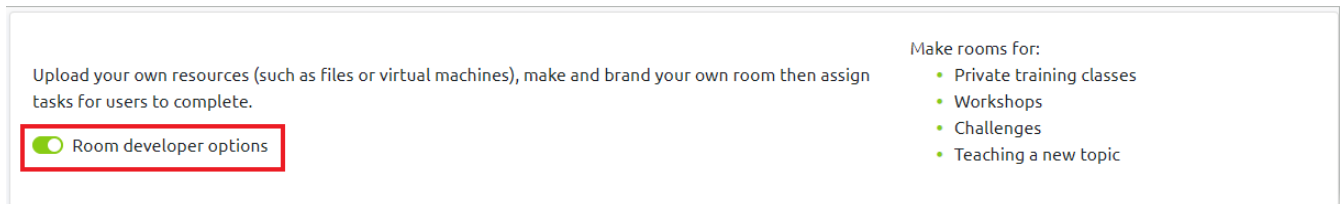


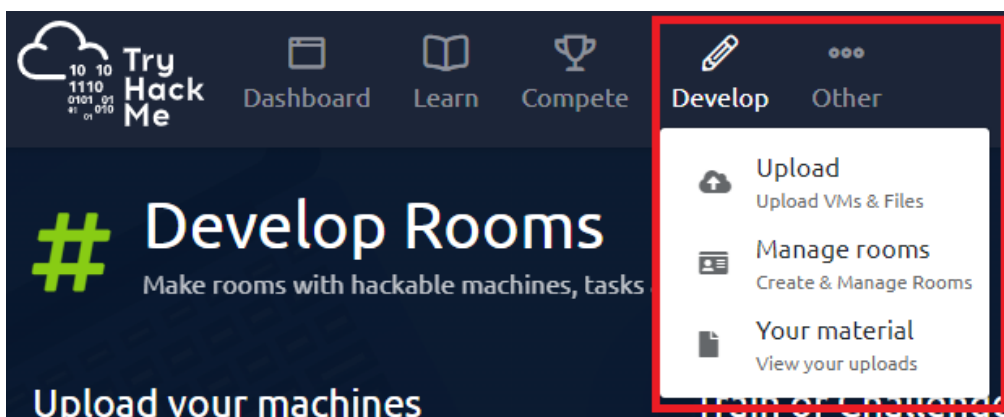
TryHackMe CTF-Erstellung

1. Vorbereitung

Um überhaupt eigene Beiträge auf der Seite TryHackMe veröffentlichen zu können, müssen zunächst die **Room Developer Options** unter [Develop Rooms](#) aktiviert werden.



Anschließend sind alle Optionen in der Menüleiste unter dem Eintrag **Develop** auffindbar.



Die Erstellung eines CTFs in TryHackMe umfasst mindestens das Anlegen eines [Raumes](#). Für gewöhnlich wird diesem Raum noch [zusätzliches Material](#) beigefügt (VM oder Datei z.B. PCAP).

Laut Dokumentation sind Image-Uploads von Windows-, Ubuntu-, RHEL-, SLES-, CentOS-, Debian-, Oracle- und Fedora-VMs erlaubt ([Versionsnummer-Beschränkung](#) beachten). Allerdings werden nicht alle Images gleichermaßen gut im Upload akzeptiert. Unter anderem wurde eine Debian 8.0 VM nicht akzeptiert (Status einsehbar unter [Your Material](#)).

Der [Blog-Eintrag von CYKNOW](#) kann bei der Erstellung ebenfalls zu Rate gezogen werden.

Im Folgenden wird ein Image von Ubuntu Server verwendet.

1.1. Installation Ubuntu Server

Folgende Ubuntu Images sind erlaubt:

- Ubuntu Server 12.04, 12.10, 13.04, 13.10, 14.04, 14.10, 15.04, 16.04, 16.10, 17.04, 18.04, 20.04
- Download 20.04 unter <https://ubuntu.com/download/server>

Nach dem Download muss das Image auf einer VM installiert werden (bspw. mit Virtual Box). Die Einstellungen für RAM und Festplattengröße sind nur während des lokalen Setups der CTF hilfreich. Das gehostete VM-Image in TryHackMe richtet sich nach deren beschränkten Ressourcen (ggf. kann mehr RAM/ vCPUs beantragt werden).

