

Módulo	3. Desarrollo Seguro
Nombre y apellidos	Iván Andrés Mendoza Dávila
Fecha de entrega	2021-01-11

Caso Práctico

Para poder realizar el ejercicio utilizaremos la aplicación **wackopicko**.

Puedes encontrar esta aplicación en la máquina virtual de (<http://www.vulnerablewebapps.org/>) utilizada en el caso práctico del módulo 2.

OBJETIVO 1

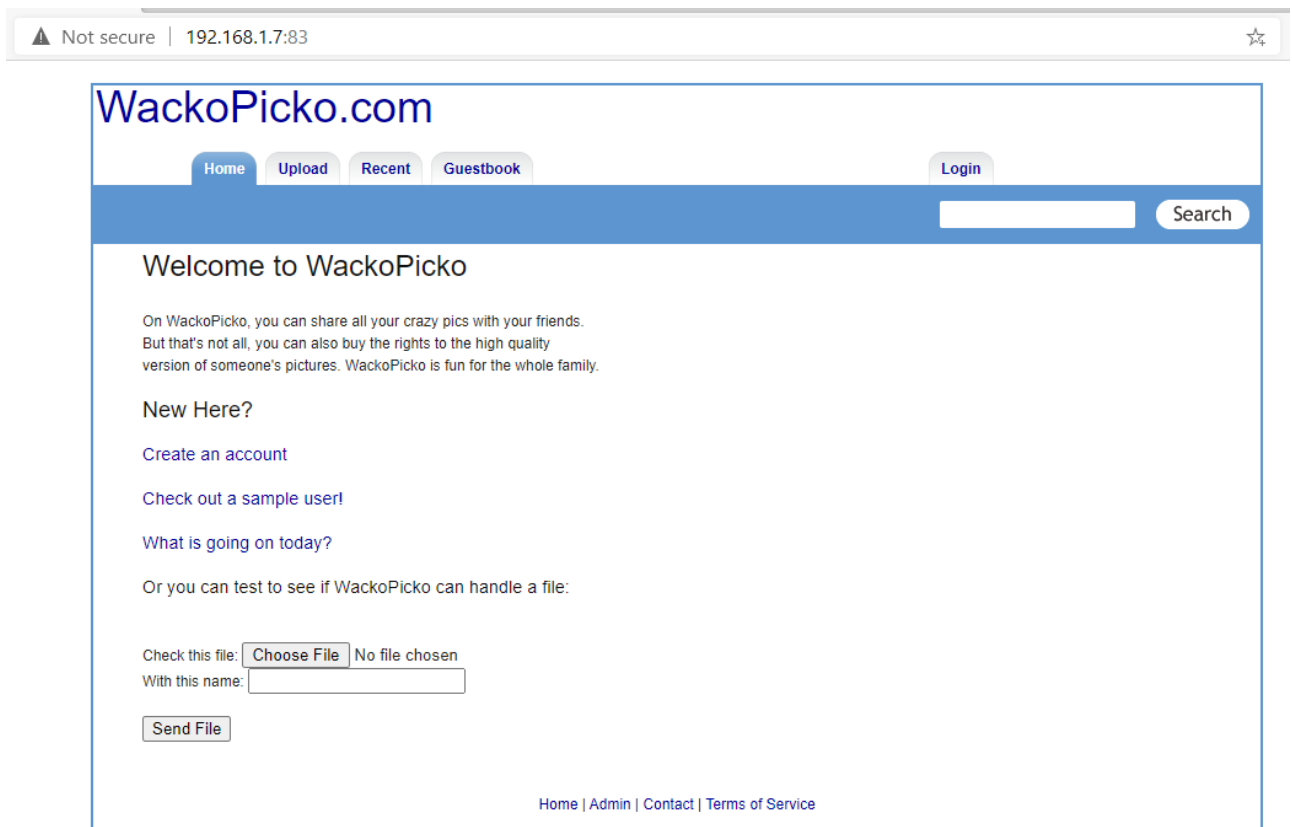
Identificar las vulnerabilidades que tiene la aplicación (utilizando una herramienta semiautomática como Vega u OWASP Zap).

