Vulnerability Name:	Weak Session IDs
Affected Vendor:	DVWA
Affected Product Name:	http://dvwa/vulnerabilities/weak_id/
Product Official Website URL:	http://dvwa/login.php
Affected Component:	Weak Session IDs

Description: - A weak session ID refers to a session identifier that is predictable, guessable, or easily brute-forced, making it vulnerable to session hijacking, fixation, or impersonation attacks. Session IDs are meant to uniquely identify a user's session in a web application, but if they are weak, attackers can exploit them to gain unauthorized access.

Root Cause: - Insufficient entropy, insecure random number generation, predictable patterns, lack of session id regeneration, Inadequate Session Management.

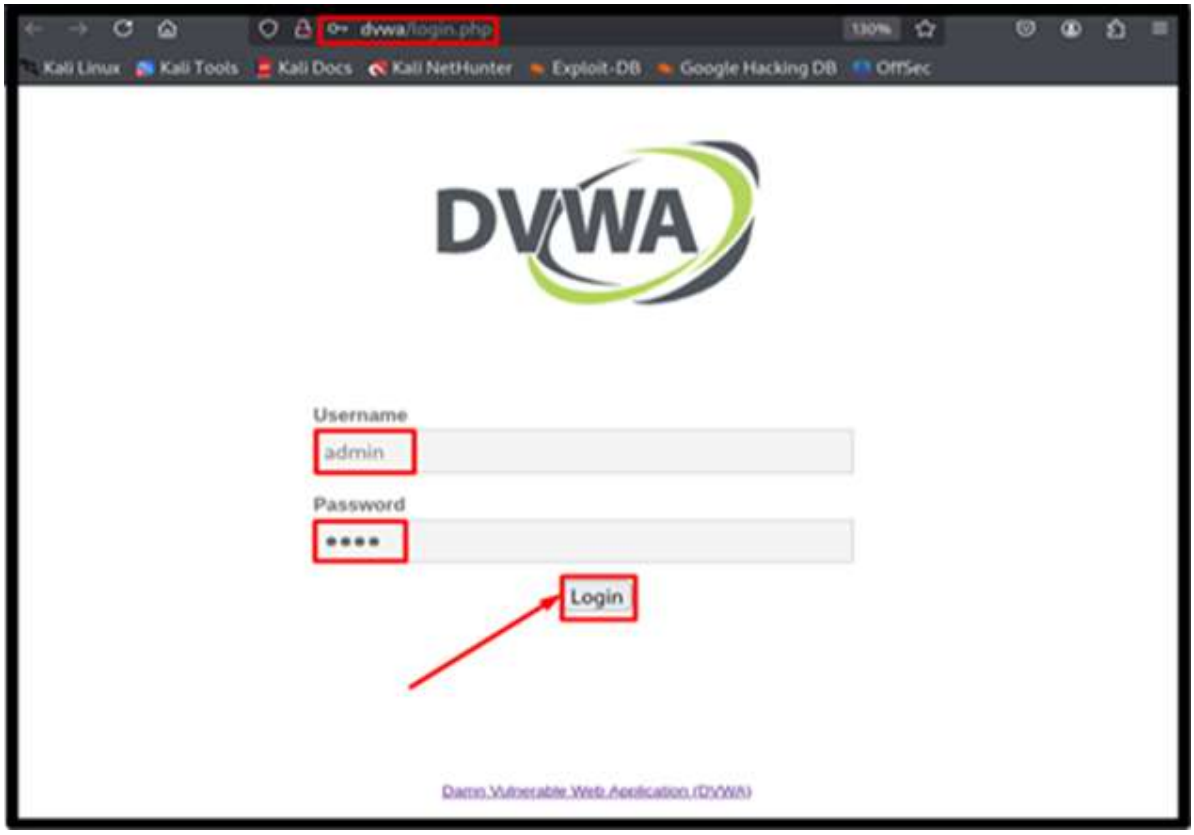
Impact: - Weak session IDs can have severe security consequences, as they allow attackers to hijack user sessions, impersonate users, and gain unauthorized access to sensitive information or functionalities within an application.

Mitigation: - Use cryptographically secure random session IDs (e.g., generated using SHA-256 or UUIDs). ensure session IDs are long and complex (128 bits or more). regenerate session IDs upon authentication and privilege escalation. Store session IDs in secure, HTTP-only cookies instead of URLs. Implement session expiration and inactivity timeouts**.**

Remediation: - To Remediation of Weak Session id Use Secure Random Number Generators: Generate session IDs using cryptographically secure random number generators. Increase Entropy: Increase the entropy of session IDs to make them less predictable. Regenerate Session IDs: Regenerate session IDs after a certain period or event, such as after logout. Implement Secure Session Management: Implement secure session management practices, such as invalidating sessions after logout. use HTTPS: Use HTTPS to encrypt session IDs in transit. use Secure Cookie Flags: Use secure cookie flags, such as "Secure" and "HttpOnly", to protect session IDs.

Proof Of Concept

Step: - First navigate to http://dvwa/login.php and login with username and Password.