Für die lokale Installation sollte eine Internetverbindung (NAT oder Netzwerkbrücke) bestehen.

Hier wurde bei der Installation ein Nutzer

```
user: user
password : $3cuR3_p4ss
```

erstellt und zusätzlich die Dienste SSH und Docker hinzugefügt.

Profile setup

[Help]

Enter the username and password you will use to log in to the system. You can configure SSH access on the next screen but a password is still needed for sudo.

Your name:

Your server's name:
The name it uses when it talks to other computers.

Pick a username:

Choose a password:

Confirm your password:

[Done

]

You can choose to install the OpenSSH server package to enable secure remote access to your server.

[X] Install OpenSSH server

Import SSH identity: [No ▼]

You can import your SSH keys from GitHub or Launchpad.

Import Username:

[X] Allow password authentication over SSH

[Done]

[Back]

These are popular snaps in server environments. Select or deselect with SPACE, press ENTER to see more details of the package, publisher and versions available.

[]	microk8s	Kubernetes for workstations and appliances	▶
[]	nextcloud	Nextcloud Server - A safe home for all your data	▶
[]	wekan	The open-source kanban	▶
[]	kata-containers	Build lightweight VMs that seamlessly plug into the c	▶
[*]	docker	Docker container runtime	▶
[]	canonical-livepatch	Canonical Livepatch Client	▶
[]	rocketchat-server	Rocket.Chat server	▶
[]	mosquitto	Eclipse Mosquitto MQTT broker	▶
[]	etcd	Resilient key-value store by CoreOS	▶
[]	powershell	PowerShell for every system!	▶
[]	stress-ng	tool to load and stress a computer	▶
[]	sabnzbd	SABnzbd	▶
[]	wormhole	get things from one computer to another, safely	▶
[]	aws-cli	Universal Command Line Interface for Amazon Web Servi	▶
[]	google-cloud-sdk	Google Cloud SDK	▶
[]	slcli	Python based SoftLayer API Tool.	▶
[]	doctl	The official DigitalOcean command line interface	▶
[]	conjure-up	Package runtime for conjure-up spells	▶
[]	postgresql10	PostgreSQL is a powerful, open source object-relatio	▶
[]	heroku	CLI client for Heroku	▶
[]	keepalived	High availability VRRP/BFD and load-balancing for Lin	▶
[]	prometheus	The Prometheus monitoring system and time series data	▶
[]	juju	Juju - a model-driven operator lifecycle manager for	▶

[Done]

[Back]

```
Install complete! [ Help ]

configuring apt configuring apt
installing missing packages
configuring iscsi service
configuring raid (mdadm) service
installing kernel
setting up swap
apply networking config
writing etc/fstab
configuring multipath
updating packages on target system
configuring pollinate user-agent on target
updating initramfs configuration
configuring target system bootloader
installing grub to target devices
finalizing installation
  running 'curtin hook'
    curtin command hook
executing late commands
final system configuration
  configuring cloud-init
  calculating extra packages to install
  installing openssh-server
    curtin command system-install
  downloading and installing security updates
    curtin command in-target
  restoring apt configuration
    curtin command in-target
subiquity/Late/run

[ View full log ]
[ Reboot Now ]
```

Nach dem Reboot und anschließendem Login müsste jetzt auch der SSH-Port zu sehen sein.

```
user@ctf:~$ ss -tulnp
Netid      State      Recv-Q     Send-Q      Local Address:Port      Peer Address:Port      Process
udp        UNCONN     0           0            127.0.0.53%lo:53        0.0.0.0:*
udp        UNCONN     0           0            10.0.2.15%enp0s3:68     0.0.0.0:*
tcp        LISTEN     0          4096         127.0.0.53%lo:53        0.0.0.0:*
tcp        LISTEN     0          128         0.0.0.0:22             0.0.0.0:*
tcp        LISTEN     0          128         [::]:22                [::]:*
```

