|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | |
| |  |  | | --- | --- | |  |  | |  |  | |  |  | |  |  | | |  | |
| PE - Security Advanced  *Handen vuil maken om ervaring op te doen in de security tools* | | | |
|  |  | | |  |  |  | | --- | --- | --- | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |
|  |  | | Groep 1: Kurt Clemens, Jelle Schreurs, Thierry Noelmans, Tom Appeltants |

Inhoud

[**Opdrachten** 3](#_Toc104549574)

[**Red Team** 3](#_Toc104549575)

[**Linux Privilege Escalation Arena**Star | Mario Wiki | Fandom 3](#_Toc104549576)

[Task 1/2/3 3](#_Toc104549577)

[Task 4 3](#_Toc104549578)

[Task 5 3](#_Toc104549579)

[Task 6 3](#_Toc104549580)

[Task 7 5](#_Toc104549581)

[Task 8/9/10/11 5](#_Toc104549582)

[Task 12 5](#_Toc104549583)

[Task 13 6](#_Toc104549584)

[Task 14 6](#_Toc104549585)

[Task 15/16/17/18/19 6](#_Toc104549586)

[Overzicht van afgewerkte room op TryHackMe 7](#_Toc104549587)

[**Frank & Herbert**Star | Mario Wiki | FandomStar | Mario Wiki | Fandom 8](#_Toc104549588)

[Task 1 8](#_Toc104549589)

[Overzicht van afgewerkte room op TryHackMe 14](#_Toc104549590)

[**Blue Team** 15](#_Toc104549591)

[**Autopsy Star | Mario Wiki | Fandom** 15](#_Toc104549592)

[Task 1 15](#_Toc104549593)

[Overzicht van afgewerkte room op TryHackMe 23](#_Toc104549594)

[**Windows Forensics Star | Mario Wiki | Fandom** 24](#_Toc104549595)

[Task 1/2/3/4/5/6/7/8/9 24](#_Toc104549596)

[Task 10 24](#_Toc104549597)

[Overzicht van afgewerkte room op TryHackMe 28](#_Toc104549598)

[**Juicy Details Star | Mario Wiki | FandomStar | Mario Wiki | Fandom** 30](#_Toc104549599)

[Task 1 30](#_Toc104549600)

[Task 2 30](#_Toc104549601)

[Task 3 31](#_Toc104549602)

[Overzicht van afgewerkte room op TryHackMe 33](#_Toc104549603)

[**Yellow Team** 34](#_Toc104549604)

[Console Application Star | Mario Wiki | Fandom 34](#_Toc104549605)

[Gebruik Auth0 Star | Mario Wiki | Fandom 38](#_Toc104549606)

[Eenvoudige MVC met Auth0 Star | Mario Wiki | FandomStar | Mario Wiki | Fandom 45](#_Toc104549607)

[Uitbreiding MVC basic / advanced Star | Mario Wiki | FandomStar | Mario Wiki | FandomStar | Mario Wiki | Fandom 54](#_Toc104549608)

# **Opdrachten**

Er is gekozen om de volgende opdrachten af te leggen:

* Red Team
  + Linux Privilege Escalation Arena Star | Mario Wiki | Fandom
  + Frank & Herbert Star | Mario Wiki | FandomStar | Mario Wiki | Fandom
* Blue Team
  + Autopsy Star | Mario Wiki | Fandom
  + Windows Forensics Star | Mario Wiki | Fandom
  + Juicy Details Star | Mario Wiki | FandomStar | Mario Wiki | Fandom
* Yellow Team
  + Console application Star | Mario Wiki | Fandom
  + Gebruik Auth0 Star | Mario Wiki | Fandom
  + eenvoudige MVC met Auth0 Star | Mario Wiki | FandomStar | Mario Wiki | Fandom
  + uitbreiding MVC basic / advanced Star | Mario Wiki | FandomStar | Mario Wiki | FandomStar | Mario Wiki | Fandom

Dit zorgt ervoor dat er in totaal voor 14 sterren aan opdrachten is gemaakt. De documentatie kan terug gevonden worden per ‘Security Team’.

# **Red Team**

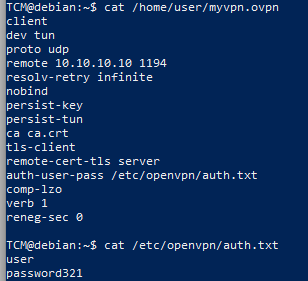
## **Linux Privilege Escalation Arena** Star | Mario Wiki | Fandom

Deze opdracht dient uitgevoerd te worden op <https://tryhackme.com/room/linuxprivescarena>

### Task 1/2/3

Tijdens deze taken wordt de opdracht duidelijk omschreven wat de situatie is en wat er dient te gebeuren.