- Como resultado de este ejercicio, se mostrará, por una parte, la cantidad de ocurrencias identificadas y clasificadas según su nivel y, por la otra, la clasificación de todas estas ocurrencias en tipologías de vulnerabilidades.

En primer lugar identificamos la dirección IP de la máquina virtual de Ubuntu con el comando *ifconfig / more* :

```
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.7 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::20c:29ff:fe7e:e3e9 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:7e:e3:e9 txqueuelen 1000 (Ethernet)
    RX packets 132135 bytes 97967719 (97.9 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 116644 bytes 147794868 (147.7 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

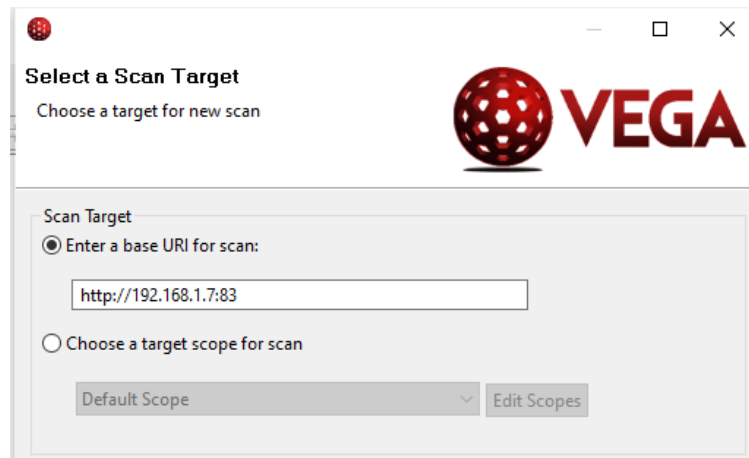
Con esta dirección IP comprobamos el acceso a la url de wackopicko en el puerto 83 de acuerdo a lo indicado en el sitio web antes mencionado.



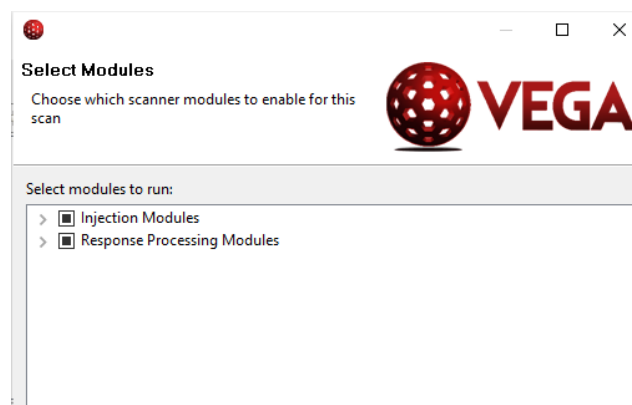
La herramienta que se ha escogido para el escaneo es VEGA, ya que la encontré más amigable y fácil de entender, para realizar el escaneo escogemos la opción Start New Scan



Esta opción permite agregar la url del sitio web objetivo:



Luego se escoge los módulos que se desea incluir en el escaneo:



Una vez hecho esto el programa procede a realizar el escaneo, el cual tomó alrededor de 30 minutos, los resultados que arrojó son:

Por el riesgo que representan:

Tipo de Vulnerabilidad	Nivel de Riesgo
Http Only and Secure Flag	Alto
SQL Injection	Alto
Cross Site Scripting	Alto
Page Fingerprint Differential Detected	Medio
Logic Flaw	Medio

Clasificación por su tipo:

Tipo de Vulnerabilidad	Clasificación Owasp Top Ten
Http Only and Secure Flag	A3-2017 Sensitiva Data Exposure
SQL Injection	A1-2017 Injection
Cross Site Scripting	A7-2017 XSS
Page Fingerprint Differential Detected	A6-2017 Security Misconfiguration