1.2. Wiederverwendbarkeit VM-Image

Um den Installationsschritt für künftige CTFs zu sparen, sollte das soeben erstellte Image nicht weiter verändert werden. Das eigentliche CTF wird von diesem Base-Image geklont und modifiziert. Dabei ist allerdings zu beachten, dass mit der Zeit möglicherweise Schwachstellen im Kernel oder den vorinstallierten Diensten entdeckt werden. Deswegen sollte man sicherheitshalber, damit kein ungewollter trivialer Exploit zum Ziel führt, nicht vergessen werden vor der Ausgestaltung des CTFs ein `apt-get update && apt-get upgrade` einzuplanen. Des Weiteren lässt sich ein Großteil der ungewünschten PrivEsc-Pfade mit den bekannten Tools `linenum.sh` oder `LinPeas` herausfiltern.

In VirtualBox klont man per Rechtsklick auf die VM und *Klonen* oder `STRG+O` (Vollständiger Klon).

1.3. Export VM-Image

Die spezifischen Planungs- und Konfigurationsschritte, um ein interessantes CTF zu gestalten werden in den folgenden Kapiteln vorgestellt. Sobald man damit fertig ist, muss das VM-Image als `.ova` oder `.vmdk` exportiert werden.

Anschließend unter *Develop* --> *Upload* hochladen:



Upload

Upload virtual machines or files to use in room tasks

Title

UbuntuServer_UploadTest

Description

Room for first year University students...

Type

☒ VM ☐ Downloadable File

Upload (Accepted file types: .ova, .vmdk)

THM_Docker.ova

Browse

Upload and Finish

Nach Bestätigung sollte der Upload-Fortschritt zu sehen sein (**Browser nicht aktualisieren!**):



Upload

Upload virtual machines or files to use in room tasks



We are uploading your file. Please be patient, manually refreshing this page will destroy all progress..

Percentage uploaded: **12.98%**

Cancel Upload

Jetzt wird die VM unter [Your Material](#) im Zustand *Converting ...* angezeigt.

Uploads

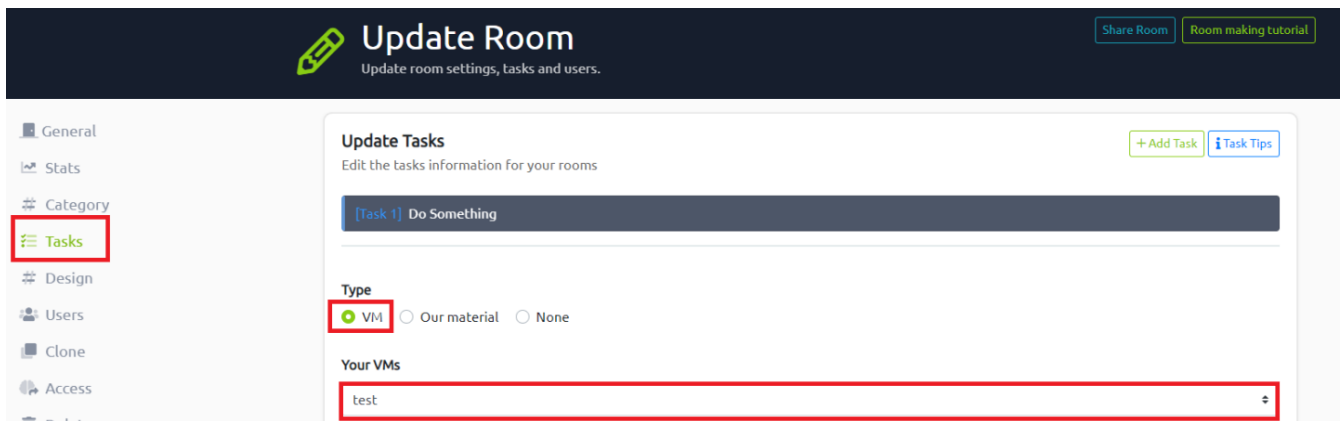
Ready to use	Name	State	Delete
	UbuntuServer_UploadTest	Converting...	