### Task 4

De gegeven commando’s in volgorde gebruikt om tot het volgende resultaat te komen: 

### Task 5

Er is één commando gegeven, deze gebruikt om tot volgend resultaat te verkrijgen.  
Afbeelding met tekst

Automatisch gegenereerde beschrijving

In het resultaat kan je afleiden dat er een poging is geweest om verbinding te maken met ‘MYSQL’ met volgende informatie:

* user -u: root
* password -p: password123

### Task 6

Eerst hebben we de inhoud van /etc/passwd en /etc/shadow gekopieerd als backup.  


Vervolgens hebben we deze files via SCP overgezet naar onze attacker-machine.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Alvorens we het commando ‘unshadow’ kunnen gebruiken hebben we deze bin moeten installeren.  
Afbeelding met tekst

Automatisch gegenereerde beschrijving

Hetzelfde dient te gebeuren voor het commando ‘hashcat’.Afbeelding met tekst

Automatisch gegenereerde beschrijving

Vervolgens hebben we de file rockyou.txt gedownload via <https://github.com/brannondorsey/naive-hashcat/download/data/rockyou.txt> om vervolgens het commando te kunnen uitvoeren.Afbeelding met tekst

Automatisch gegenereerde beschrijving

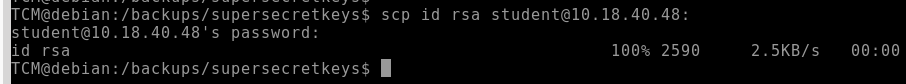
Na het uitvoeren van de hashcate kregen we het passwoord te zien. Afbeelding met tekst

Automatisch gegenereerde beschrijving

De /etc/shadow file had ook de volgende permissies: 

### Task 7

De gegeven commando’s uitgevoerd:   


Deze zijn dan via ‘scp’ naar de attacker-machine gekopieerd. 

Hier zie je dan ook het volledige pad waar de file opgeslagen is.

### Task 8/9/10/11

Er zijn geen antwoorden nodig voor deze opdrachten. Alle commando’s uitgevoerd.

### Task 12

Er dient een Nginx version: 1.6.2-5+deb8u3 opgestart worden.  
Als Nginx gestart is kan je hier de CVE exploit vinden.   
Afbeelding met tekst

Automatisch gegenereerde beschrijving

Een tweede terminal openen om een set van commando’s uit te voerenAfbeelding met tekst

Automatisch gegenereerde beschrijving

Na het uitvoeren van deze commando’s kregen we een nieuwe shell in onze eerste terminal (Nginx VM). Hier kan je zien dat we dit via /usr/sudo SUID binary gedaan werd. Afbeelding met tekst

Automatisch gegenereerde beschrijving

### Task 13

Gegeven commando’s gebruikt om het laatste resultaat te zoeken van “strings /usr/local/bin/suid-env”. Afbeelding met tekst

Automatisch gegenereerde beschrijving

### Task 14

Gegeven commando’s gebruikt om het laatste resultaat te zoeken van “strings /usr/local/bin/suid-env2”.Afbeelding met tekst

Automatisch gegenereerde beschrijving

### Task 15/16/17/18/19

Er zijn geen antwoorden nodig voor deze opdrachten. Alle commando’s uitgevoerd.

### Overzicht van afgewerkte room op TryHackMe



## **Frank & Herbert** Star | Mario Wiki | FandomStar | Mario Wiki | Fandom

Deze opdracht dient uitgevoerd te worden op <https://tryhackme.com/room/frankandherby>

### Task 1

Deze opdracht bestaat uit 1 grote taak. Het geleverde werk is opgesplitst in de 4 vragen van deze opdracht.

#### Task 1.1

Er wordt gevraagd op welke poort Frank zijn webpagina heeft. We moeten hiervoor de juiste poort vinden dus weten we dat we de tool ‘nmap’ hiervoor kunnen gebruiken.

Dus hebben we het IP-adres van Frank & Herb hun app door ‘nmap’ laten scannen. Afbeelding met tekst

Automatisch gegenereerde beschrijving

Hieruit blijkt dat er 3 poorten gebruikt worden:

22 voor ssh verbinding

3000 voor ppp

31337 voor http

Omdat http voor website gebruikt wordt is dit de poort die Frank voor zijn webpagina gebruikt.

#### Task 1.2

Er wordt gevraagd om te zoeken wat Frank beschikbaar heeft gelaten op zijn website.