Logic Flaw	A5-2017 Broken Access Control

OBJETIVO 2

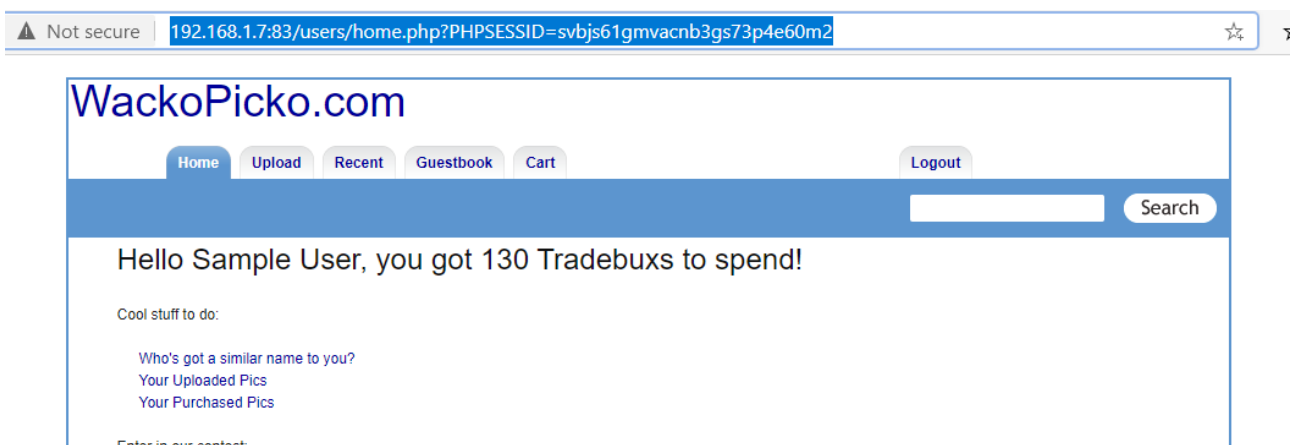
Explotar las vulnerabilidades, indicando cómo se ha explotado cada vulnerabilidad y cuál ha sido el resultado (adjuntando pantallazos).

Sesión de Cookie sin uso de banderas Http Only y Secure

Vega encontró que este sitio web no usa las banderas HttpOnly y Secure, dado que las sesiones de cookies son autenticación de credenciales, si un atacante accede a esta información puede usar Cross Site Scripting.

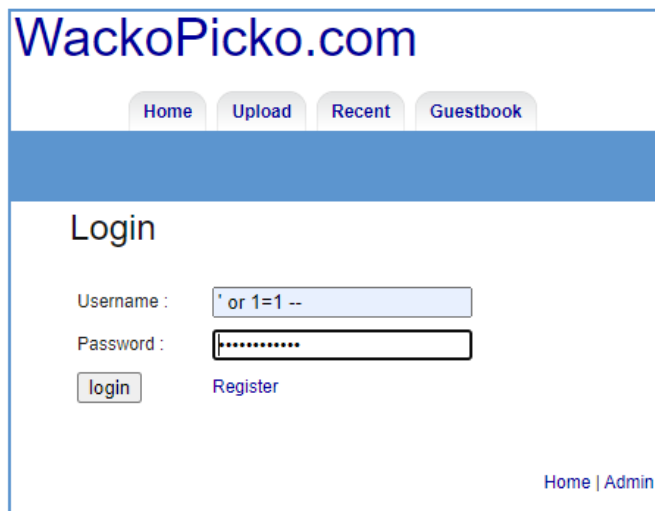
```
HTTP/1.1 200 OK
Date: Wed, 06 Jan 2021 01:14:19 GMT
Server: Apache/2.4.7 (Ubuntu)
X-Powered-By: PHP/5.5.9-1ubuntu4.24
Set-Cookie: PHPSESSID=svbjs61gmvacnb3gs73p4e60m2; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-cache, no-store
Pragma: no-cache
Vary: Accept-Encoding
Content-Length: 3339
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html
```

Este identificador de PHPSESSID se usará en un comando del tipo:



El cual queda abierto para que desde otro navegador incluso otra computadora pueda mantenerse en la misma sesión usando dicha ruta de la barra de direcciones.

SQL Injection



WackoPicko.com

Home Upload Recent Guestbook

Login

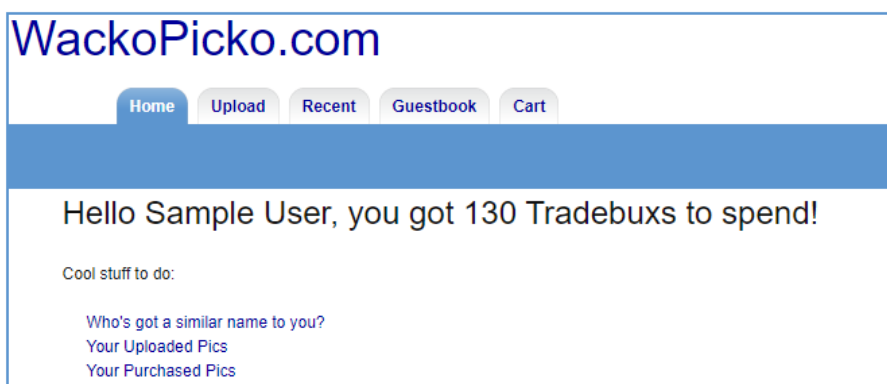
Username : ' or 1=1 --

Password :

login Register

Home | Admin

Si usamos la forma ' or 1=1 -- entraremos en la consulta y nos responderá con el usuario o usuarios existentes en la base de datos, si colocamos cualquier valor en el campo:



WackoPicko.com

Home Upload Recent Guestbook Cart

Hello Sample User, you got 130 Tradebuxs to spend!

Cool stuff to do:

Who's got a similar name to you?

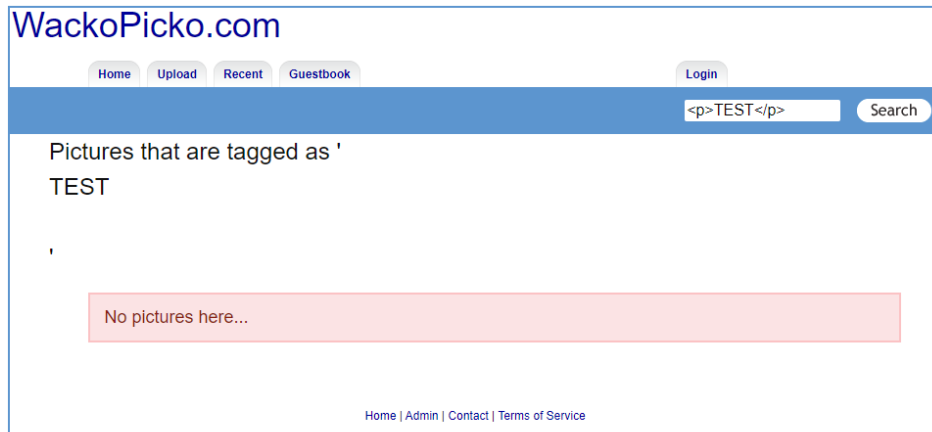
Your Uploaded Pics

Your Purchased Pics

Vemos que nos da acceso como el usuario "Sample User" lo que se traduce en una falla de seguridad porque hemos tenido acceso a la base de datos y al sitio web.

