

Problem Statement-2

Deploy a local k8s cluster (using minikube, k3s, or anything else) and deploy the DVWA application. Showcase/demo 3 attack surfaces as mentioned in its documentation.

1. Sample k8s setup

Deployment File

```
-----
apiVersion: apps/v1
kind: Deployment
metadata:
  name: dvwa
spec:
  replicas: 1
  selector:
    matchLabels:
      app: dvwa
  template:
    metadata:
      labels:
        app: dvwa
    spec:
      containers:
        - name: dvwa
          image: vulnerables/web-dvwa
          ports:
            - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: dvwa
spec:
  selector:
    app: dvwa
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
-----
```

The is a Kubernetes deployment and service configuration for deploying the DVWA (Damn Vulnerable Web Application) application. DVWA is intentionally designed to have multiple vulnerabilities for practicing and understanding web application security.

The deployment file deploys a single replica of the DVWA application using the vulnerables/web-dvwa Docker image. The application listens on port 80. The service definition creates a Kubernetes service named "dvwa" to expose the DVWA deployment internally within the cluster on port 80.

```
one@ghost77: ~/Desktop/docker-dvwa
one@ghost77: ~/Desktop/docker-dvwa$ minikube start
minikube v1.33.0 on Debian bookworm/sid
Using the docker driver based on existing profile
Starting "minikube" primary control-plane node in "minikube" cluster
Pulling base image v0.0.43 ...
Restarting existing docker container for "minikube" ...
Preparing Kubernetes v1.30.0 on Docker 26.0.1 ...
Verifying Kubernetes components...
  Using image gcr.io/k8s-minikube/storage-provisioner:v5
  Using image quay.io/metallb/speaker:v0.9.6
  Using image quay.io/metallb/controller:v0.9.6
  Using image docker.io/kubernetesui/metrics-scraper:v1.0.8
  Using image docker.io/kubernetesui/dashboard:v2.7.0
Some dashboard features require the metrics-server addon. To enable all features please run:

    minikube addons enable metrics-server

Enabled addons: default-storageclass, metallb, storage-provisioner, dashboard
Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
one@ghost77: ~/Desktop/docker-dvwa$ kubectl apply -f dvwa-deployment.yaml
deployment.apps/dvwa unchanged
service/dvwa unchanged
one@ghost77: ~/Desktop/docker-dvwa$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
dvwa-6db8c479d4-cb46z             1/1     Running   1 (5d19h ago)   5d20h
one@ghost77: ~/Desktop/docker-dvwa$ kubectl port-forward svc/dvwa 8080:80
Forwarding from 127.0.0.1:8080 -> 80
Forwarding from [::1]:8080 -> 80
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
Handling connection for 8080
```

To showcase three attack surfaces mentioned in the DVWA documentation:

- 1.SQL Injection: Exploit the vulnerable login page by entering malicious SQL code in the username or password field to manipulate the database or retrieve sensitive information.
- 2.Cross-Site Scripting (XSS): Inject JavaScript code in the user input field vulnerable to XSS attacks to execute malicious scripts on other users' browsers.
- 3.Command Injection: Exploit vulnerabilities in the DVWA application to execute arbitrary commands on the underlying system by injecting malicious commands in user inputs.

The above surface attacks 1.SQL Injection 2.Cross-Site Scripting 3.Command Injection are demonstrated with images and explanation.

SQL Injection:

- The input ' OR '1'='1 is a common example of a SQL injection payload.
- By entering this payload in the username or password field, the attacker is attempting to manipulate the SQL query used for authentication.
- The condition '1'='1' always evaluates to true, effectively bypassing the login mechanism.
- As a result, the attacker gains unauthorized access to the application, bypassing the need for valid credentials.

The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The browser address bar displays the URL: `localhost:8080/vulnerabilities/sqli/?id='+OR+'1'%3D'1&Submit=Submit#`. The page title is "Vulnerability: SQL Injection". On the left, there is a sidebar with navigation links: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, CSRF, File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection (highlighted), SQL Injection (Blind), Weak Session IDs, XSS (DOM), XSS (Reflected), XSS (Stored), CSP Bypass, JavaScript, DVWA Security, PHP Info, About, and Logout. The main content area shows the "User ID:" field with the payload `' OR '1'='1` entered, and a "Submit" button. Below the input field, the application displays the results of the query, showing user details for the injected payload. The results are as follows:

