

ASSIGNMENT 6

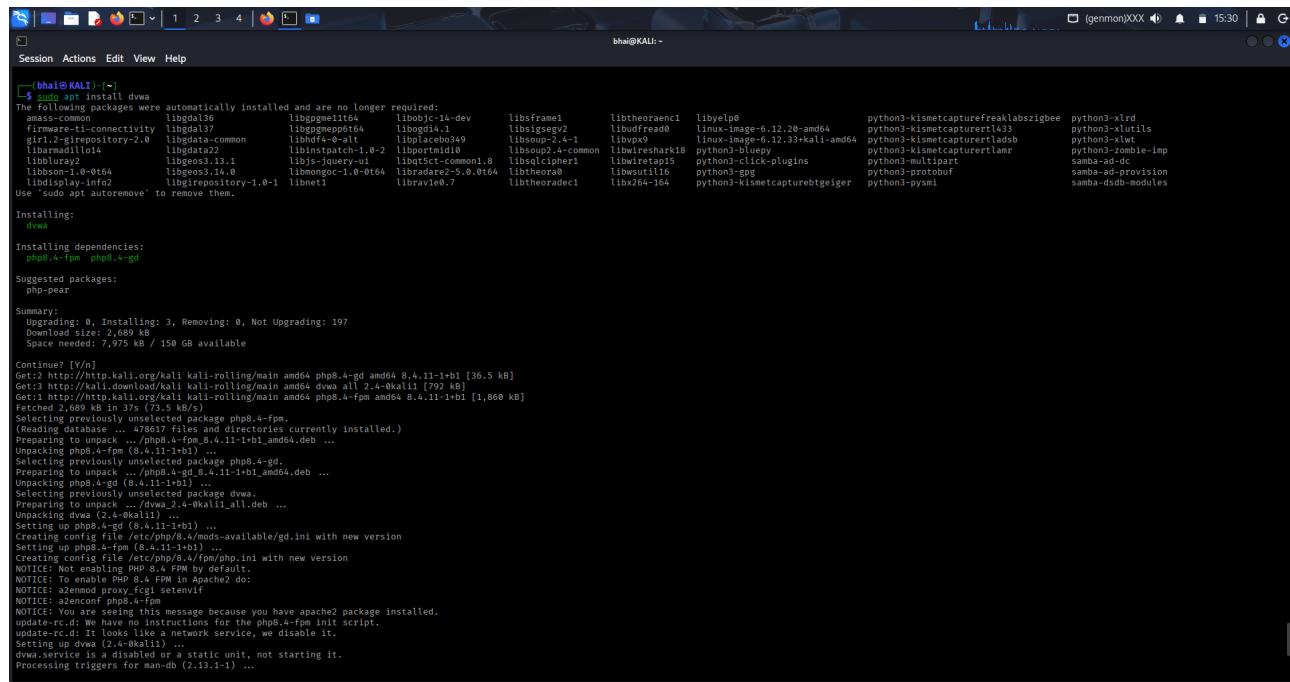
DVWA Installation + Vulnerability Discovery

Step 1 - DVWA Installation

Installed DVWA on Kali using apt and started service.