Cross Site Scripting

En primer lugar se accede a la url que especificó VEGA para esta vulnerabilidad, la cual nos menciona está en la caja de búsqueda imágenes, hacemos la prueba de inserción de un comando que nos permite inyectar el código html, en este caso accederemos el código `<p>TEST</p>`, así vemos que:



En la ventana imprime el código HTML deseado especificado en la caja de texto de búsqueda, de esto podemos colegir que es posible inyectar código de tipo javascript, para esto se hace la prueba de concepto y se hace la inserción del código `<script>alert('test')</script>`, y se obtiene lo siguiente:

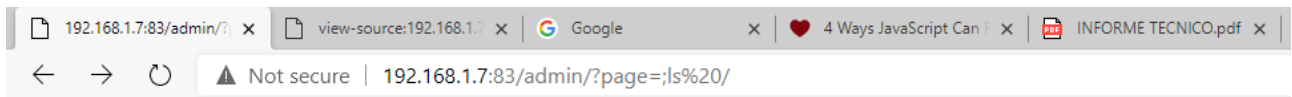


Ahora corroboramos que el código tipo script fue insertado exitosamente, esto puede usarse por un atacante para dañar la página, acceder a una cookie o redireccionar/abrir una nueva url, en este caso vamos a abrir una nueva ventana con la url de ejemplo www.virus.com con el código `<script>window.open("http://www.virus.com");</script>` y se presenta el siguiente escenario:



Page Fingerprint Differential Detected

De acuerdo a esta vulnerabilidad una posible inclusión de archivos puede presentarse, esto se puede ver al escribir la url 192.168.1.7:83/admin/?page=../../../../, lo cual indica la siguiente ventana:



Warning: require_once(): failed to open stream: No such file or directory in /app/admin/index.php on line 4

Fatal error: require_once(): Failed opening required 'ls /php' (include_path='.:usr/share/php:usr/share/pear') in /app/admin/index.php on line 4

Esto es una manera de enumerar directorios del servidor lo que implica una falla de seguridad, también podría permitir una posible inclusión de archivos en un directorio, el cual generalmente es un el archivo /etc/passwd.

Local File Sytem Path Found

Vega detectó que el sitio muestra una ruta del sistema con la url 192.168.1.7:83/admin/

► RESOURCE CONTENT