Security Level :- Low

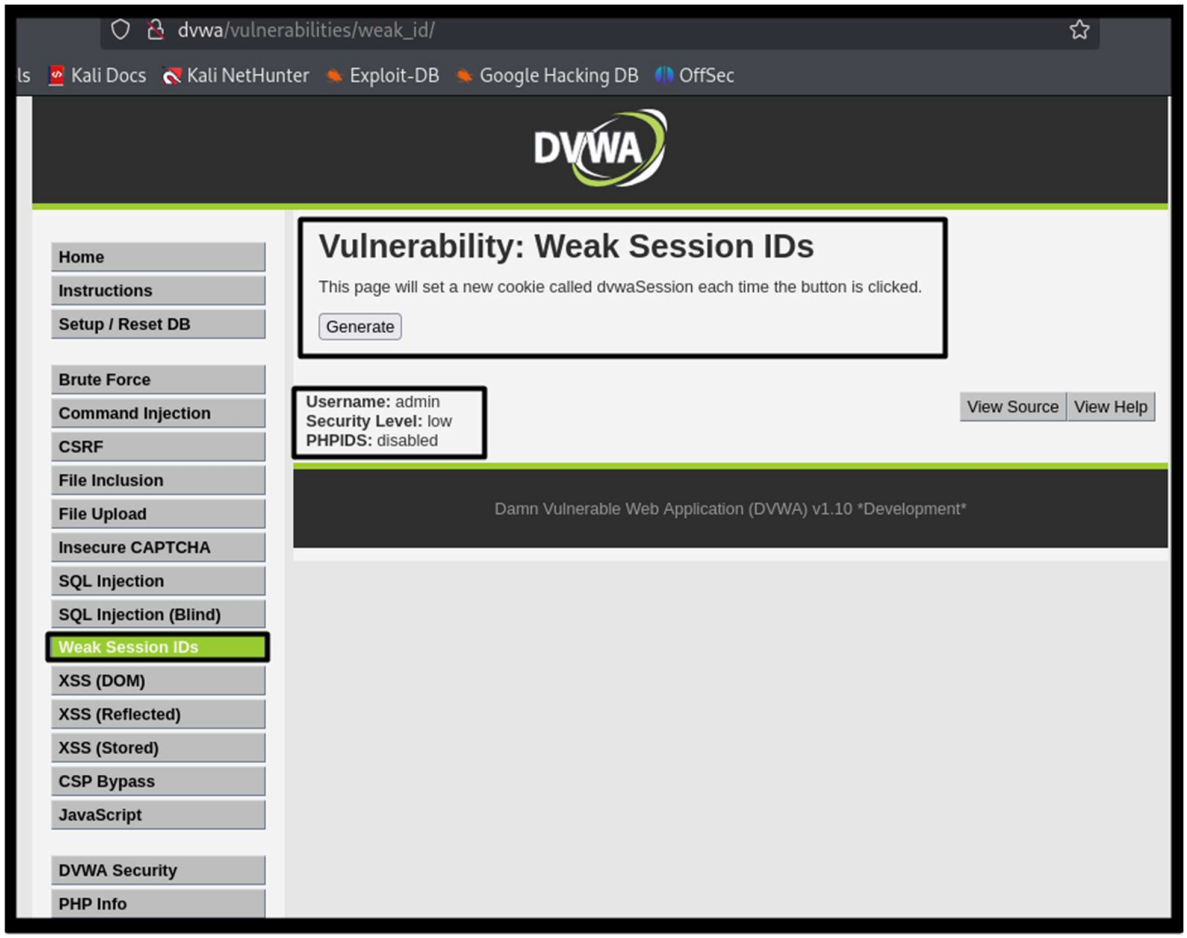
As we Know, we will first view the source code.

Weak Session IDs Source

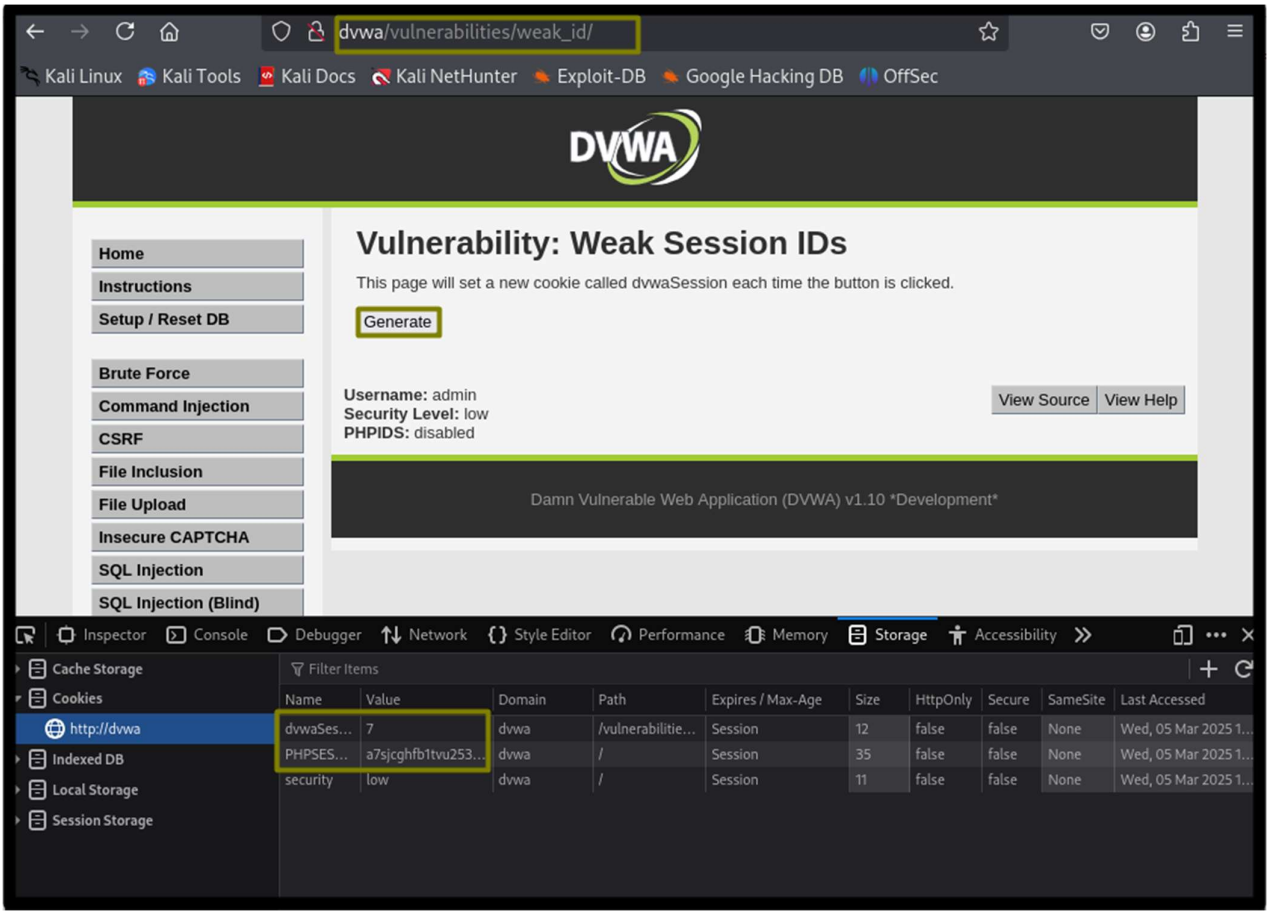
vulnerabilities/weak_id/source/low.php

```
<?php

$html = "";

if ($_SERVER['REQUEST_METHOD'] == "POST") {
    if (!isset ($_SESSION['last_session_id'])) {
        $_SESSION['last_session_id'] = 0;
    }
    $_SESSION['last_session_id']++;
    $cookie_value = $_SESSION['last_session_id'];
    setcookie("dvwaSession", $cookie_value);
}
?>
```