Commands:

```
sudo apt update
sudo apt install dvwa
sudo dvwa-start
```



A terminal window titled 'bhai@KALI:~' showing the output of the apt command. It lists packages being installed, dependencies, and suggested packages. The summary shows 3 packages installed, 0 removed, and 197 not upgraded. The log ends with the successful start of the dvwa service.

```
bhai@KALI:~$ sudo apt install dvwa
The following packages were automatically installed and are no longer required:
amass-common libgdal36 libgpgme11t64 libobjc-14-dev libssframe1 libtheoraenc1 libvelp0
firmware-ti-connectivity libgdal37 libgpgmeppct64 libobjc4.0 libsigsegv2 libvudread0
firmware-ti-connectivity2.0 libgdal38 libgpgmeppct64 libobjc4.1 libsigsegv2-1 libvudread0
libbamadillo14 libgdal39 libinstpatch-1.0-2 libportmid0 libsoup2.4-common libwreshark18
libblura2 libgdal39.1 libjs-jquery-ui libqt5ct-common1.8 libsqlcipher1 libwretap15
libbson-1.0-0t64 libgdal39.14.0 libmongoc-1.0-0t64 libradare2-5.0.0t64 libtheora0 libwsutil16
libdisplay-info2 libgirepository-1.0-1 libneti libravie0.7 libtheoradec1 libx264-164
Use "sudo apt autoremove" to remove them.

Installing:
dvwa

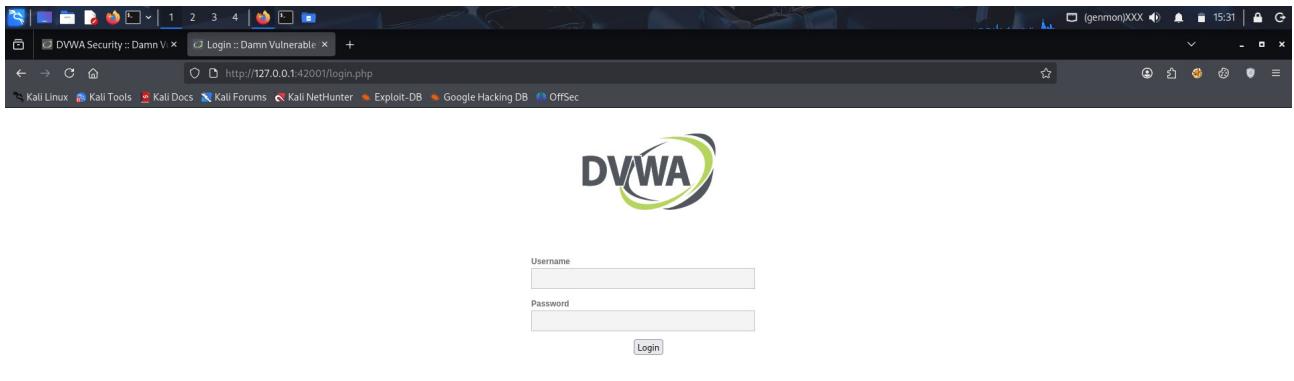
Installing dependencies:
php8.4-fpm php8.4-gd

Suggested packages:
php-pear

Summary:
Upgrading: 0, Installing: 3, Removing: 0, Not Upgrading: 197
Download size: 2,689 kB
Space needed: 7,975 kB / 150 GB available

Continue? [Y/n]
Get:2 http://http.kali.org/kali kali-rolling/main amd64 php8.4-gd amd64 8.4.11-1+b1 [36.5 kB]
Get:3 http://http.kali.org/kali kali-rolling/main amd64 dvwa all 2.4-0kali1 [792 kB]
Get:1 http://http.kali.org/kali kali-rolling/main amd64 php8.4-fpm amd64 8.4.11-1+b1 [1,860 kB]
Fetched 18,718/73,519 kB
Selecting previously unselected package php8.4-fpm.
(Reading database ... 478617 files and directories currently installed.)
Preparing to unpack .../php8.4-fpm_8.4.11-1+b1_amd64.deb ...
Unpacking php8.4-fpm (8.4.11-1+b1) ...
Selecting previously unselected package php8.4-gd.
Preparing to unpack .../php8.4-gd_8.4.11-1+b1_amd64.deb ...
Unpacking php8.4-gd (8.4.11-1+b1) ...
Selecting previously unselected package dvwa.
Preparing to unpack .../dvwa_2.4-0kali1_all.deb ...
Unpacking dvwa (2.4-0kali1)
Setting up php8.4-gd (8.4.11-1+b1) ...
Creating config file /etc/php/8.4/mods-available/gd.ini with new version
Setting up php8.4-fpm (8.4.11-1+b1) ...
Creating config file /etc/php/8.4/fpm/php.ini with new version
NOTICE: Not enabling PHP 8.4 FPM by default.
NOTICE: To enable PHP 8.4 FPM in Apache2 do:
    nano /etc/apache2/conf-available/enable-mod-fpm.conf
NOTICE: a2enconf php8.4-fpm
NOTICE: You are seeing this message because you have apache2 package installed.
update-rc.d: We have no instructions for the php8.4-fpm init script.
update-rc.d: Starting a network service, we disable it.
Setting up dvwa (2.4-0kali1) ...
dvwa.service is a disabled or a static unit, not starting it.
Processing triggers for man-db (2.13.1-1) ...