```
/usr/share/php
```

Esta información es sensible ya que puede revelar parámetros del ambiente del servidor a los atacantes, nunca se debe enviar mensajes de respuesta con este tipo de información hacia el lado del cliente, debe ser re-direccionado hacia otro tipo de respuesta, puede ser a un log de errores para su análisis.

Logic Flaw

Recorriendo la página en la ficha Home, podemos ver un enlace que dice: Whats is going on today?

WackoPicko.com

[Home](#) [Upload](#) [Recent](#) [Guestbook](#) [Cart](#)

Welcome to WackoPicko

On WackoPicko, you can share all your crazy pics with your friends.
But that's not all, you can also buy the rights to the high quality
version of someone's pictures. WackoPicko is fun for the whole family.

New Here?

[Create an account](#)

[Check out a sample user!](#)

[What is going on today?](#)

En este enlace si adelantamos con el boton *What about tommorrow?* Llegamos a encontrar un cupon valido para días posteriores, con este cupon intentamos agregar en el carro de compras una imagen.

WackoPicko.com

[Home](#) [Upload](#) [Recent](#) [Guestbook](#) [Cart](#)

WackoPicko Calendar

What is going on Friday 22nd of January 2021?


We're throwing a party!

Use this coupon code: SUPERYOU21 for 10% off in celebration!

[What about tomorrow?](#)

[Home](#) | [Admin](#) | [Contact](#) | [Terms of Service](#)

Welcome to your cart Sample User

Pic name	Sample Pic	Price
The house I share		20 Tradebux
Coupon Code		Coupon Amount
SUPERYOU21		10% Off
SUPERYOU21		10% Off
SUPERYOU21		10% Off
SUPERYOU21		10% Off
SUPERYOU21		10% Off
SUPERYOU21		10% Off
SUPERYOU21		10% Off
SUPERYOU21		10% Off
SUPERYOU21		10% Off
SUPERYOU21		10% Off
Remove From Cart		

Al agregar múltiples veces el cupon a la cuenta, nos permite hacerlo varias veces y el precio del ítem baja considerablemente, este es un error ya que permite al usuario ejecutar una función sin límite lo que perjudica al negocio.

Confirm your purchase Sample User		
Pic name	High Quality Link	Price
The house I share	http://192.168.1.7:83/pictures/high_quality.php?picid=14&key=MzM4OTU3MA%3D%3D	20 Tradebux
Total : 6.973568802 Tradebux		
Purchase		
Home Admin Contact Terms of Service		


Así podemos ver que del precio de 20 Tradeux podemos obtener el ítem por 6.97 Tradeux por ejemplo.

Objetivo 3

Proponer, al menos, una medida de prevención para cada vulnerabilidad analizada (se entiende como medidas de prevención las modificaciones en el código donde se encuentra el problema). Asimismo, se podrán indicar medidas adicionales sobre buenas prácticas de seguridad relativas a la tipología de la vulnerabilidad.

Sesión de Cookie sin uso de banderas Http Only y Secure

La forma de remediación es configurar las banderas con valor TRUE el archivo php.ini que se encuentra en la ruta:

 Vulnerable Web Apps - VMware Workstation 16 Player (Non-commercial use only)

Player ▾ |   