Vraag: Hoe vind je beschikbare webbestanden?  
Oplossing: google search term = “Linux program to find hidden web files”

Afbeelding met tekst, schermafbeelding, zwart, scherm

Automatisch gegenereerde beschrijving

Dit gaf ons ‘UrlBuster’ als resultaat.  
Hierna zijn we gaan kijken hoe ‘UrlBuster’ werkt en hoe te gebruiken. Het lukte ons niet onmiddellijk via ‘UrlBuster’. Vervolgens zijn we naar alternatieven gaan zoeken. Een volgende oplossing was ‘DirBuster’, na het installeren van ‘DirBuster’ vonden we uitleg om de <url:poort> te scannen. Dus in ons geval <http://10.10.217.45:31337>. Dit zou 43553999 dagen duren dus hebben we afgebroken en gekeken wat we nog konden doen.

Na wat zoekwerk vonden we een Kali-Linux tool dirsearch wat opdezelfde manier zou werken als DirBuster.

Dus hebben we onze <url:poort> ook eens door ‘dirsearch’ gehaald.

Dit geeft al vrij snel resultaten weer waarvan 2 met een 200 response.Afbeelding met tekst, schermafbeelding, computer, zilver

Automatisch gegenereerde beschrijving

In de hint viel ons onmiddellijk op “git you here”. Was dit een spellingsfout?  
Maar na onze ‘dirsearch’ hadden we een 200 response van /.git-credentials.

De website van Frank was 10.10.217.45:31337 dus hebben we eens geprobeerd om /.git-credentials erachter toe te voegen en dit gaf ons een pop-up om een bestandje te saven.Afbeelding met tekst

Automatisch gegenereerde beschrijving

Afbeelding met tekst, schermafbeelding, monitor, zwart

Automatisch gegenereerde beschrijving

We hebben een bestand gevonden in .git-credentials. Wat ons verteld dat .git-credentials is blootgesteld.

#### Task 1.3

Het eerder gevonden bestand gaf ons “http://frank:f%40an3-1s-E337%21%21@192.168.100.50”.

Dit lijkt enorm om hoe je een ssh verbinding maakt (ssh username@ip-address). Wij veronderstellen dat dit gaat zijn: [frank@192.168.100.50](mailto:frank@192.168.100.50) met zijn password. Ook omdat er bij de ‘nmap’ search een open poort 22 voor ssh verbinding stond.

De volledige tekst die wij kregen staat in URL vorm.

Door in google “URL-code” in te geven kregen we als eerste een online rul enocode/decode website. Afbeelding met tekst, schermafbeelding, monitor, zwart

Automatisch gegenereerde beschrijving

We hebben hier de tekst eens ingeven en dit gaf ons volgende resultaat: Afbeelding met tekst

Automatisch gegenereerde beschrijving

We hebben dan ssh-verbinding proberen te maken met [frank@192.168.100.50](mailto:frank@192.168.100.50) wat niet lukte.  
Maar zijn website zit op 10.10.217.45 dus hebben hier ssh-verbinding proberen mee te maken en dat is gelukt.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Er wordt gevraagd wat de user.txt flag is.

Na een ls hebben we gezien dat Frank een bestandje heeft user.txt. Om te kijken wat erin staat hebben we het commando ‘cat’ hiervan gedaan en kregen we het gevraagde resultaat.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

#### Task 1.4

Probleem: Waar beginnen we?  
Oplossing: Wie zijn we  
Afbeelding met tekst

Automatisch gegenereerde beschrijving

Hier komen de microk8s van pas waar de inleiding van deze oefening over sprak.

Probleem: Wat zijn microk8s, waar dienen ze voor en zijn er exploits?  
Oplossing: Google-search resultaat  
Afbeelding met tekst

Automatisch gegenereerde beschrijvingAfbeelding met tekst

Automatisch gegenereerde beschrijvingAfbeelding met tekst, schermafbeelding, zwart, scherm

Automatisch gegenereerde beschrijving

Hier vinden we dat er een exploit aanwezig is die een gebruiker met weinig privileges toch root kan maken.  
Afbeelding met tekst

Automatisch gegenereerde beschrijving

Hier staat een heel Proof-Of-Concept exploit uitgelegd. Dus hebben we deze stappen maar eens gevolgd om te kijken wat dit ons kan opleveren. Eerst moesten we een ‘pod.yaml-file’ aanmaken met volgende gegevens:

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Nadien moesten we deze file toepassen d.m.v. ‘microk8s.kubectl apply -f pod.yaml’. Dit gaf ons een foutmelding of the juiste host of poort gebruikt was.