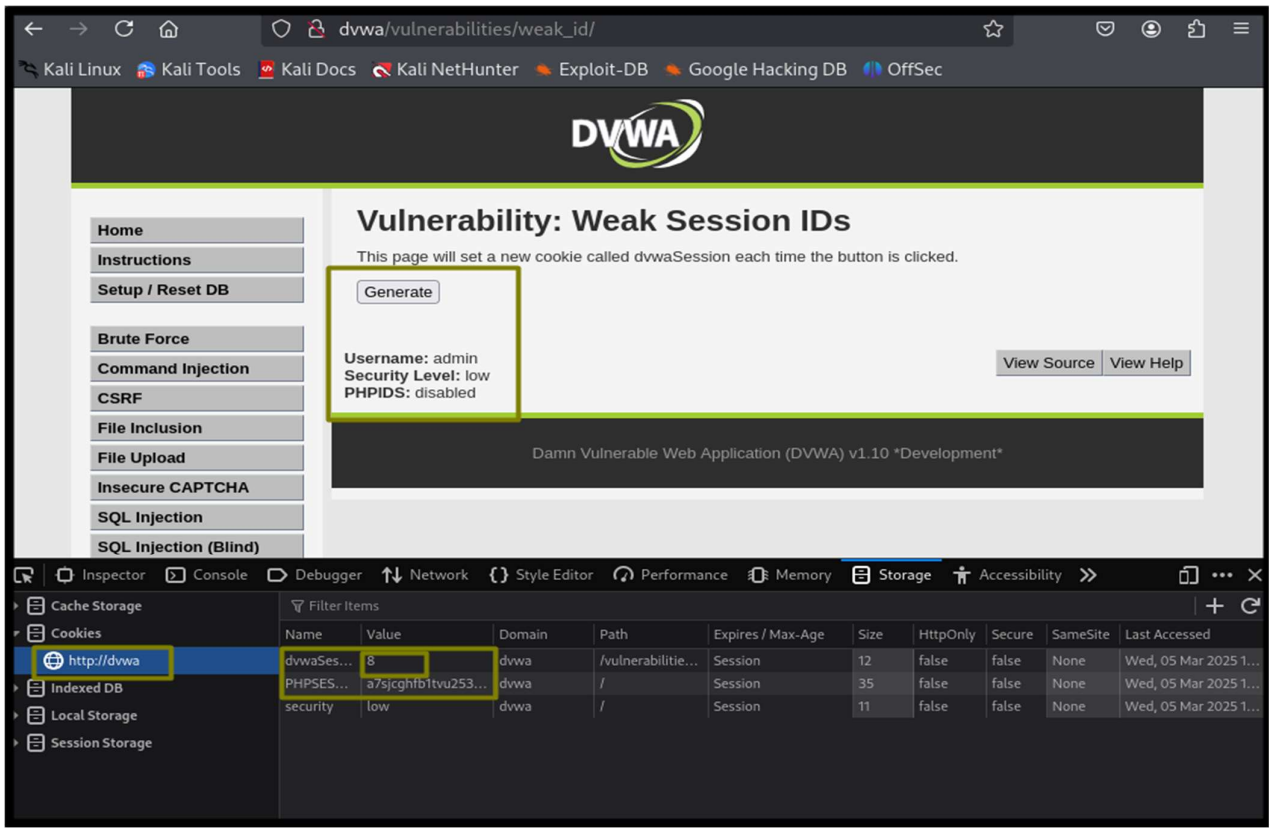
Step: -2 log in the home page of DVWA then click to the Weak Session ID section.



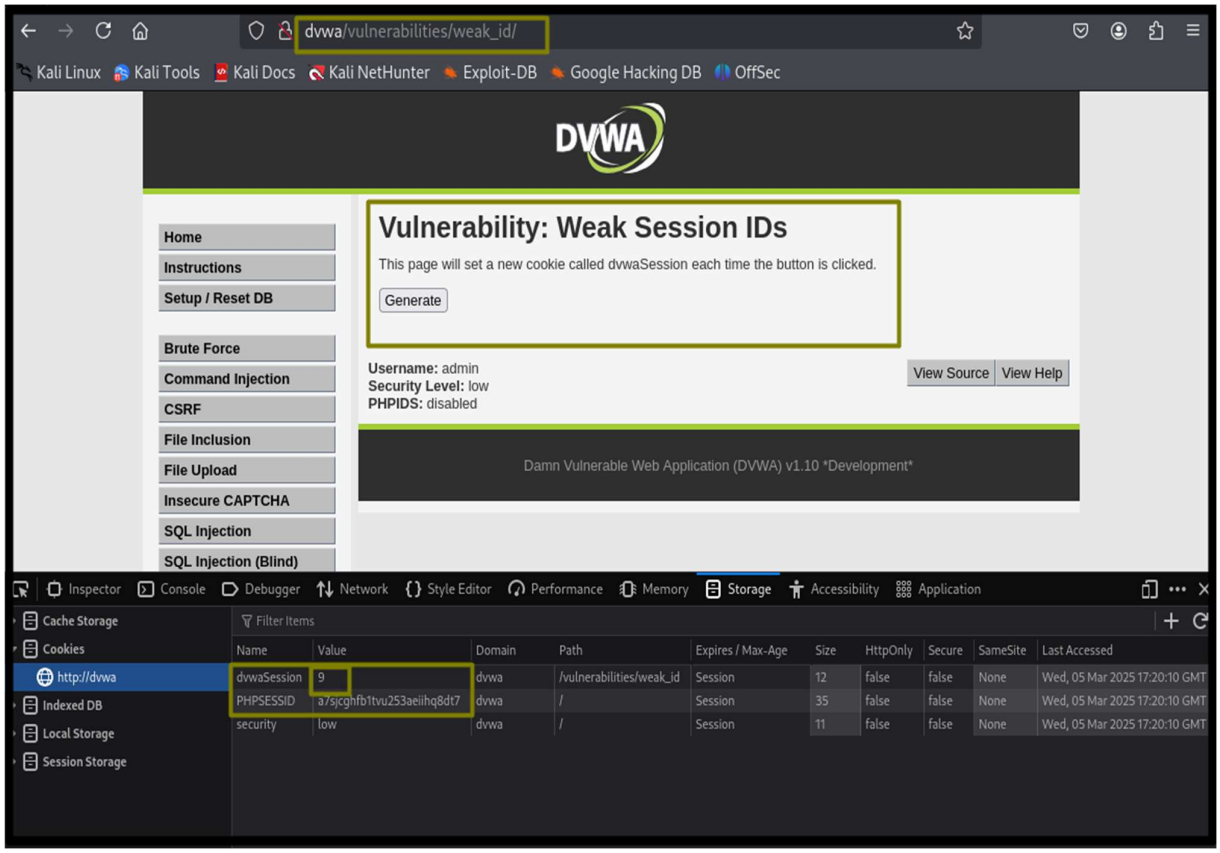
Step: -3 If we click generate we can see the dvwaSession cookie's value via Inspect > Storage > Cookies:



Step: -4 Each time we click Generate this value increments by 1, so after clicking 5 times.



Step: -5 Each time we click Generate this value increments by 1, so after clicking 5 times.



SECURITY LEVEL (MEDIUM)

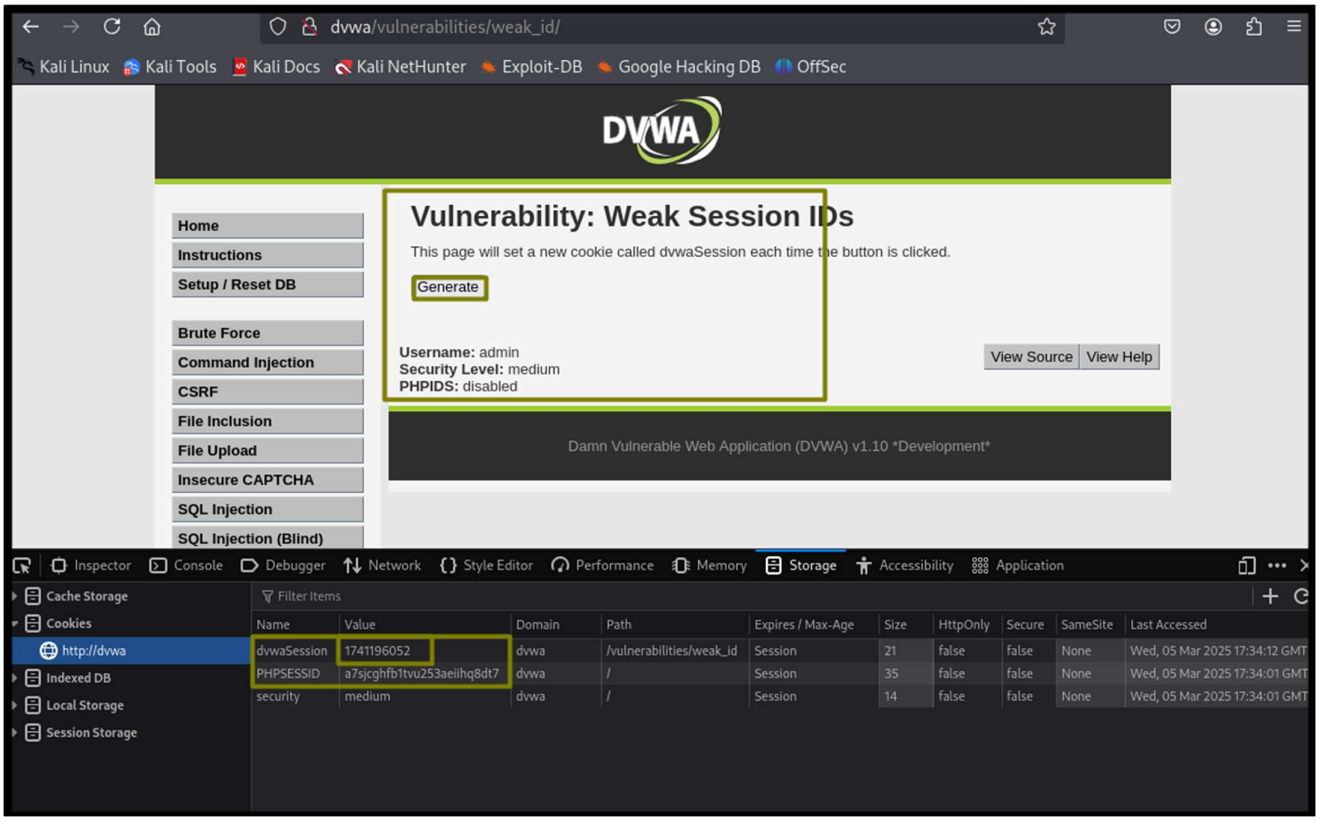
As We know we can view the source code.