Nach etwa 30 Minuten sollte sich der Zustand verändern. Gelegentlich kann es aber sein, dass nach längerer Zeit die VM bei **Converted 19%** stecken bleibt. In diesem Fall hilft es den Upload abubrechen und erneut zu versuchen.

Uploads

Ready to use	Name	State	Delete
	test	Successfully converted.	

Die soeben erstellte VM findet man in der Raumgestaltung unter dem Menüpunkt *Tasks*.



Über die Raum-URL (<https://tryhackme.com/room/<roomCode>>) sollte nun die Option *Start Machine* zu sehen sein.



Der finale Test wäre nun noch zu überprüfen, ob die Dienste der VM den vorherigen Konfiguration entsprechen. In diesem einfachen Beispiel wurde der SSH-Dienst aktiviert für den Nutzer `user` mit Passwort `$3cuR3_p4ss`.

Wahlweise kann man mit der TryHackMe [VPN-Verbindung](#) oder mithilfe der AttackBox im Browser prüfen.

```
root@ip-10-10-203-114:~# ssh user@10.10.227.162
The authenticity of host '10.10.227.162 (10.10.227.162)' can't be established.
ECDSA key fingerprint is SHA256:ocqF7k6y6eYcSxY8ey+iM2akmepKrI0BrcMi07PmA+A.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.10.227.162' (ECDSA) to the list of known hosts.
user@10.10.227.162's password:
Welcome to Ubuntu 20.04.5 LTS (GNU/Linux 5.4.0-135-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Wed 21 Dec 2022 08:43:35 AM UTC

System load:  0.0               Processes:            112
Usage of /:   43.6% of 6.06GB   Users logged in:     0
Memory usage: 27%              IPv4 address for docker0: 172.17.0.1
Swap usage:   0%               IPv4 address for eth0:  10.10.227.162

updates can be applied immediately.
see these additional updates run: apt list --upgradable

New release '22.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Tue Dec 20 14:53:56 2022
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

user@ctf:~$
```

Zusammengefasst wurde mit diesen vorbereitenden Schritten eine wiederverwendbare Basis zur Konzeption von CTFs und zugehörigem Material (VM/Dateien) aufgebaut. Die nächsten Kapitel befassen sich mit der Ausgestaltung eines individuellen CTFs.

2. Aufgabenplanung

In dem bereits erwähnten [Blog-Eintrag](#) wurden von dem Autor und einigen Veteranen/ Mitarbeitern in TryHackMe Empfehlungen für die Vorbetrachtung ausgesprochen. Zusammengefasst kann die vorbereitende Auseinandersetzung mit folgenden Fragestellungen hilfreich sein:

- Wie sieht die Killchain (*initial access* und *privEsc*) aus?
- Welches Lernziel bzw. welchen Schwerpunkt soll der Raum bieten?
- Wie schwer soll der Raum sein; sollten Rabbit Holes eingebaut werden?
- Welche Dienste müssen auf dem System laufen?
 - ... und welche nicht (sonst Vereinfachung über *Living off the Land*)
- Wie muss das Netzwerk konfiguriert werden (i.d.R. nur die Firewall)?

Gleichzeitig darf man nicht außer Acht lassen, dass repetitive Vorgehensweisen das Publikum mit der Zeit langweilen.