We hebben dit moeten opzoeken en de oplossing is dat je de ‘pod.yaml-file’ moet toepassen op de image die al draait en dit kan je vinden door:

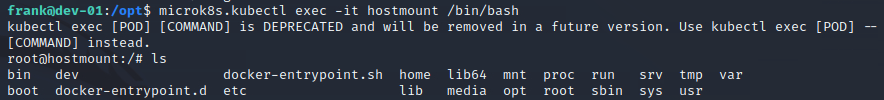
Afbeelding met tekst

Automatisch gegenereerde beschrijving

In onze situatie gaan we de image op van localhost gebruiken. Dus passen we in onze pod.yaml file de image: ubuntu:latest aan naar localhost:32000/bsnginx en proberen het microk8s.kubectl apply -f pod.yaml commando nog eens. Nu zien we dat de pod/hostmount is gelukt.



De volgende stap die we moesten volgen was een nieuwe terminal openen en hebben we hier al eens een ls gedaan:



Omdat in de ‘pod.yaml-file’ het volume “root” hebben genoemd en het mount Path: /opt/root is gaan we cd’en naar opt/root om dan de root folder te raadplegen:

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Hier hebben we een ls gedaan en vonden we het bestand root.txt.

Door te kijken wat er in dit bestand zit door cat root.txt hebben we de gevraagde flag gevonden.

### Overzicht van afgewerkte room op TryHackMe

Afbeelding met tekst, schermafbeelding, monitor

Automatisch gegenereerde beschrijving

# **Blue Team**

## **Autopsy Star | Mario Wiki | Fandom**

Deze opdracht dient uitgevoerd te worden op <https://tryhackme.com/room/autopsy2ze0> .

### Task 1

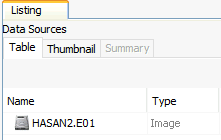
Deze opdracht bestaat uit 1 grote taak. Het geleverde werk is opgesplitst in de 15 vragen van deze opdracht.

#### Task 1.1

We hebben de ‘autopsy-file’ ingeladen die in de map “Case Files” stond. Na het inladen krijgen we een .E01 image file te zien. Na het selecteren van dit bestand kan je onderaan “File Metadata” selecteren waar de MD5 te zien is.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

 Afbeelding met tekst

Automatisch gegenereerde beschrijving

#### Task 1.2

Op locatie Results\Extracted Content\Operating System Information hebben we de informatie van het systeem gevonden met de computer name.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

#### Task 1.3

Op locatie Results\Extreacted Content\Operating System User Account hebben we de informatie van de user accounts. Hier hebben we software-, administrator-, guest-, default accounts niet genomen maar enkel de gebruikersnamen.

Afbeelding met tafel

Automatisch gegenereerde beschrijving

#### Task 1.4

Op locatie Results\Extreacted Content\Operating System User Account hebben we de informatie van de user accounts en onder de kolom “Data Accessed” kunnen we zien wanneer dit account het laatst is ingelogd.

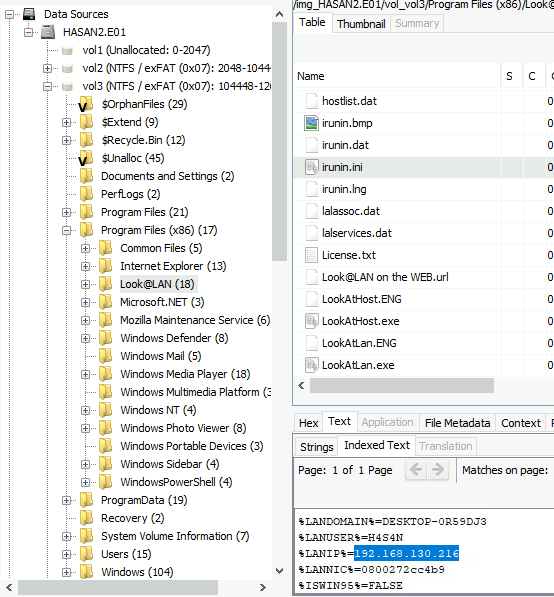
Afbeelding met tekst

Automatisch gegenereerde beschrijving

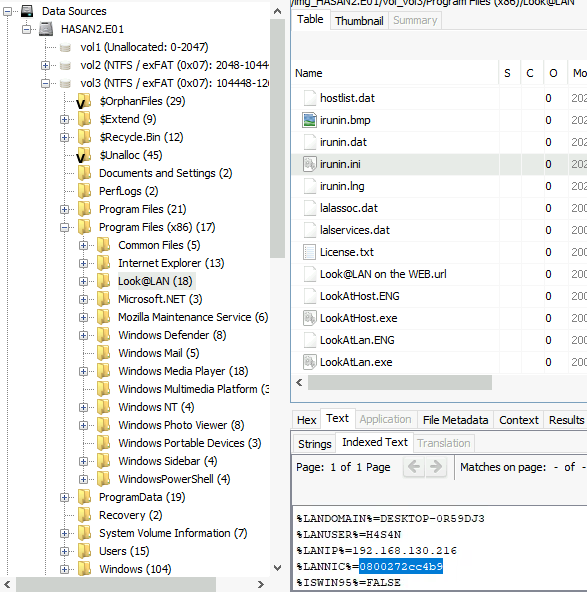
#### Task 1.5

Op locatie Op locatie Results\Extreacted Content\Installed Programs vinden we een network monitoring tool (Look@lan). Nu gaan we in “Data Sources” kijken of we dit programma terugvinden in “Program Files”. Dit hebben we teruggevonden in “vol3”. In de logfile “irunin.ini” vinden we de log bestanden met hierin “LANIP” wat ons IP address is.