Weak Session IDs Source

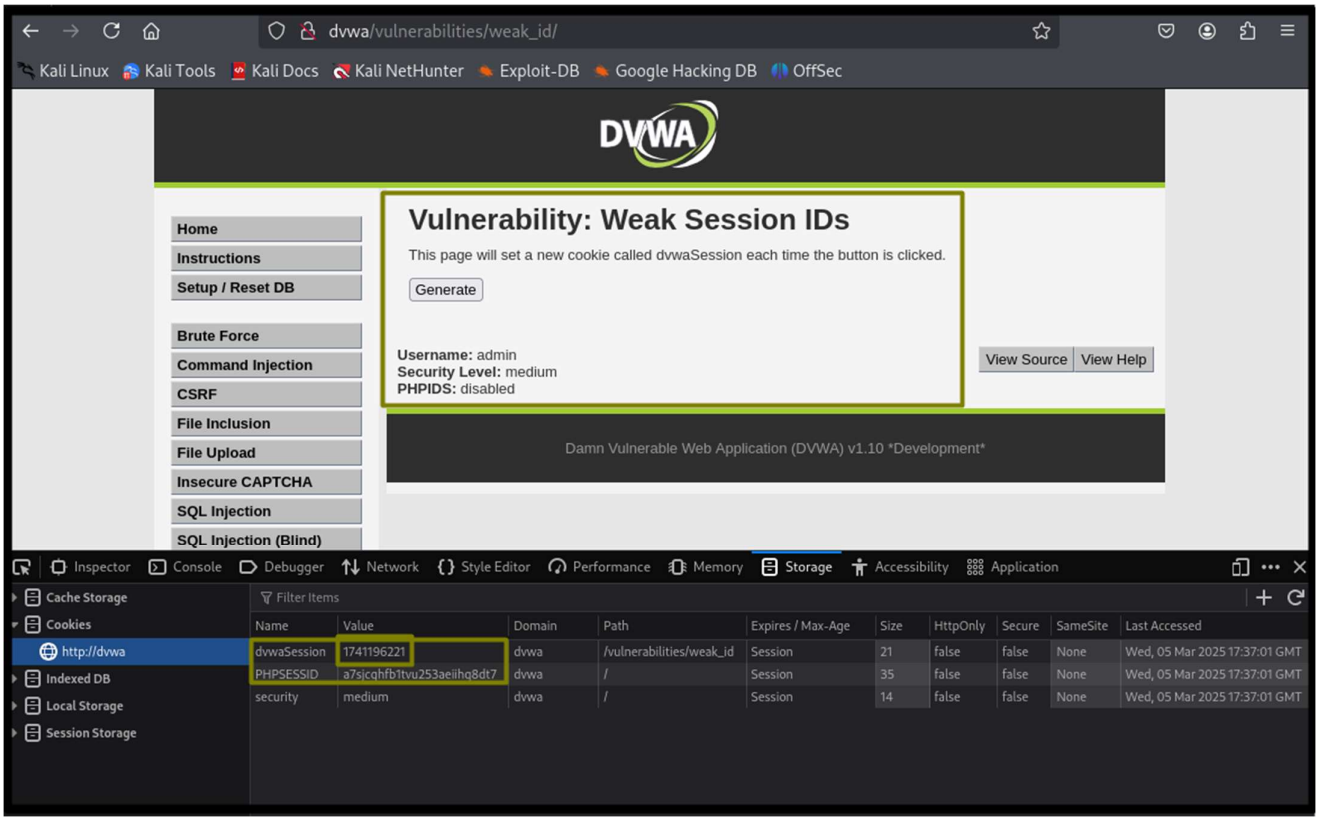
vulnerabilities/weak_id/source/medium.php

```php
<?php

$html = "";

if ($_SERVER['REQUEST_METHOD'] == "POST") {
    $cookie_value = time();
    setcookie("dvwaSession", $cookie_value);
}
?>
```

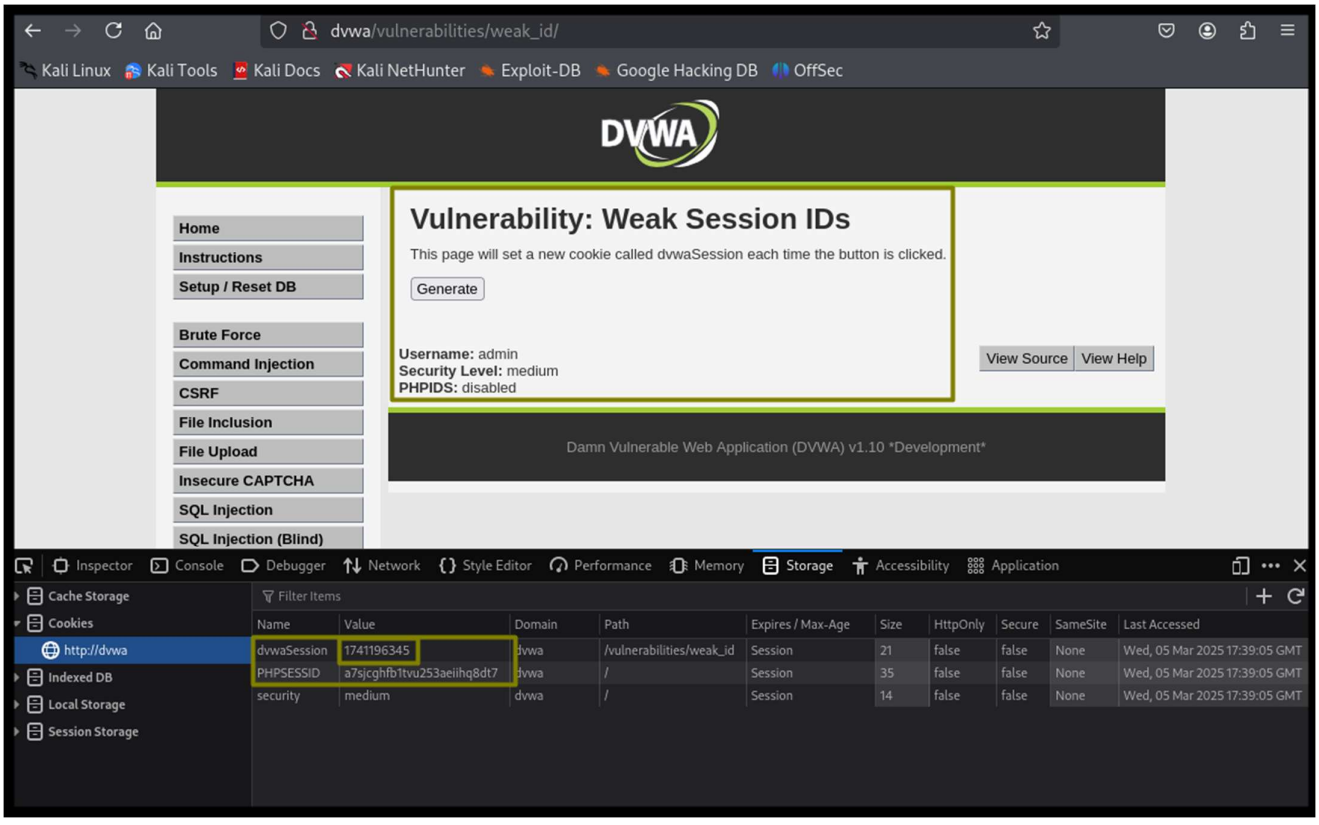
Step: -1 If we click Generate now the cookie looks more complicated.



