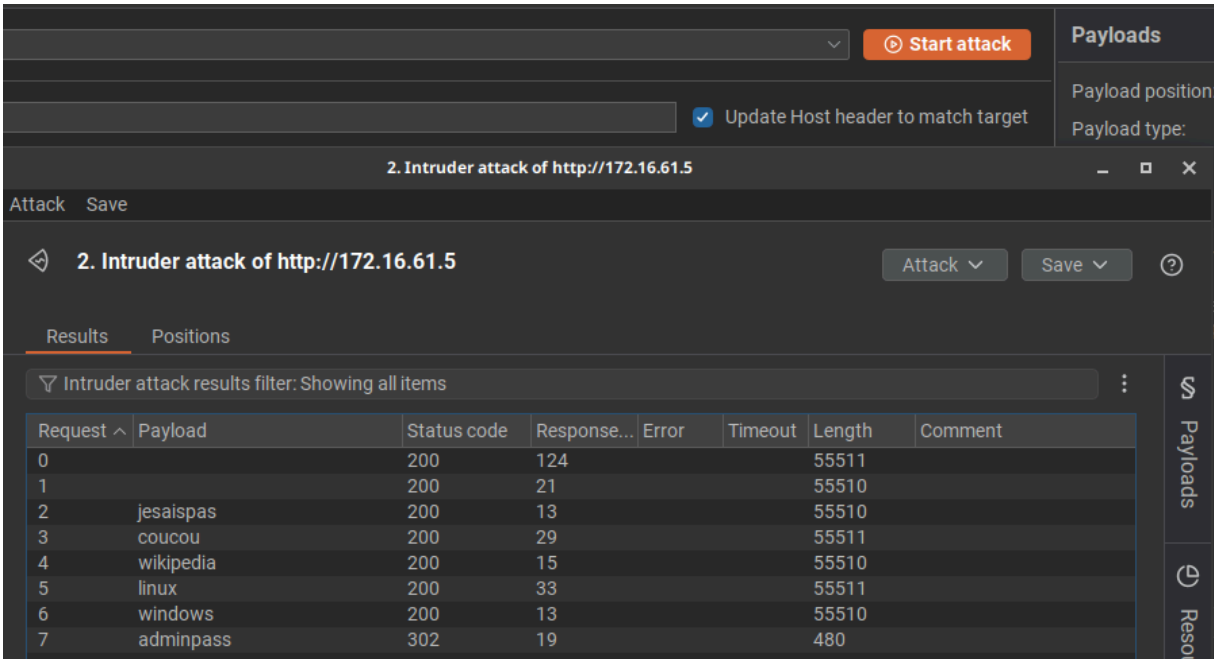
Résultat, au niveau de la requête, on peut voir à la dernière ligne qu'il y a deux paramètres : username (comme valeur "admin") et password avec la valeur un mot de passe erroné. Avec cette requête, on peut tester toutes les combinaisons des mots de passe : c'est l'intruder qui se charge de ça.
Pour cela, cliquer sur "Send to Intruder" à partir du menu.



Dans l'intruder, on met une liste de mots de passe pour permettre de se connecter au compte **admin**. Côté requête, il faut remplacer la valeur de la clé "password" par cette liste de mots de passe qui vont être testées une par une.
Pour cela, sélectionner le mot de passe erroné et cliquer sur le bouton *"Add §"*



Pour lancer l'attaque brute force, cliquer sur le bouton "Start Attack".



Dans le tableau présenté ci-dessus, on constate que tous les payloads (qui sont en réalité des mots de passe) obtiennent un statut de type 200, pour dire que la réponse a bien été reçue.
Par contre, dans la dernière ligne du tableau, on remarque que le statut n'est pas le même que celui des autres : elle est de type 302, désignant une redirection. Donc cela signifie que le mot de passe nommé "adminpass" correspond bien à celui du compte admin.

Codage sécurisé et analyse du code source

Après avoir réalisé ce type d'attaque, côté du site "mutillidae", le niveau de sécurité est défini à 5. Les procédures sont les mêmes pour réaliser l'attaque brute force.

Même si le niveau de sécurité est élevé à la fois côté client et côté serveur, ça ne suffit pas pour éviter ce type d'attaque car dans la ligne du tableau où il y a "adminpass", on obtient toujours le statut de redirection HTTP, signifiant que l'attaquant a réussi à se connecter en tant qu'administrateur.

```
case "5": // This code is fairly secure
        /*
         * Concerning SQL Injection, use parameterized stored procedures. Parameterized
         * queries is not good enough. You cannot use least privilege with queries.
         */
        $lEnableJavaScriptValidation = TRUE;
        $lEnableHTMLControls = TRUE;
        $lProtectAgainstMethodTampering = TRUE;
        $lEncodeOutput = TRUE;
        break;
```

Grâce à la variable **`$lEnableHTMLControls`**, on peut ajouter des attributs dans la balise **`<input>`** pour ajouter des vérifications côté client, dont notamment le nombre de caractères minimales dans le champ du mot de passe.

```
<input type="password" name="password" size="15"
        <?php
                if ($lEnableHTMLControls) {
                        echo('minlength="1" maxlength="15" required="required"');
                }// end if
        ?>
/>
```

Dans la balise **`<input>`** de type **`password`**, on voit qu'on peut mettre au minimum 1 caractère et au maximum 15 caractères.

```
<?php
        if($lEnableJavaScriptValidation){
                echo "var lValidateInput = \"TRUE\"" . PHP_EOL;
        }else{
                echo "var lValidateInput = \"FALSE\"" . PHP_EOL;
        }// end if
?>

function onSubmitOfForm(/*HTMLFormElement*/ theForm){
        try{
                if(lValidateInput == "TRUE"){
                        var lUnsafeCharacters = /[\W]/g;
                        if (theForm.username.value.length > 15 ||
                                theForm.password.value.length > 15){
                                        alert('Username too long. We c

                                        return false;
                        };// end if
```

Puis, côté JavaScript, on vérifie si le nom d'utilisateur ou le mot de passe ne dépasse pas 15 caractères. Sinon, il affichera une alerte.

Même avec le niveau de sécurité définie à 5, ça ne suffit pas pour forcer les utilisateurs à sécuriser leurs mots de passe.

Pour faire face à cette faille de sécurité, il est possible :

- de faire des vérifications côté serveur car cela permet à l'utilisateur de ne pas manipuler les validations à la fois dans le HTML et dans le JavaScript et cela ajoute une couche de sécurité supplémentaire à l'application web.
- de limiter le nombre de tentatives de connexion à un compte. Par exemple, si on a raté 5 fois, alors l'application va mettre en place un délai de plusieurs minutes voire plusieurs heures pour atténuer l'attaque brute force.