#### Task 1.6

In dezelfde file die we gebruikt hebben in de vorige vraag vinden we ook “LANNIC” wat ons MAC address is van de computer.  


#### Task 1.7

De network card details hebben we gevonden door naar het keyword “ethernet” te zoeken en als we door de files gingen hebben we onder DiagnosticLogCSP\_Collector\_DeviceProvisioning…. de ethernet adapter gevonden.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

#### Task 1.8

Dit bestand hadden we nodig om vraag 1.5 en 1.6 op te kunnen lossen. En het network monitoring tool was Look@Lan.

#### Task 1.9

We moeten een “Bookmark” zoeken. We zien in Autopsy dat er een folder staat “Web Bookmarks”. Als we hierop klikken kunnen we in de details een bookmark voor Google Maps vinden met de coördinaten.

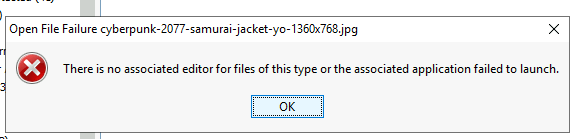
Afbeelding met tekst

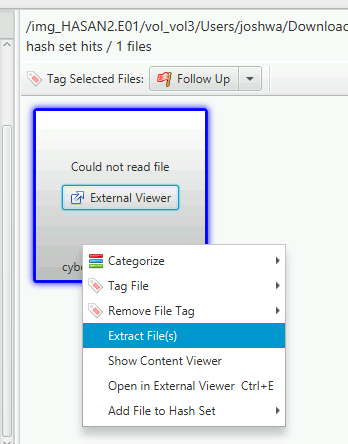
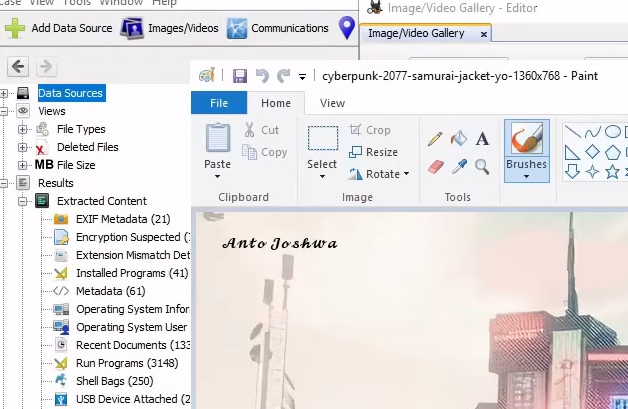
Automatisch gegenereerde beschrijving

#### Task 1.10

In Autopsy hebben we een menu “Images/Videos”. Als we hierop klikken kunnen we van elke user hun afbeeldingen bekijken. We hebben elke afbeelding bij elke user bekeken tot we bij Joshwa zijn naam hebben gezien op een van de afbeeldingen.

Probleem: We konden de afbeelding niet openen in Images viewer  
Oplossing: We konden de afbeelding extracten en dan openen in Paint.





#### Task 1.11

Als we een aangepaste file moeten zoeken denken we onmiddellijk aan logs. Ook het programma “powershell” is gebruikt.

Probleem: Waar staan deze logfiles?  
Oplossing: Google Search

Na opzoekwerk vonden we dat als de gebruiker powershell heeft gebruikt, dit terug te vinden was op volgende locatie: \user\AppData\Roaming\Microsoft\Windows\Powershell.

We zijn dan bij elke user gaan kijken of we logfiles konden vinden die horen bij “Powershell”. We hebben een txt-bestand gevonden bij Shreya in de folder “PSReadLine” onder “Powershell”. Dit bestand hebben we geselecteerd en in de meta data hebben we onder cd .\Desktop\ de eerste gegevens “Add-Content” gevonden. Dit heeft ons doen vermoeden dat dit de eerste flag van het bestand was.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

#### Task 1.12

Probleem: Waar vinden we exploit?  
 We moeten een boodsschap ind e vorm van een “flag {}” vinden.

Oplossing: Via keyword gezocht op exploit maar niet gevonden wat we zochten  
 Via keyword gezocht op flag om te kijken of dit een resultaat gaf.