Step: - 2 If we click a second time:



Step: -3 So the first cookie has the value of 1741196052 and the second one the value of 1741196221; only the last 3 digits changed. If we click a third time similar changes occur:



SECURITY LEVEL (HIGH)

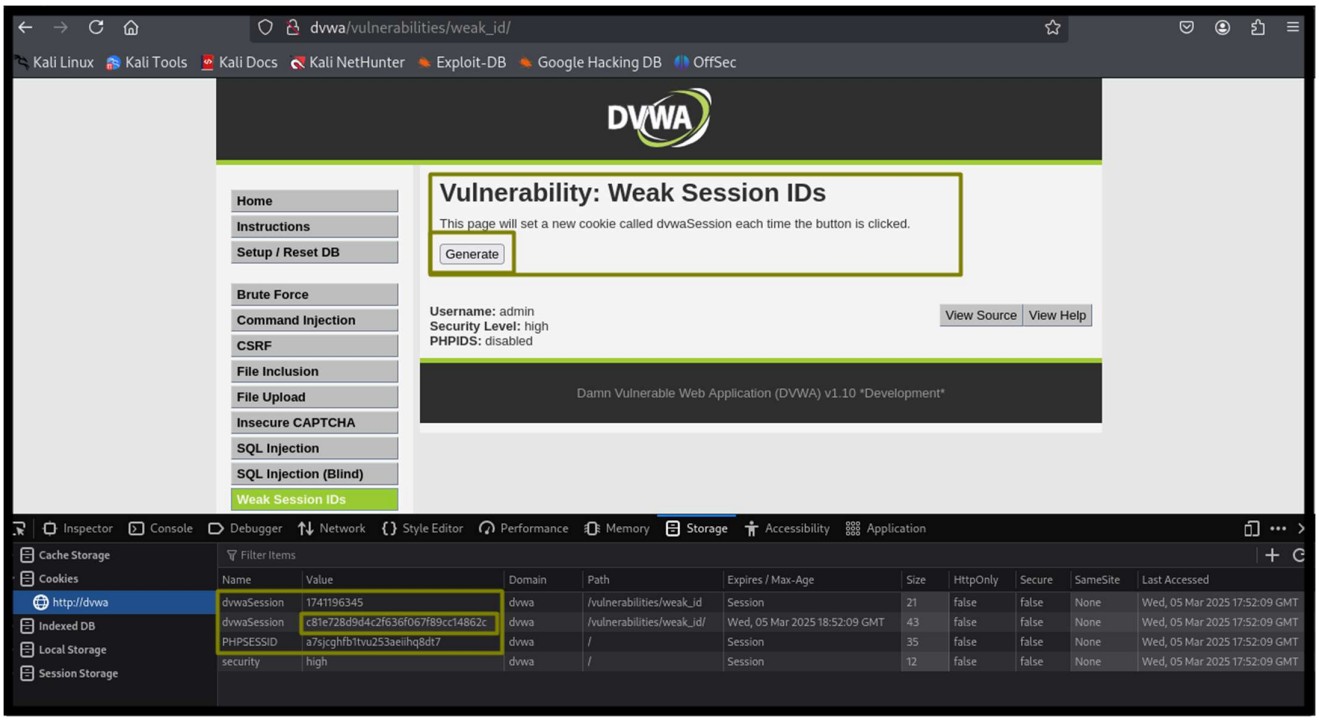
As we know we can see the source code.

Weak Session IDs Source

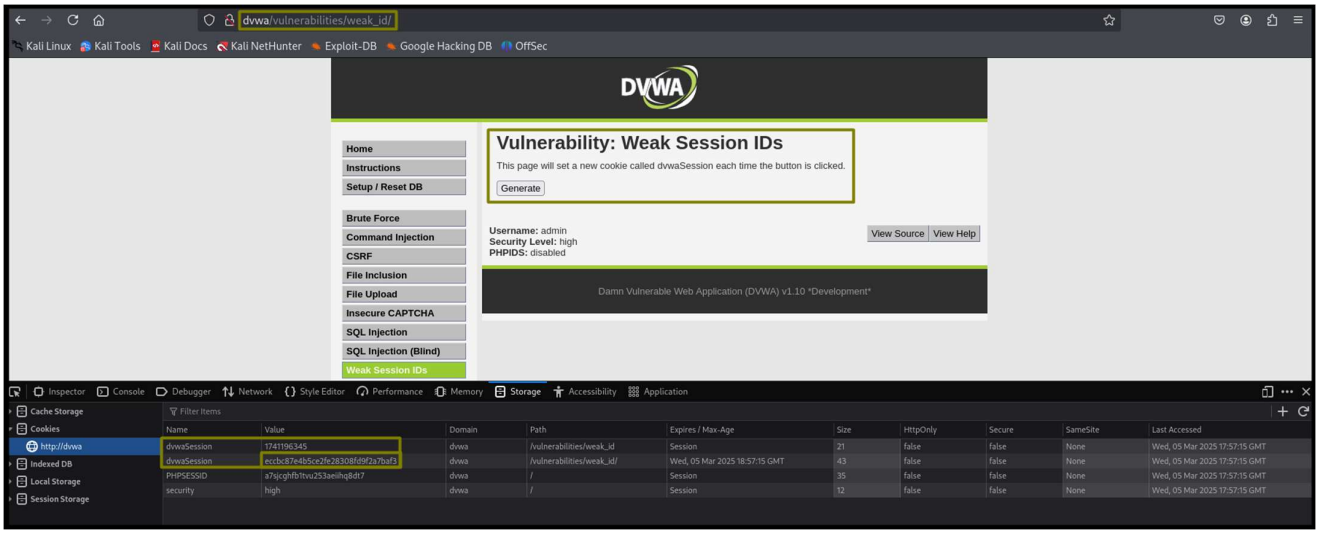
vulnerabilities/weak_id/source/high.php

```
<?php

$html = "";

if ($_SERVER['REQUEST_METHOD'] == "POST") {
    if (!isset ($_SESSION['last_session_id_high'])) {
        $_SESSION['last_session_id_high'] = 0;
    }
    $_SESSION['last_session_id_high']++;
    $cookie_value = md5($_SESSION['last_session_id_high']);
    setcookie("dvwaSession", $cookie_value, time()+3600, "/vulnerabilities/
weak_id/", $_SERVER['HTTP_HOST'], false, false);
}

?>
```