```



The screenshot shows the DVWA homepage. The left sidebar has a 'DVWA Security' section highlighted. The main content area is titled 'Welcome to Damn Vulnerable Web Application!' and contains sections for 'General Instructions', 'WARNING!', 'Disclaimer', and 'More Training Resources'. A sidebar on the right lists various security vulnerabilities: Brute Force, Command Injection, CSRF, File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection, SQL Injection (Blind), Weak Session IDs, XSS (DOM), XSS (Reflected), XSS (Stored), CSP Bypass, JavaScript, Authorisation Bypass, Open HTTP Redirect, Cryptography, and PHP Info.

Step 2 - Security Levels

Changed DVWA security to Low → Medium → High and tested again.

The screenshot shows the 'Security Level' page of DVWA. The security level is currently set to 'Low'. The page explains that security levels change the vulnerability level of DVWA. It lists four levels: 1. Low (completely vulnerable), 2. Medium (example of bad security practices), 3. High (extension to medium difficulty), and 4. Impossible (highest level). A note states that prior to DVWA v1.9, the level was known as 'high'. Below this is a dropdown menu with 'Low' selected and a 'Submit' button. The left sidebar is identical to the previous screenshot, with the 'DVWA Security' section highlighted.

Step 3 - Vulnerability Tests

1. SQL Injection

Performed basic SQL injection to bypass login and extract DB info.

Payloads:

1' OR '1'='1

The screenshot shows a Firefox browser window with the DVWA Security application open. The URL is `http://127.0.0.1:8001/vulnerabilities/sql/?id=1+OR+'1'%3D'1&Submit=Submit#`. The DVWA logo is at the top. On the left is a sidebar menu with various security vulnerabilities listed. The 'SQL Injection' option is highlighted. The main content area is titled 'Vulnerability: SQL Injection'. It contains a form with a 'User ID:' field containing the payload '1' OR '1'='1'. Below the form, the results of the injection are displayed in a table:

User ID	First name	Surname
ID: 1' OR '1'='1	admin	admin
ID: 1' OR '1'='1	Gordon	Brown
ID: 1' OR '1'='1	Hack	Me
ID: 1' OR '1'='1	Pablo	Picasso
ID: 1' OR '1'='1	Bob	Smith

Below the table, there's a 'More Information' section with links to external resources about SQL injection. At the bottom left, it says 'Username: admin' and 'Security Level: low'. At the bottom right, there are 'View Source' and 'View Help' buttons.

2. Cross-Site Scripting (XSS)

Injected JavaScript to confirm reflected and stored XSS.

Payloads:

<h1>vikram<h1>

The screenshot shows the DVWA Reflected XSS page. The URL is http://127.0.0.1:42001/vulnerabilities/xss_r/?name=<h1>vikram</h1>. The left sidebar has a menu with various attack types, and 'XSS (Reflected)' is selected. The main content area has a form with the placeholder 'What's your name? <h1>vikram</h1>' and a 'Submit' button. Below the form, the text 'Hello Vikram' is displayed in red. A 'More Information' section lists several links related to XSS attacks.

```
<script>alert(1)</script>
```

The screenshot shows a browser window with the same URL as the previous screenshot. A modal dialog box is centered on the screen with the text '127.0.0.1:42001' at the top, followed by the number '1' and an 'OK' button at the bottom. This indicates that the reflected XSS payload was successfully executed and alerted the user.

3. Command Injection

Executed system commands via vulnerable parameter.

Payloads:

127.0.0.1; ls

The screenshot shows a browser window for DVWA (Damn Vulnerable Web Application) version 1.0. The URL is http://127.0.0.1:42001/vulnerabilities/exec/. The main content area displays the output of a ping command to 127.0.0.1, showing four ICMP packets sent and received with times ranging from 0.033 ms to 0.050 ms. Below the command input field, there is a "More Information" section with links to various resources about command injection.

4. File Upload Vulnerability

Uploaded test PHP file by bypassing file extension filters.