Hier hebben in de resultaten een mapje exploit gevonden waarin we de gevraagde flag hebben gevonden.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

#### Task 1.13

Probleem: Welke tools worden gebruikt om paswoorden te vinden in een systeem  
 Waar kunnen we deze tools vinden

Oplossing: Google Search: tools genoeg  
 In autopsy hebben we een mapje “run programs” gevonden.

Dit gaf alle gestarte programma’s weer. Als je al deze resultaten op “username” sorteert zie je in de resultaten username “H4S4N” terugkomen. Dit is dezelfde naam als de “Data Source” die we te zien kregen na het inladen van de autopsy file. Na de sortering hebben we eens gekeken welke soorten “program names” gestart werden bij deze user. We hebben alle voor ons onbekende programma namen genoteerd en dan in google gekeken waarvoor deze programma’s dienen. Hieruit hebben we 2 resultaten gevonden -> Mimikatz en Lazagne. Tools gebruikt om authenticatie gegevens op te zoeken (paswoorden en usernames).

Afbeelding met tekst, schermafbeelding, zwart, scherm

Automatisch gegenereerde beschrijvingAfbeelding met tekst, schermafbeelding, monitor, zwart

Automatisch gegenereerde beschrijving

#### Task 1.14

Probleem: Wat is een “YARA-file”  
 Waar kunnen we dit terug vinden?

Oplossing: Google Search: “ Yara is een tool dat vooral gebruikt wordt voor malware search and   
 detection. Het baseert zich op regels die bestaan uit strings en boolean   
 expressies. En wordt weergegeven als een .yar file.”  
 Geen idee, we hebben geprobeerd door in “search keyword” .yar in te geven en gekeken   
 welke resultaten dit met zich mee bracht.

In deze resultaten vonden we een file naam kiwi\_passwords.yar.ink. omdat dit een .yar file is en het betrekking heeft tot paswoorden hadden we een vermoeden dat het deze file kon zijn. In de meta data van dit bestand hebben we ons resultaat niet gevonden. In dit resultaat hebben we wel het path van mimikatz\_trunkx (desktop van H4S4N) gevonden en zijn we dit ook eens gaan controleren.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Er stond geen mimikatz\_trunk op de dekstop. Dus hebben we dit ook eens gezocht via keyword. Hier hebben we bij results een 7z file van mimikatz\_trunk gevonden met locatie “Downloads”.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

In downloads hebben we een zip file gevonden van mimikatz\_trunk waarin opnieuw het bestand kiwi\_passwords.yar stond wat we eerder ook al zagen in het path van Desktop.

Toen we dit openden vonden we gegevens zoals description en author, deze hadden we nodig!

#### Task 1.15

Probleem: Wat is een MS-NRPC based exploit?

Oplossing: Google Search: Artikel gevonden over een bedrijf waarvan een niet-geverifieerde   
 comprise was van de domain controllers. Hier werd er de nadruk gelegd op de naam   
 “Zerologon”.

Afbeelding met tekst

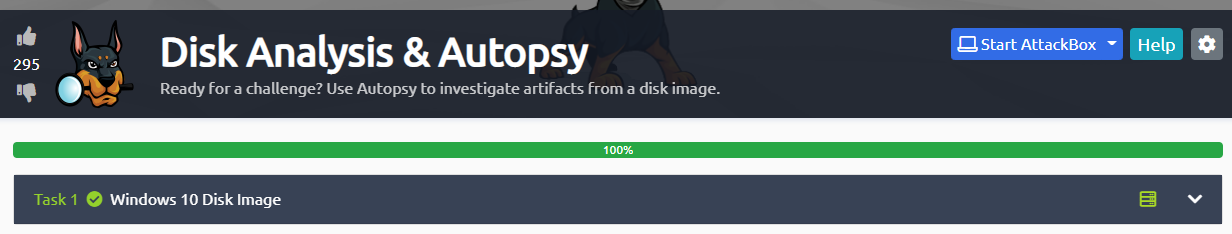
Automatisch gegenereerde beschrijving

We hebben dit eens gezocht via keyword search en vonden een resultaat.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

### Overzicht van afgewerkte room op TryHackMe



## **Windows Forensics Star | Mario Wiki | Fandom**

Deze opdracht dient uitgevoerd te worden op <https://tryhackme.com/room/windowsforensics1> .

### Task 1/2/3/4/5/6/7/8/9

Deze opdrachten kan je afwerken door de gegeven informatie te lezen en de oplossingen hier uit af te leiden. De antwoorden van alle oefeningen staan in de tekst en/of de schermafdrukken.