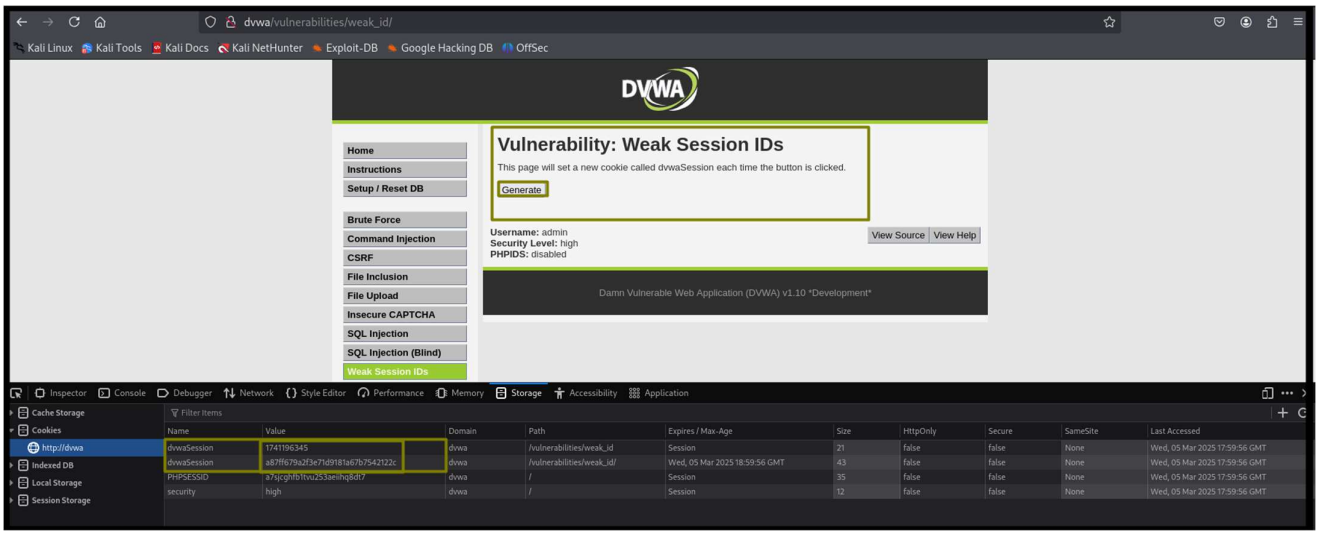
Step: -1 Let's start by generating 2 cookies in a row and see how they look using inspect this time.



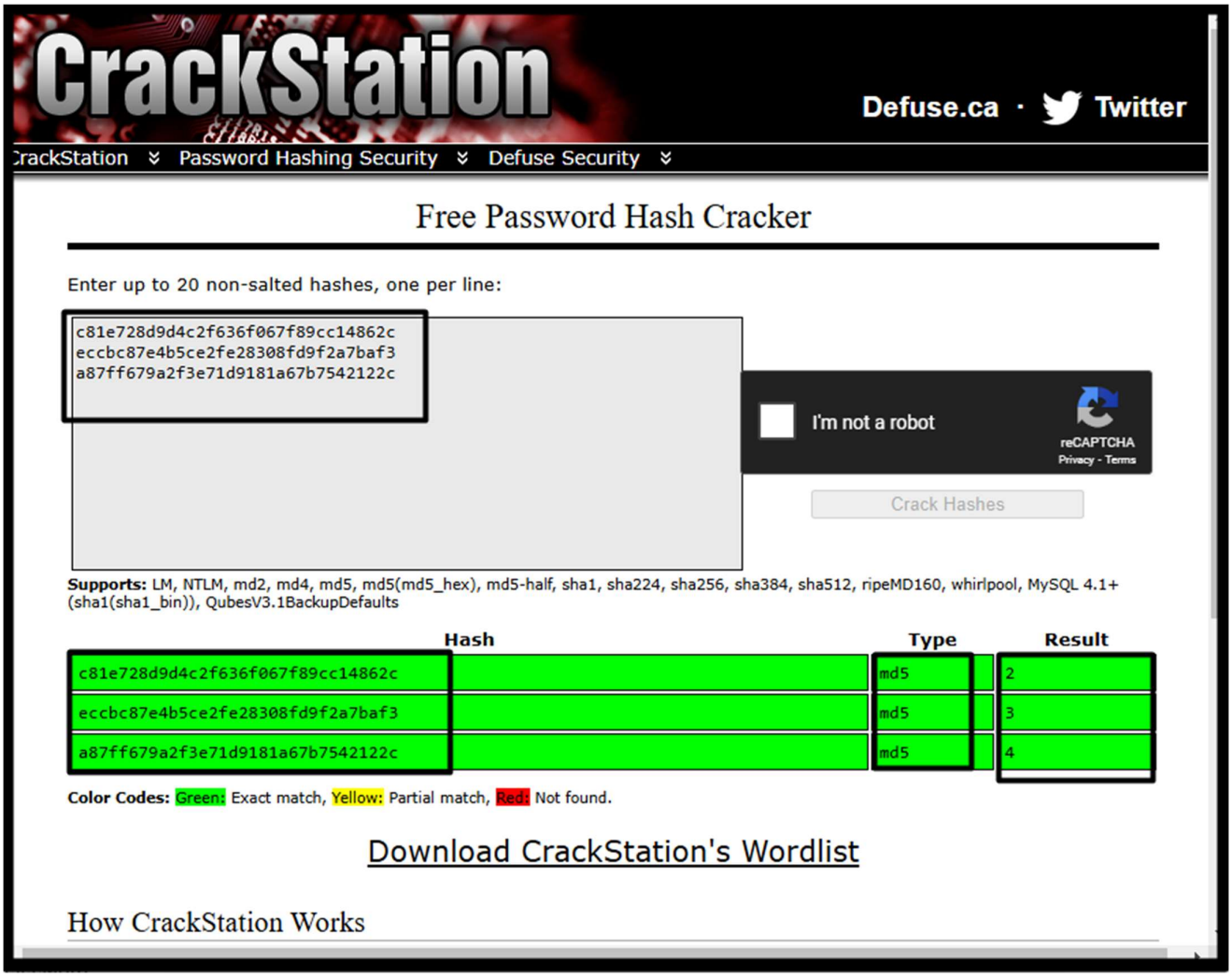
Step: -2 Let's start by generating 2 cookies in a row and see how they look using Inspect this time.