Payload:

```
<?php echo "test"; ?>
```

Rename as: shell.php.jpg

The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. On the left is a sidebar with various exploit categories: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, CSRF, File Inclusion, **File Upload**, Insecure CAPTCHA, SQL Injection, SQL Injection (Blind), Weak Session IDs, XSS (DOM), XSS (Reflected), XSS (Stored), CSP Bypass, JavaScript, Authorisation Bypass, Open HTTP Redirect, Cryptography, DVWA Security, PHP Info, About, and Logout. The 'File Upload' option is currently selected. The main content area is titled 'Vulnerability: File Upload' and contains a form with a 'Browse...' button and an 'Upload' button. Below the form, a message says 'Your image was not uploaded.' To the right of the main content, a smaller window shows the source code of the upload script:

```
<?php
if( isset( $_POST[ 'Upload' ] ) ) {
    // Where are we going to be writing to?
    $target_path = DVWA WEB PAGE TO ROOT . 'hackable/uploads/';
    $target_path .= basename( $_FILES[ 'uploaded' ][ 'name' ] );
    // Can we move the file to the upload folder?
    if( !move_uploaded_file( $_FILES[ 'uploaded' ][ 'tmp_name' ], $target_path ) ) {
        // No:
        echo '<p>Your image was not uploaded.</p>';
    }
    else {
        // Yes!
        echo "<p>$target_path successfully uploaded!</p>";
    }
}

```

At the bottom of the main content area, it says 'Username: admin' and 'Security Level: low'.

5. IDOR (Insecure Direct Object Reference)

Accessed other user info by modifying ID value.(performed on THM)

URL Test:

?id=10

The screenshot shows a browser developer tools Network tab with several requests listed. One request has a status of 'Success' and a response type of 'JSON'. The response body is displayed as follows:

```
user_id: 10
username: "niels"
email: "niels@webmail.thm"
firstname: "Niels"
lastname: "Tester"
id_number: "NIELS-001"
address1: "42 chill Street"
address2: "Apt 1"
city: "TryTown"
state: "THM"
postal_code: "42424"
country: "Netherlands"
children: (5)[{...}, {...}, {...}, {...}, {...}]
```

6. Security Misconfigurations / Headers

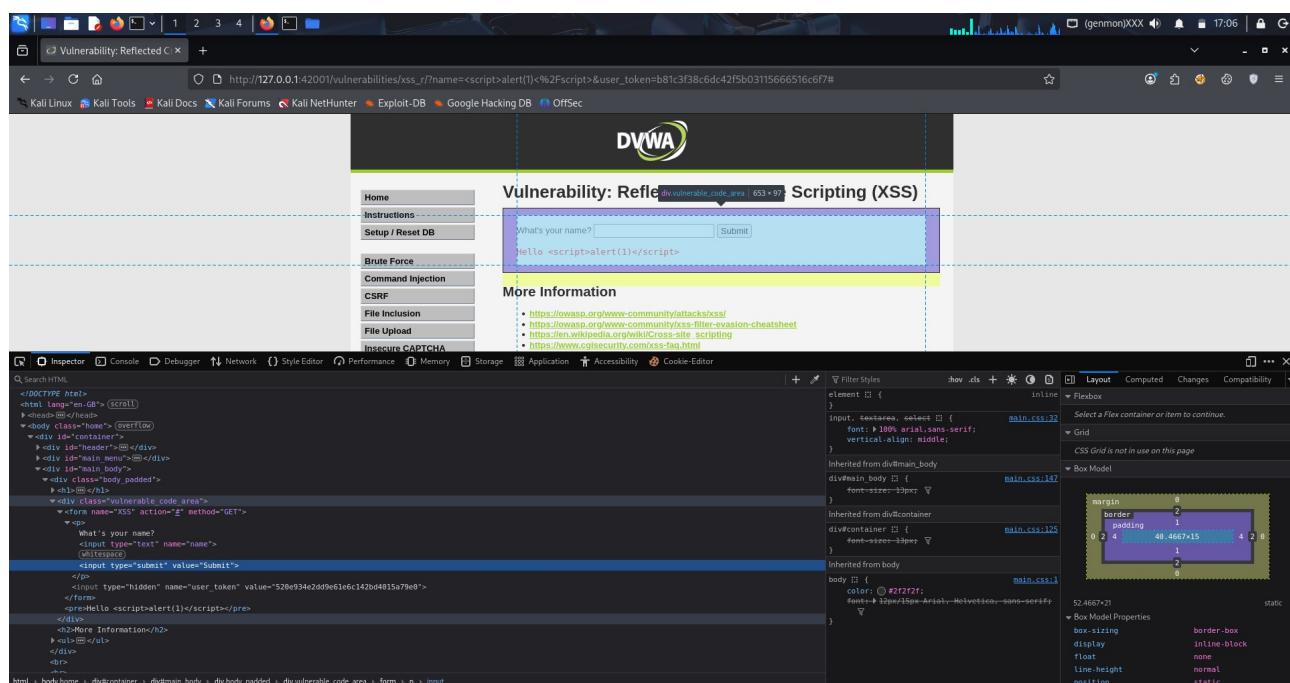
Checked missing security headers using curl.