```
root@ccc607c6b6e93:/etc/php5/apache2# sudo find -name php.ini
./php.ini
root@ccc607c6b6e93:/etc/php5/apache2#
```

Para esto se debe usar el comando :

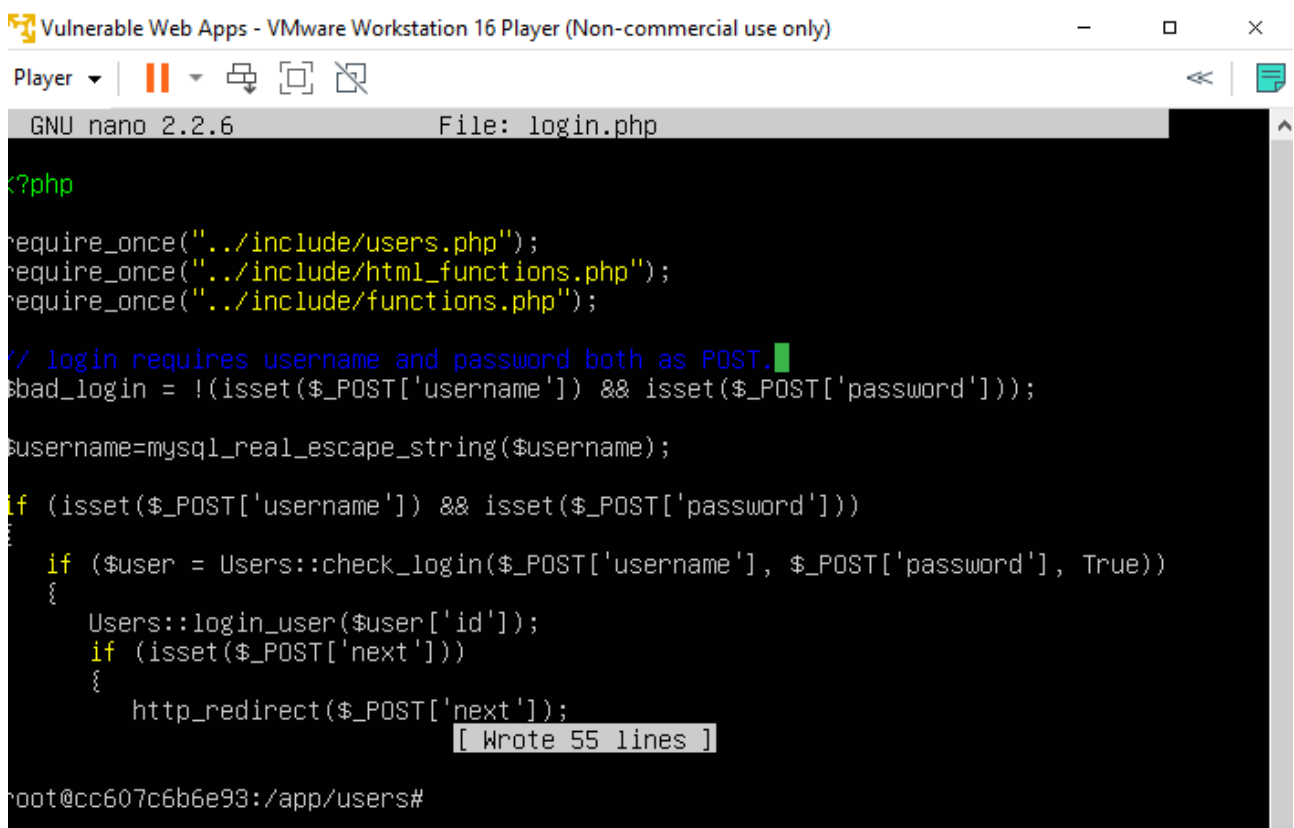
```
setcookie($name, $value, $expire, $path, $domain, $secure, $httponly);
```

Lo cual se vería así:

```
setcookie($name, $value, $expire, $path, $domain, true, true);
```

SQL Injection

Para remediar debemos escapar los caracteres propios de mysql para consulta en la base de datos, para esto usamos el comando: `mysql_real_escape_string`



```
GNU nano 2.2.6 File: login.php

<?php

require_once("../include/users.php");
require_once("../include/html_functions.php");
require_once("../include/functions.php");

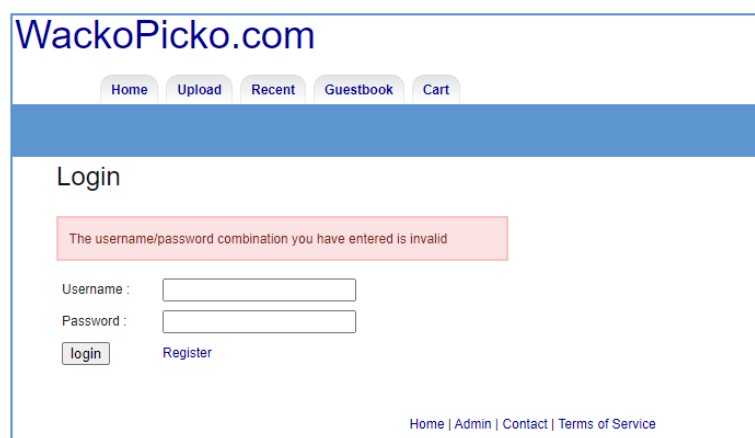
// login requires username and password both as POST.
bad_login = !(isset($_POST['username']) && isset($_POST['password']));

$username=mysql_real_escape_string($username);

if (isset($_POST['username']) && isset($_POST['password']))
{
    if ($user = Users::check_login($_POST['username'], $_POST['password'], True))
    {
        Users::login_user($user['id']);
        if (isset($_POST['next']))
        {
            http_redirect($_POST['next']);
        }
    }
}