2.1. Beispiel (Container basiertes CTF)

Definition der Killchain:

1. Container-Image in offener Container-Registry mit Klartext DB-Passwort
2. Extraktion der Nutzer und Passwort-Hashes mithilfe DB-Passwort aus MySQL-Datenbank
 - nun ist SSH-Login für Nutzer `claire-r` und für 4 User Web-App Login möglich
3. Ausnutzung des JS `eval()`-Befehls bei unsicherer Implementierung in Web-App
 - Reverse Shell in den Web-App Container
4. Da der Web-App Container als root-User läuft und lokale Schreibrechte auf das gemountete Container-Volume bestehen, können analog zu [Privilege Escalation with 2 Shells and host mount](#) eine root-Shell erzeugt werden.

Die primären Lernziele liegen in Schwachstellen der Container-Konfiguration. D.h.:

- containerisierte Prozesse unter dem root-User
- fehlkonfigurierte volume mounts
- Credential-Leak über Umgebungsvariablen im Container Image
- Interaktion mit einer Container-Registry

Zweitrangig ist die `eval`-Schwachstelle in der Webanwendung. Diese ist lediglich ein Mittel, um den in 4 aufgeführten PrivEsc-Pfad zu ermöglichen.

3. Technische Planung

Da die mit dem Raum bereitgestellte VM innerhalb des TryHackMe-Netzwerks keine Internetverbindung haben wird, müssen unbedingt alle Ressourcen, die lokal benötigt werden könnten zuvor heruntergeladen werden. Das gilt beispielsweise auch für Container Base-Images, die dann nicht von Docker Hub heruntergeladen werden können.

3.1. Kommunikationsmatrix

Dienst	Port	Interface
SSH	22/tcp	eth0
Webserver	80/tcp	eth0, docker network
MySQL	3306/tcp	eth0, docker network
Docker Registry	5000/tcp	eth0

3.2. Nutzer

Die einzige Bruteforce-Attacke, die im gesamten CTF vorgesehen ist, ist das Hashcracking. Aus diesem Grund sind die Nutzernamen so gewählt, dass es viel zu lange dauern würde die Kombination aus Nutzernamen + Passwort durchzuprobieren.

3.2.1. Lokale Nutzer

Nutzerkonto	Passwort	Gruppen/Rechte
root		uneingeschränkt
claire-r	Password1	eingeschränkt, Lese/Schreibrechte auf gemountetes Volume des Web-App Containers

3.2.2. Dienste

Nutzerkonto	Dienst	Passwort	MD5-Hash
claire-r	SSH	Password1	
root	MySQL	Ng1-f3!Pe7-e5?Nf3xe5	
claire-r	Web-App	Password1	2ac9cb7dc02b3c0083eb70898e549b63
chris-r	Web-App	letmein	0d107d09f5bbe40cade3de5c71e9e9b7
jill-v	Web-App	sunshine1	d5c0607301ad5d5c1528962a83992ac8
barry-b	Web-App	sandwich	4a04890400b5d7bac101baace5d7e994

3.3. Web-Applikation

Um die Anwendung einigermaßen realistisch wirken zu lassen, wird eine simplistische Time-Tracking Applikation erstellt:

- Login analog zu: <https://codeshack.io/basic-login-system-nodejs-express-mysql/>
- Time Tracking Interface

Auf dem **Time Tracking Interface** ist eine Programmierschwachstelle eingebaut, worüber der Angreifer eine Reverse Shell zu dem zugrundeliegenden Container aufbauen kann.

Der Volume Mount wird zum Logging der Anwendung verwendet. Auf Hostseite müssen dem Mount die Rechte `chmod 766 logs` gegeben werden.

Insgesamt verfügt die Anwendung über 3 Endpunkte

Endpunkt	Methode	Verhalten
/	GET	Falls Login-Session etabliert: Anzeige des TimeTracking-Interface, sonst Loginseite
/auth	POST	Übergabe Nutzernamen und Passwort, falls Nutzernamen + Passworthash als DB-Eintrag vorhanden --> Login-Session
/time	POST	Falls Login-Session etabliert: Zeit auf eingeloggtes Nutzerkonto in Datenbank addieren.

Sowohl `/auth` und `/time` werden *form encoded* direkt über die Webseite angesprochen. Das heißt es soll nicht notwendig sein die Endpunkte über *Directory Busting* aufzuspüren (diese Technik ist nach persönlichem Empfinden zu stark in THM-Räumen repräsentiert).