### Task 10

De notitie van de eerste oefening zegt dat we de registry exploreren moeten gebruiken dus hebben we alle volgende opdrachten opgelost via deze tool. (gegeven in de virtuele machine van TryHackMe)

#### Task 10.1

De hint zegt dat we de SAM hive moeten gebruiken om users terug te vinden en dat accounts startend met 10xx user accounts zijn. De file “SAM” geladen in registry explorer (te vinden in de triage map onder C\Windows\Systems32\config.) -> Dit hebben we geleerd in het deel “Accessing registry hives offline”.

Afbeelding met tekst

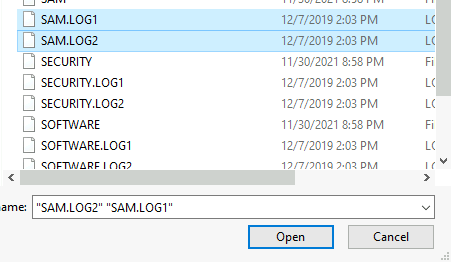
Automatisch gegenereerde beschrijving

Probleem: Tijdens het inlezen van het SAM bestand krijgen we een waarschuwing van diry hive.

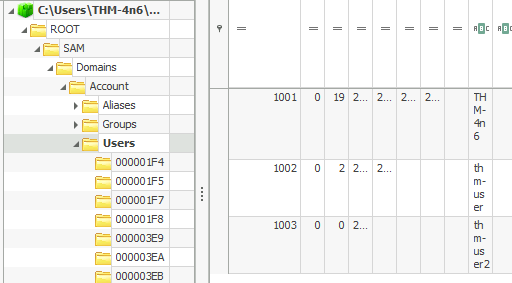
Afbeelding met tekst

Automatisch gegenereerde beschrijving

Oplossing: Omdat we een clean hive willen hebben, laden we de log files in i.p.v. de algemene SAM   
 hive. De LOG1 en LOG2 bestanden van SAM ingeladen om te vergelijken en een ‘clean’   
 bestand opgeslagen om dit dan te gebruiken. Daarna file opgeslagen op de desktop als   
 SAM\_CLEAN. De ID’s van de users gezocht in ‘Users’.



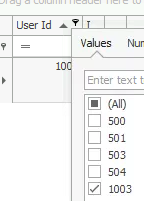
Afbeelding met tekst

Automatisch gegenereerde beschrijving 

#### Task 10.2

In de kolom last login time “empty” geselecteerd en daar de user met een id 10XX gezocht.

Afbeelding met tekst

Automatisch gegenereerde beschrijving Afbeelding met tafel

Automatisch gegenereerde beschrijving

#### Task 10.3

In de kolom “Password hint” staat count bij deze user, dus de hint is count.

Afbeelding met tafel

Automatisch gegenereerde beschrijving

#### Task 10.4

Probleem: Locatie vinden in Registry Explorer waar de info van “Changedlog.txt” staat.  
 NTUSER.DAT was comprised.

Oplossing: Na opzoek werk in het deel van “Usage of knowledge of files/folders” hebben we de NTUSER.DAT file ingeladen van het admin account (THM-4n6) en daarna de folder   
“recent docs” geselecteerd die we gevonden hebben in Software\Microsoft\Windows\CurrentVersion\Explorer\

De LOG1 en LOG2 bestanden van NTUSER genomen om te vergelijken. Vervolgens het bestand opgeslagen op de desktop als NTUSER.DAT\_clean.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Afbeelding met tafel

Automatisch gegenereerde beschrijving

Afbeelding met tekst

Automatisch gegenereerde beschrijving



#### Task 10.5

Na opzoekwerk in het deel van “Evidence of Execution” stond dat UserAssist een log bijhoudt van gestarte applicaties. We hebben we elke entry in software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\ doorzocht naar een path met ‘python-3.8.2’.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Afbeelding met tafel

Automatisch gegenereerde beschrijving

#### Task 10.6

Probleem: Waar kunnen we de info van de verbonden USB toestellen vinden?  
 De ‘timestamp’ van de laatste connectie van de gevonden USB was niet gegeven   
 in de map van Windows Portable Devices.

Oplossing: Op beide problemen hebben we de info gevonden in het deel van “External Devices/USB device forensics”.