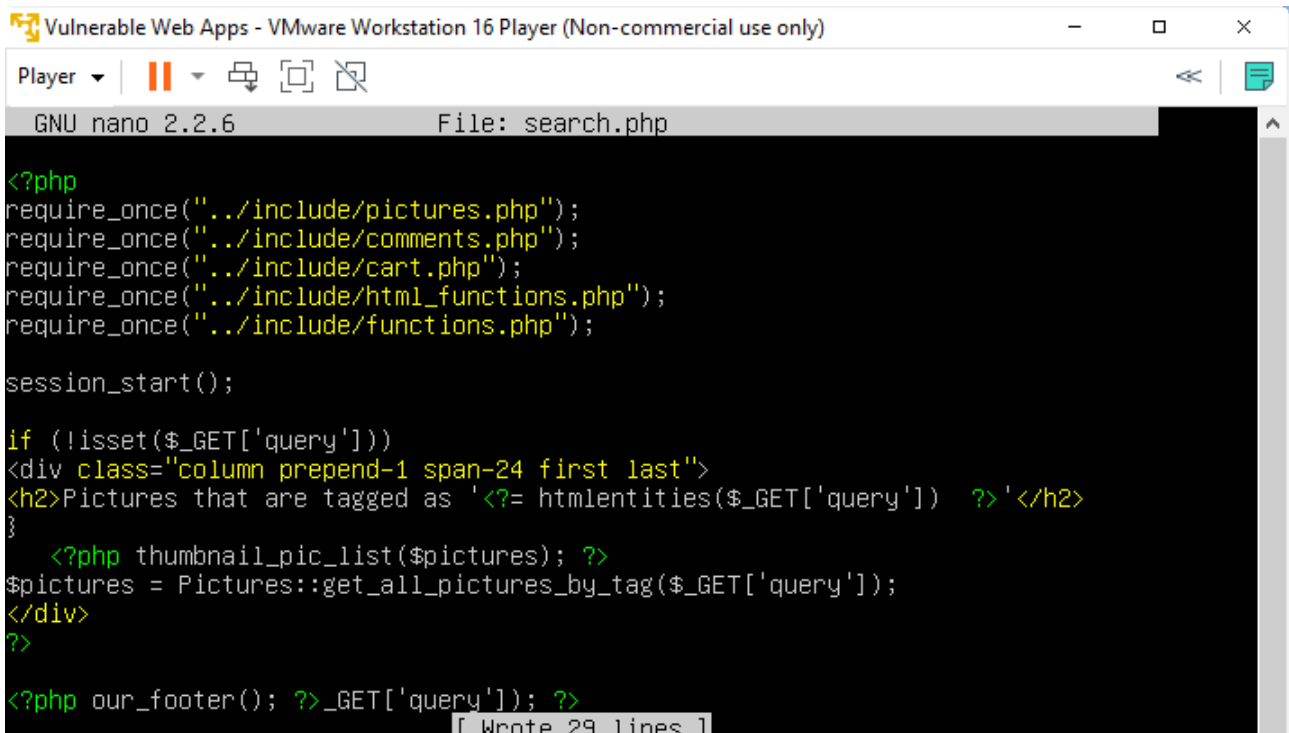
root@ccc607c6b6e93:/app/users#
```

Ahora al ingresar los comandos de prueba nos muestra un error y ya no nos permite acceder como Sample User



Cross Site Scripting

Para remediar este caso, se debe evitar que se muestre caracteres como “ ‘ ”, que van a permitir dar la pauta de que se puede inyectar el código html y específicamente los script, para esto usamos el comando html entities antes de GET ‘query’, como se muestra en la imagen.



```
<?php
require_once("../include/pictures.php");
require_once("../include/comments.php");
require_once("../include/cart.php");
require_once("../include/html_functions.php");
require_once("../include/functions.php");

session_start();

if (!isset($_GET['query']))
<div class="column prepend-1 span-24 first last">
<h2>Pictures that are tagged as '= htmlentities($_GET['query']) ?'</h2>
}

<?php thumbnail_pic_list($pictures); ?>
$pictures = Pictures::get_all_pictures_by_tag($_GET['query']);
</div>
?>

<?php our_footer(); ?>_GET['query']; ?>
```

Así al ingresar el código html para un script, por ejemplo que la página emita un cuadro de alerta, nos mostrará de la siguiente manera:



Ahora ya no es posible para un atacante insertar el código para ejecutar un script.