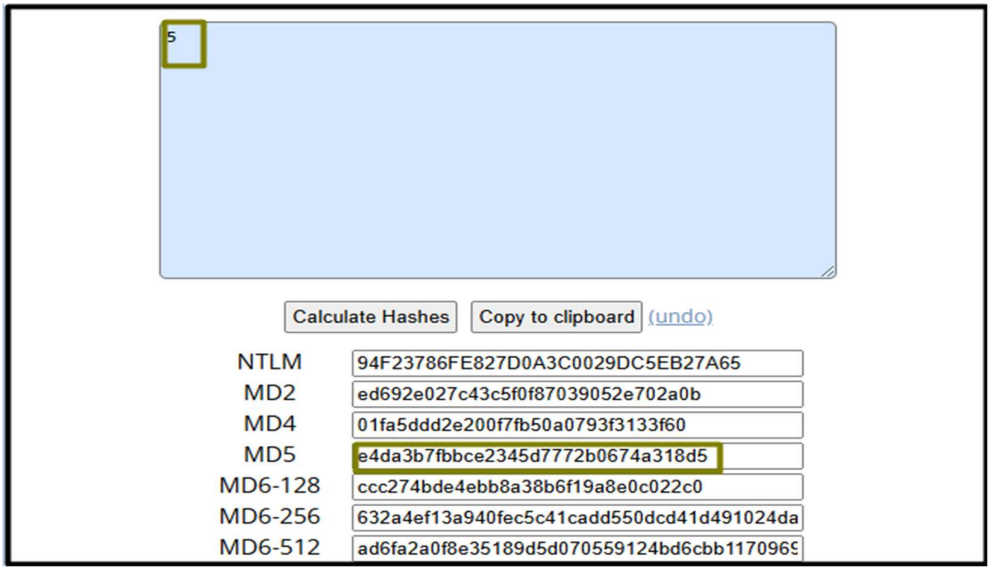
Step: -3 Let's start by generating 2 cookies in a row and see how they look using Inspect this time.



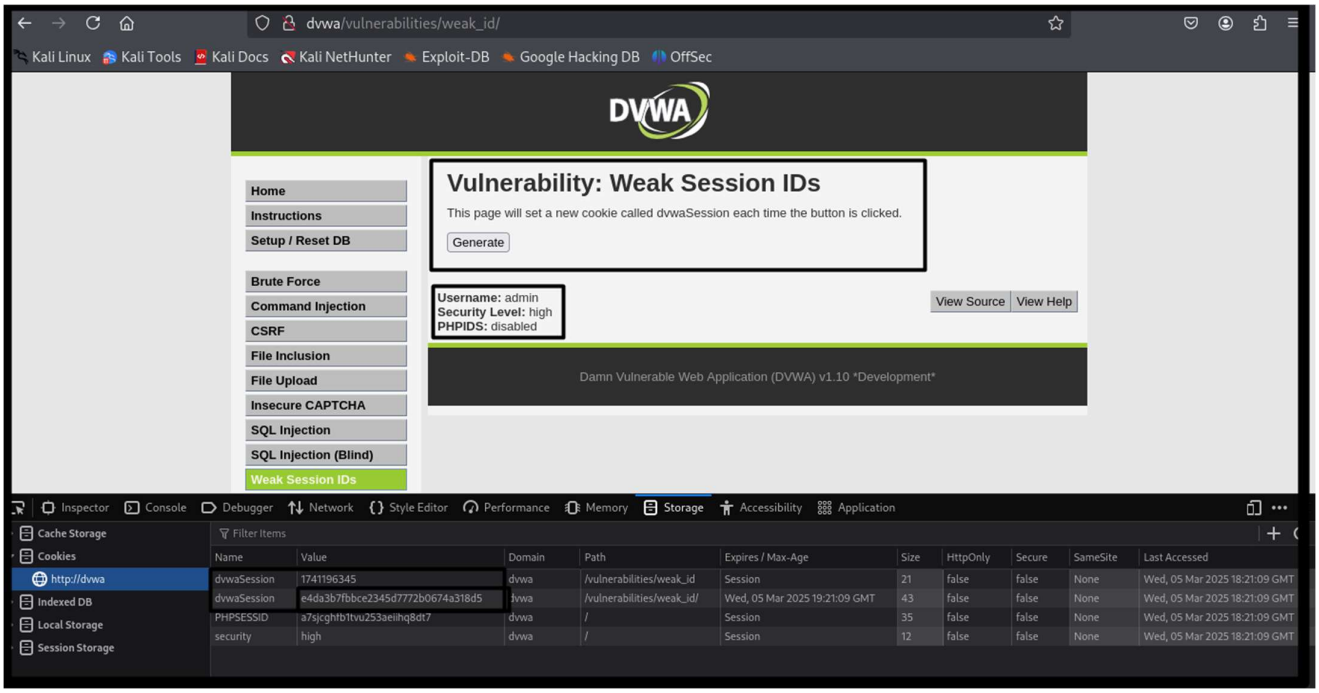
Step: -4 In this Step I am Cracking the all hashes value we can see.



Step: -5 In this step, I am guessing a session ID of **5**, which corresponds to the hash value **e4da3b7fbbce2345d7772b0674a318d5**. Next, I check the following session ID to determine whether the hash values are similar or different.



Step: -6 In this step I am comparing both hashes value are same.



Step: -7 Comparison between hash value

Hash Value	Type	Result
c81e728d9d4c2f636f067f89cc14862c	md5	2
eccbc87e4b5ce2fe28308fd9f2a7baf3	md5	3
a87ff679a2f3e71d9181a67b7542122c	md5	4
e4da3b7fbbce2345d7772b0674a318d5	md5	5