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

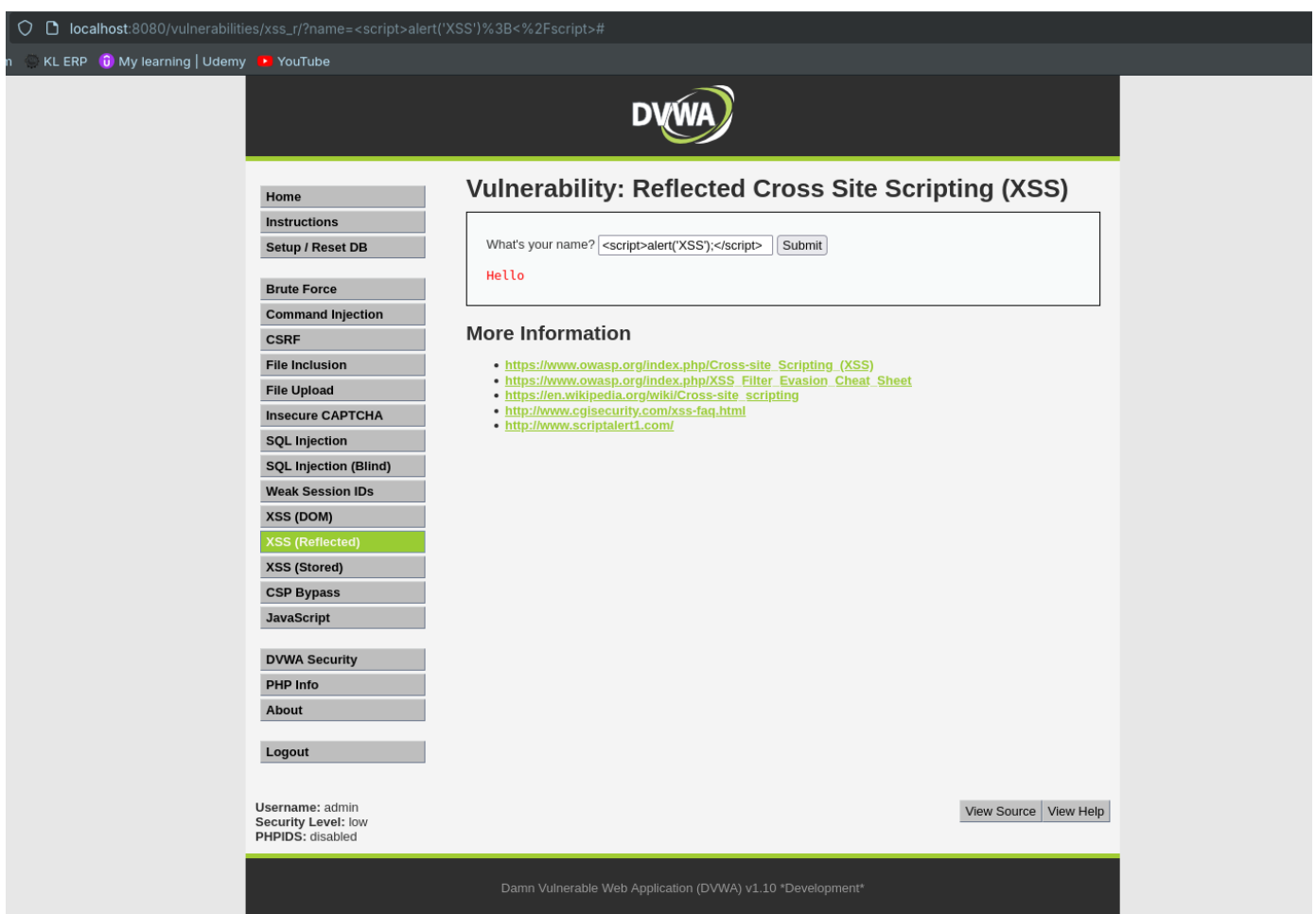
Below the results, there is a "More Information" section with a list of links to external resources:

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

At the bottom of the page, there is a footer that reads: "Damn Vulnerable Web Application (DVWA) v1.10 *Development*".

Cross-Site Scripting (XSS)

- The payload `<script>alert('XSS');</script>` is an example of an XSS payload.
- By injecting this payload into an input field, the attacker aims to execute arbitrary JavaScript code on other users' browsers.
- When a victim user views the compromised content containing the injected payload, the JavaScript code `alert('XSS');` executes, displaying an alert box with the message "XSS".
- This showcases the ability to run unauthorized scripts on the victim's browser, which could be used for various malicious purposes.



Command Injection

- The payload `127.0.0.1; ls -la` is an example of a command injection payload.
- By injecting this payload into an input field vulnerable to command injection, the attacker attempts to execute arbitrary commands on the underlying system.
- In this example, the injected command is `ls -la`, which lists the files and directories in the current directory on the target system.
- If the command injection vulnerability is present and not properly mitigated, the attacker can execute unauthorized commands and potentially gain control over the system.

localhost:8080/vulnerabilities/exec/#

KL ERP My learning | Udemey YouTube

DVWA

Vulnerability: Command Injection

Ping a device

Enter an IP address:

```
PING 127.0.0.1 (127.0.0.1): 56 data bytes
64 bytes from 127.0.0.1: icmp_seq=0 ttl=64 time=0.032 ms
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.033 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.023 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.040 ms
--- 127.0.0.1 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.023/0.032/0.040/0.000 ms
total 20
drwxr-xr-x 1 www-data www-data 4096 Oct 12 2018 .
drwxr-xr-x 1 www-data www-data 4096 Oct 12 2018 ..
drwxr-xr-x 1 www-data www-data 4096 Oct 12 2018 help
-rw-r--r-- 1 www-data www-data 1830 Oct 12 2018 index.php
drwxr-xr-x 1 www-data www-data 4096 Oct 12 2018 source
```

More Information

- <http://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution>
- <http://www.ss64.com/bash/>
- <http://www.ss64.com/nt/>
- https://www.owasp.org/index.php/Command_Injection

View Source View Help

Username: admin
Security Level: low
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.10 *Development*