3.4. MySQL Datenbank

Der MySQL-Server wird gemeinsam mit der Web-Anwendung über `docker-compose` gestartet. Die Datenbank selbst wird über ein `docker-entrypoint-initdb.d`-SQL-Script befüllt. Es muss nur eine Relation (users) mit folgenden

Attributen hinzugefügt werden:

- **user:** varchar(10)
- **pass:** varchar(32)
- **time:** integer

3.5. Container Registry

Es wird eine containerisierter, lokaler Registry-Server wie in der [Docker Dokumentation](#) beschrieben aufgesetzt. Darauf wird das gebaute Container Image der Webanwendung hochgeladen, sodass der Angreifer dieses außen per `docker pull` beziehen und daraus sowohl die Umgebungsvariablen (DB-Nutzer und -Passwort) als auch den Quelltext der Webanwendung extrahieren kann.

3.6. Docker

Auf dem System laufen drei Container:

- Webanwendung (Port 8080)
- SQL-Server (Port 3306)
- Docker-Registry (Port 5000)

, welche allesamt als `restart: always` konfiguriert werden müssen.

Der lokale Nutzer `claire-r` darf nicht Teil der Docker-Gruppe sein. Sonst wäre ein ungewünschter, trivialer PrivEsc-Pfad direkt vorhanden.

3.7. SSH

Die Default-Konfiguration der `sshd.conf` erlaubt allen Nutzern des Systems sich auch per SSH anzumelden. Hier ist also keine weitere Anpassung notwendig.

3.8. Flags

Flag	Typ	Ablageort
<code>THM{d832c0e4cf71312708686124f7a6b25e}</code>	User Flag	<code>/home/claire-r/flag.txt</code>
<code>THM{1e15fbe7978061c6bb1924124fd9eab2}</code>	Root Flag	<code>/root/flag.txt</code>

4. Technische Implementierung

Der zeitaufwändigste Teil der Implementierung ist die Entwicklung der Web-Anwendung. Man findet den Quelltext gemeinsam mit den Docker-Dateien innerhalb des [Github-Repositories](#).

Im Anschluss wird das Ubuntu-Server Base-Image folgendermaßen konfiguriert:

1. Nutzer claire-r anlegen

```
adduser claire-r
# Password : Password1
```

2. Flags platzieren

```
sudo echo "THM{d832c0e4cf71312708686124f7a6b25e}" > /home/claire-r/flag.txt
sudo echo "THM{1e15fbe7978061c6bb1924124fd9eab2}" > /root/flag.txt
```

3. Webanwendung klonen und mit MySQL deployen

```

su claire-r
cd ~
git clone https://github.com/brunofight/ContainerSecurity.git

# Nur den Quellcode-Anteil des Repos behalten.
# Der Quelltext wird bewusst auf dem System liegen gelassen, um dem Angreifer
# mehrere Wege zu lassen, wie er auf die eval-Schwachstelle kommt.
cp -r ContainerSecurity/THM-Room ./timeTracker-src
rm -rf ContainerSecurity
docker build . -t umbrella/timetracking:latest
docker-compose up -d

