Opdracht 2.1: Wiki

Op de (fictieve) website van de UU is er ook een Wiki te vinden. Studenten kunnen hierop informatie toevoegen of verbeteren door pagina's te wijzigen. Gegeven is een deel van de code.

1. **Wat doet het? Beschrijf de functionaliteit.**

    // if ($_POST && array_key_exists('new_contents', $_POST)
    In deze regel wordt ervoor gezorgd dat zolang er keys zijn (dus alle waarden in de array), de inhoud van de methode uitgevoerd wordt'. $_POST zorgt ervoor dat waarden automatisch vanaf hun index vorm vertaald worden naar de namen van de variabelen.

    // file_put_contents($this_file, $_POST['new_contents']);
    Deze regel zorgt ervoor dat alle waardes bij de key $_POST['new_contents'] in $this_file worden gezet.

    // header("Location: http://" .
    $_SERVER['HTTP_HOST'] .
    dirname($_SERVER['PHP_SELF']) . "/" . $this_file);
    Hier worden de header en url ingevuld.

In het HTML bestand wordt een tabel aangemaakt waarin de bovenstaande acties plaatsvinden. In dit bestand staat dus dat je tekst toe kunt voegen en aan kunt passen.

Al met al, op de Wiki kun je de code zien van de pagina zelf en kun je die aanpassen.

2. Omdat de code zelf online gezet wordt en er in die code direct aanpassingen gemaakt kunnen worden, is een manier om dit te slopen de if-statement in de PHP code of de regel ***file_get_contents($this_file)*** in de HTML code verwijderen of aanpassen, zodat het niet meer zichtbaar is. Dan kan niemand er meer bij.

Een mogelijke oplossing is een restrictie zetten op code die er al staat weghalen, door bijvoorbeeld het bestand te vergelijken met een ander bestand met daarin de basis code. Als deze basis code niet geheel in de code op de website wordt bevat, is er iets weggehaald wat niet mag. Dan zou dit een foutmelding op moeten leveren.

3. Door een inlogsysteem toe te voegen dat oppopt nadat iemand iets probeert te submitten, kunnen studenten en docenten inloggen om de pagina's te wijzigen. Iemand zonder deze inloggegevens kan dus niets aanpassen, maar het wel lezen.

    Hiervoor heb je een aantal extra PHP bestanden nodig: main_login, check_login and login_success en een members tabel. De namen van de bestanden komen overeen met de inhoud ervan. Ook kan er een logout.php bestand toegevoegd worden om de sessie te beëindigen.

4. **Een Wiki die op deze manier gebouwd is, geeft gebruikers veel mogelijkheden, bijvoorbeeld het uitvoeren van willekeurige functies. Dit schrikt de systeembeheerders af: ze willen graag controle hebben over welke functies er niet uitgevoerd mogen worden. In een bestand youshallnot.txt (een functie per regel) willen ze dit vastleggen. Beschrijf hoe je de code van de Wiki zou aanpassen om dit bestand te gebruiken en ervoor te zorgen dat de genoemde functies niet gebruikt kunnen worden, zonder dat de functionaliteit van de Wiki verloren gaat.**

Je kunt de file inladen en deze met youshallnot.txt vergelijken om te kijken of een van de functies die in youshallnot.txt staat ook in $this_file staan. Als dit het geval is, moet je een foutmelding geven.

TODO: HOE CODE AANPASSEN?

Code van ficitieve wiki:

```
<?php
$this_file = basename(__FILE__);
if ( $_POST && array_key_exists('new_contents', $_POST) ) {
   file_put_contents($this_file, $_POST['new_contents']);
   header( "Location: http://" .
   $_SERVER['HTTP_HOST'] .
   dirname($_SERVER['PHP_SELF']) . "/" . $this_file );
     exit;
}
?>

<!DOCTYPE html>
<html>
        <head>
                <title>UU Wiki</title>
         </head>
         <body>
                <form method="POST">
                        <textarea name="new_contents" rows="70" cols="70"><?=
                        htmlspecialchars( file_get_contents($this_file) );
                ?></textarea>
                <input type="submit" value="Submit"/>
                </form>
        </body>
</html>
```

http://php.net/manual/en/reserved.variables.post.php

https://www.owasp.org/index.php/PHP_Security_Cheat_Sheet#Input_handling