Command:

```
curl -I http://localhost/dvwa
```

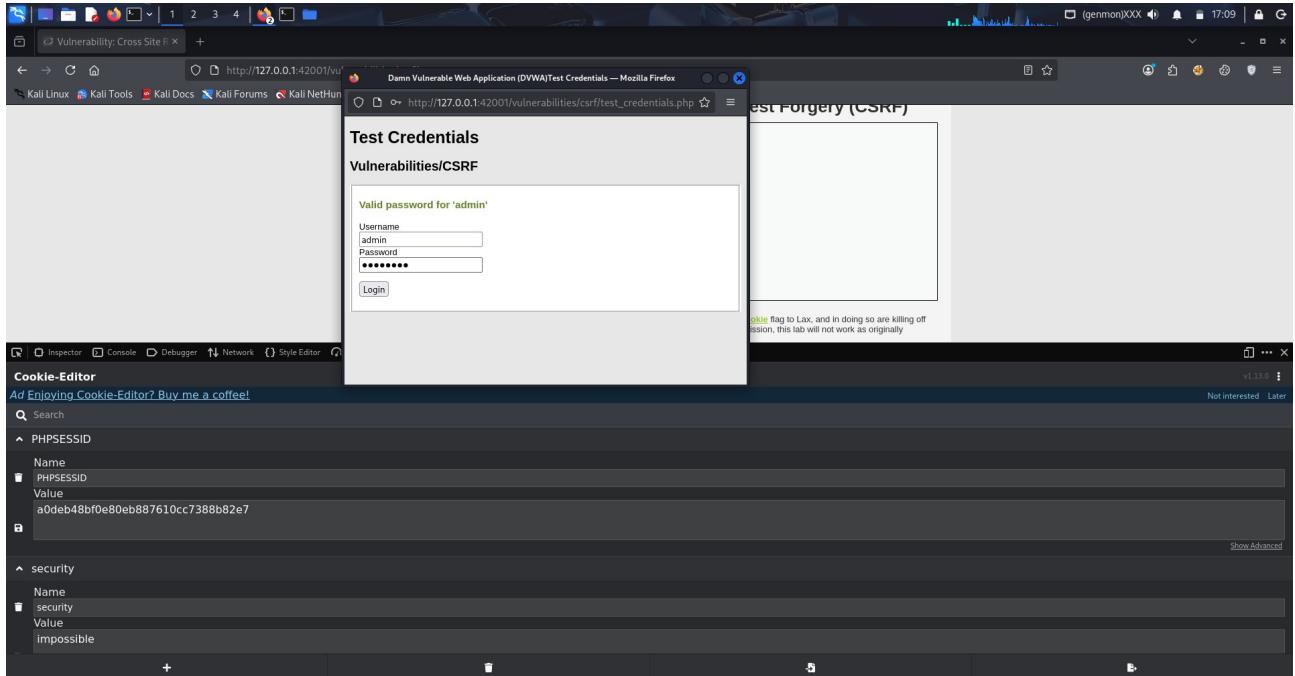
7. Sensitive Info in Source / JS

Viewed page source to identify exposed info.



8. Session Management Issues

Verified insecure cookies and missing flags.



Step 4 - Light Automation

Nikto Scan

Scanned server for misconfigurations.

Command:

```
nikto -h http://localhost/dvwa
```

```
(bhai㉿KALI)-[~]
$ nikto -h http://127.0.0.1:42001/
- Nikto v2.5.0

+ Target IP:      127.0.0.1
+ Target Hostname: 127.0.0.1
+ Target Port:    42001
+ Start Time:    2025-12-11 17:12:22 (GMT5.5)

+ Server: nginx/1.28.0
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type.
+ Content-Type header/encoding may be forged
+ Root page / redirects to: login.php
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ /login.php: Admin login page/section found.
+ 8074 requests: 0 error(s) and 3 item(s) reported on remote host
+ End Time:       2025-12-11 17:12:59 (GMT5.5) (37 seconds)

+ 1 host(s) tested

(bhai㉿KALI)-[~]
$
```

WhatWeb Fingerprinting

Identified technologies running on DVWA.