```

4. Registry-Dienst starten und Image pushen

```

docker run -d -p 5000:5000 --restart=always --name registry registry:2
docker image tag umbrella/timetracking:latest localhost:5000/umbrella/timetracking:latest
docker image push localhost:5000/umbrella/timetracking:latest

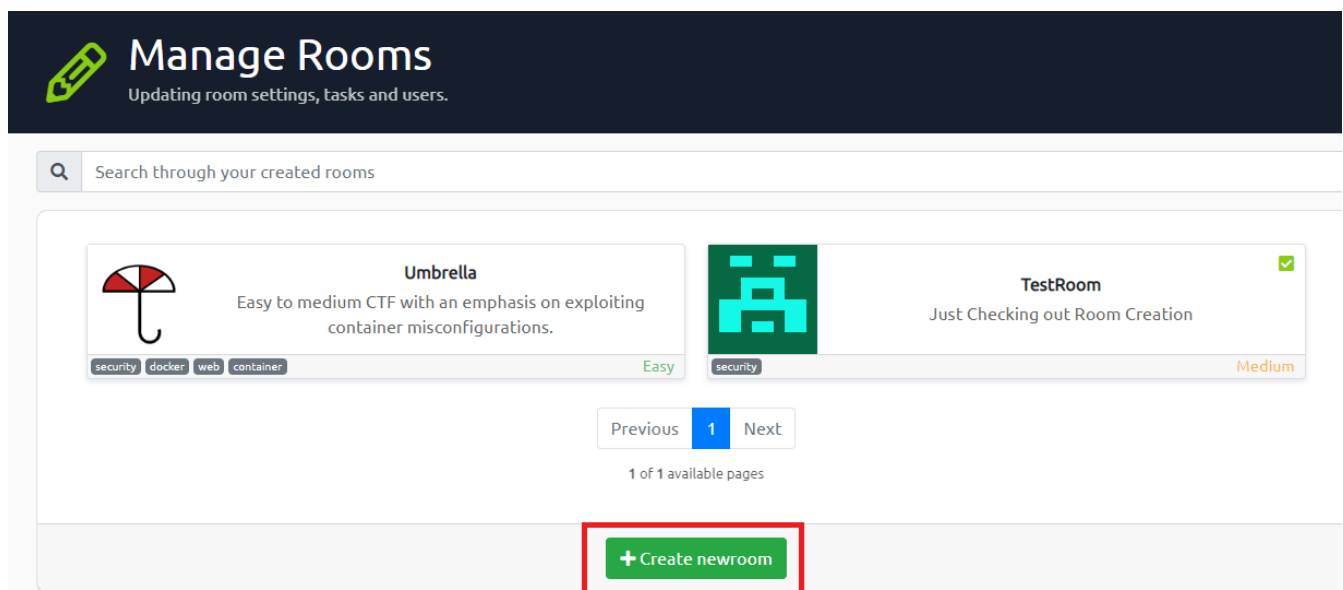
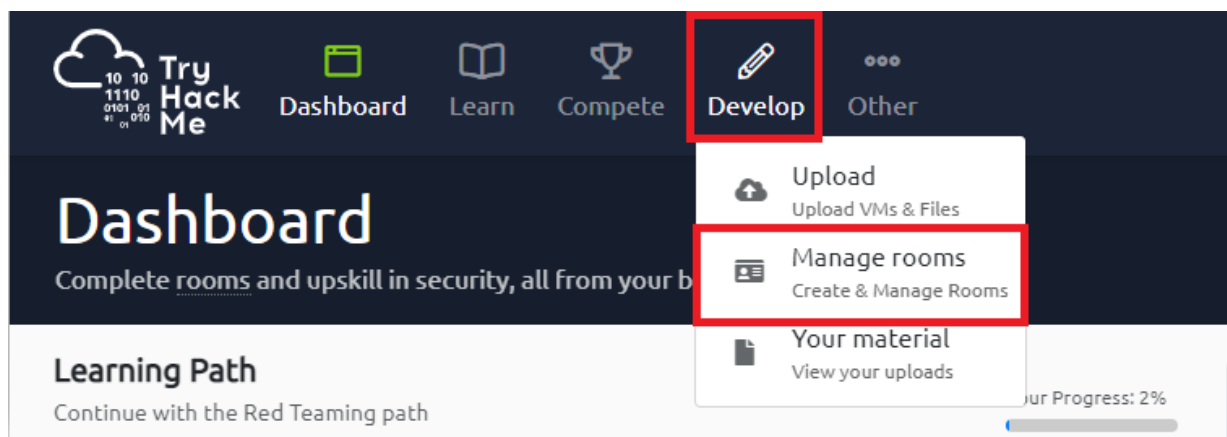
# Anmerkung, da die Registry kein TLS-Zertifikat verwendet, muss der Angreifer
# seine Docker-Daemon config um eine insecure registry ergänzen
# {"insecure-registries":["<ip>:5000"]}

```

5. Raumgestaltung

Als Erstes wird das VM-Image analog zu 1.3. exportiert und auf TryHackMe hochgeladen.