Command:

```
whatweb http://localhost/dvwa
```

```
(bhai㉿KALI)-[~]
$ whatweb http://127.0.0.1:42001/
http://127.0.0.1:42001/ [302 Found] Cookies[PHPSESSID,security], Country[RESERVED][zz], HTTPServer[nginx/1.28.0], HttpOnly[PHPSESSID,security], IP[127.0.0.1]
, RedirectLocation[login.php], nginx[1.28.0]
http://127.0.0.1:42001/login.php [200 OK] Country[RESERVED][zz], DVWA, HTML5, HTTPServer[nginx/1.28.0], IP[127.0.0.1], PHP, PasswordField[password], Title[Login :: Damn Vulnerable Web Application (DVWA)], nginx[1.28.0]

(bhai㉿KALI)-[~]
$
```

DIRB Directory Scan

Enumerated hidden directories and files.

Command:

```
dirb http://localhost/dvwa
```

The screenshot shows a terminal window on a Kali Linux desktop environment. The terminal title is 'bhei@KALI: ~'. The command entered is 'dirb http://127.0.0.1:42001/'. The output of the scan is displayed below:

```
DIRB v2.22
By The Dark Raver

START_TIME: Thu Dec 11 17:16:51 2025
URL_BASE: http://127.0.0.1:42001/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

GENERATED WORDS: 4612

— Scanning URL: http://127.0.0.1:42001/ —
⇒ DIRECTORY: http://127.0.0.1:42001/config/
⇒ DIRECTORY: http://127.0.0.1:42001/database/
⇒ DIRECTORY: http://127.0.0.1:42001/docs/
⇒ DIRECTORY: http://127.0.0.1:42001/external/
+ http://127.0.0.1:42001/favicon.ico (CODE:200|SIZE:1406)
+ http://127.0.0.1:42001/index.php (CODE:302|SIZE:0)
+ http://127.0.0.1:42001/php.ini (CODE:200|SIZE:154)
+ http://127.0.0.1:42001/phpinfo.php (CODE:302|SIZE:0)
+ http://127.0.0.1:42001/robots.txt (CODE:200|SIZE:25)

— Entering directory: http://127.0.0.1:42001/config/ —
— Entering directory: http://127.0.0.1:42001/database/ —
— Entering directory: http://127.0.0.1:42001/docs/ —
+ http://127.0.0.1:42001/docs/copyright (CODE:200|SIZE:1085)
⇒ DIRECTORY: http://127.0.0.1:42001/docs/graphics/

— Entering directory: http://127.0.0.1:42001/external/ —
```

Nuclei Scan

Ran templates for common misconfigurations.

Command:

```
nuclei -u http://localhost/dvwa
```

```
bhai@KALI:~$ nuclei -u http://127.0.0.1:42001/
   _/\_ \_/\_ / \_/\_ \_/\_
  / \_/\_ \_/\_ \_/\_ \_/\_ \
 / \_/\_ \_/\_ \_/\_ \_/\_ \
v3.4.10
projectdiscovery.io

[INF] Your current nuclei-templates v10.3.4 are outdated. Latest is v10.3.5
[WRN] Found 1 templates with syntax error (use -validate flag for further examination)
[INF] Current nuclei version: v3.4.10 (outdated)
[INF] Current nuclei-templates version: v10.3.4 (outdated)
[INF] New templates added in latest release: 0
[INF] Templates loaded for current scan: 8855
[INF] Executing 8853 signed templates from projectdiscovery/nuclei-templates
[WRN] Loading 2 unsigned templates for scan. Use with caution.
[INF] Targets loaded for current scan: 1
[INF] Templates clustered: 1853 (Reduced 1740 Requests)
[INF] Using Interactsh Server: oast.live
[dwba-default-login] [http] [critical] http://127.0.0.1:42001/index.php [password="password",username="admin"]
[cookies-without-secure] [javascript] [info] 127.0.0.1:42001 ["security","PHPSESSID"]
[external-service-interaction] [http] [info] http://127.0.0.1:42001/
[external-service-interaction] [http] [info] http://127.0.0.1:42001/
[waf-detect:nginxgeneric] [http] [info] http://127.0.0.1:42001/
[INF] Scan completed in 1m. 5 matches found.

(bhai@KALI)-[~]
$
```