5.1. Anlegen eines neuen Raumes





Create a room

Create a room, upload your material and assign tasks!

Title

Umbrella

Description

Room for first year University students...

Anzeigertext im Banner des Raumes

Type

☒ Challenge ☐ Walkthrough

Challenge: ohne Hilfe, wenig Beschreibungstext

Walkthrough: maßgebliche Hilfestellung, idR mehr Fragen und Begleittexte zur Hilfe

Tags (Help people find your room)

security x docker x container x web x

Seperate by commas Frei wählbar, keine vordefinierten Tag-Namen

Room Completion Time (in minutes)

90

Grobe Schätzung, nahezu willkürlich wählbar

Room Icon (Optional)

Choose file

Browse

Creating a room is private and will not be public.

Create Room

5.2. Anpassung

Nach der initialen Erstellung gelangt man zurück auf das Menü *Manage Rooms*. Von hier aus kann man den neuen Raum auswählen, um das *Update Room* Interface zu öffnen.

Unter dem Menüpunkt *Tasks* werden Fragen erstellt und das hochgeladene VM-Image zugeordnet.

Falls das zuvor hochgeladene Image schon vollständig konvertiert und akzeptiert wurde, sollte diese sich unter *Your VMs* auswählen lassen (Unterpunkt nur sichtbar, wenn *Type VM* Radiobutton ausgewählt).

Im Bereich *Questions, Answers and Hints* können neue (leere) Fragen mit "Add more questions" hinzugefügt werden. Die Reihenfolge der Fragen kann per *Drag&Drop* vertauscht werden.

Questions, Answers and Hints

Question #1	DB Password	Answer	Ng1-f3!Pe7-e5?Nf3xe5	▼
Question #2	User <u>Flag</u>	Answer	THM{d832c0e4cf71312708686124f7a6b25e}	▼
Question #3	Root <u>Flag</u>	Answer	THM{1e15fbe7978061c6bb1924124fd9eab2}	▼

+ Add more questions Delete last question

Jede Frage kann um eine Antwort und einen Hinweis ergänzt werden:

Question #4	To hack into the machine and ...	Add Hint	▼
		Add Answer	


+ Add more questions Delete last question

Fragen ohne Antworten sind für Challenge-Räume ungewöhnlich, werden allerdings in Walkthrough-Räumen durchaus zur Bestätigung (*I have read and understood xyz*) eingesetzt.

Fragen können bei größeren Räumen oder Walkthroughs thematisch als *Tasks* gruppiert werden.

5.3. Veröffentlichung

Per Default ist der Raum privat. D.h. der Raum wird nicht unter *Rooms* gelistet und Nutzer können nur über den *Share Link* oder *Room Code* darauf zugreifen.

 **Update Room**
Update room settings, tasks and users.

Share Room Room making tutorial

General

General

Share your room

Privately share your room with the following link:

Copy

Users can also join this room by going to their "My Rooms" page, and entering


Check your room out!

Unter General --> *Publicly Accessible* kann ein Raum öffentlich gemacht werden. Damit dieser aber wirklich von den Betreibern publiziert wird, muss:


- ein offizielles Write-Up unter *Design* --> *Official Writeup* hochgeladen
- von den Betreibern geprüft und akzeptiert

werden.

Es ist auch möglich bei der Raumgestaltung mit anderen Nutzern zu kollaborieren:

 General


 Stats


 Category


 Tasks

 Design

 Users


 Clone

 Access


 Delete

Access - Share access to your room and allow others to collaborate!

[Redacted share link]

 Copy Share Link



 Warning! If multiple access users edit tasks, it will overwrite with the last edit.