We hebben dan de SOFTWARE hive van systems32 config file “Software” geladen om hier “windows portable devices” te raadplegen onder de map Microsoft. Dit gaf ons enkel het GUID van de USB die we zochten.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

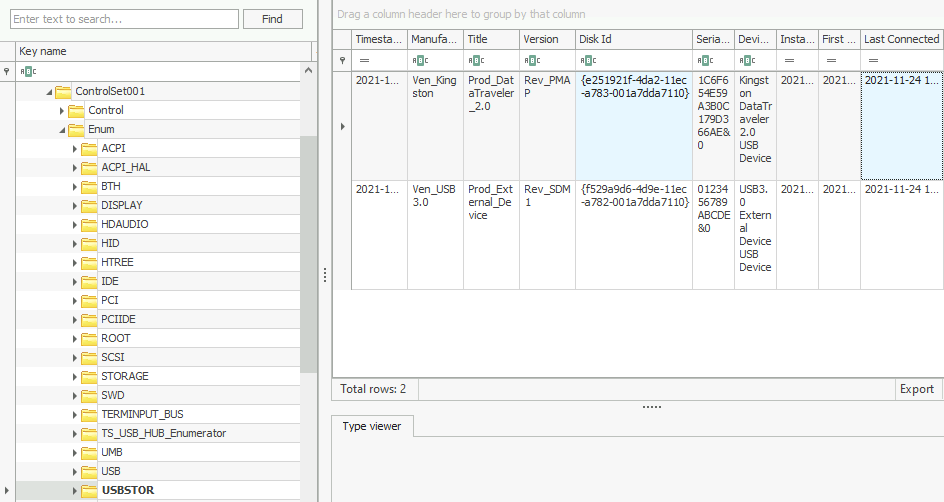
Afbeelding met tafel

Automatisch gegenereerde beschrijving

De SYSTEM hive te laden om te gaan kijken onder USB en USBSTOR. Hier stond de GUID van de USB die we nodig hebben met een kolom “last connected”

Afbeelding met tekst

Automatisch gegenereerde beschrijving



### Overzicht van afgewerkte room op TryHackMe



## **Juicy Details Star | Mario Wiki | FandomStar | Mario Wiki | Fandom**

Deze opdracht dient uitgevoerd te worden op <https://tryhackme.com/room/juicydetails> .

### Task 1

Inderdaad zijn we klaar om deze opdracht af te leggen.

### Task 2

#### Task 2.1

De hint zegt dat we in de access.log moesten zoeken. We zijn door heel de file gegaan (van boven naar onder) en alle tools die er gebruikt zijn genoteerd. Daarna opgezocht via google welke van deze tools door de aanvallers gebruikt worden, deze vergeleken met de resultaten uit de file en zo de oplossing gevonden.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Afbeelding met tekst

Automatisch gegenereerde beschrijvingAfbeelding met tekst

Automatisch gegenereerde beschrijving

Afbeelding met tekst

Automatisch gegenereerde beschrijving

#### Task 2.2

Probleem: Wat is een endpoint?

Oplossing: Google Search: “In een API worden deze gecategoriseerd als SOAP of REST. (REST   
 gebruikt ook GET, POST, PUT en DELETE).”

Omdat bruteforce attacks vaak gepaard gaan met dictionary attacks (uit de vorige vraag blijkt dat hydra hier gebruik van maakt) zijn we in de file access.log gaan kijken waar hydra voor gebruikt werd. Afbeelding met tekst

Automatisch gegenereerde beschrijving

#### Task 2.3

Uit vraag 1 hebben we geleerd dat ‘SQLmap’ gebruikt maakt van ‘sql-injection’. Dus zijn we gaan kijken naar de logs van ‘SQLmap’:



#### Task 2.4

In de resultaten van vorige vraag zien we dat er na “search” volgende wordt meegegeven -> ?q=

Hieruit kan je afleiden dat de parameter “q” gebruikt wordt.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

#### Task 2.5

Retrieving files doet ons denken aan files downloaden/overzetten van 1 plaats naar een andere zoals bij bv FileZille. Dit heeft een ftp protocol om de uitwisseling te vergemakkelijken. We hebben eens gekeken of we dit terug vonden in de file access.log.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

### Task 3

#### Task 3.1

De hint zegt dat we moeten gaan kijken daar waar klanten meestal commentaar over een winkel website kunnen achterlaten, dus wij dachten aan een review schrijven.

Door in de file access.log te kijken of we ergens het trefwoord “review” vonden stond dit bij “products”

Afbeelding met tekst

Automatisch gegenereerde beschrijving

#### Task 3.2

In de file kan je zien dat “hydra” probeert in te loggen door de POST /rest/user/login, dit via een brute force manier. Deze geven response codes 401 (unauthorized) en 500 (generic error) response. Maar één van deze pogingen geef een response code 200 (succeeded) terug.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Afbeelding met tekst

Automatisch gegenereerde beschrijving

#### Task 3.3

In access.log zien we veel GET /rest/products/search?=q… . Op een bepaald ogenblik is een ene GET met %20SELECT%20id,%20email,%20password in de code waaruit we kunnen afleiden dat de aanvaller probeert het email en password van de user te verkrijgen.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

#### Task 3.4

Zoals de vraag aangeeft moeten we gaan kijken naar de ftp uit vraag 5 van Task2. Bij deze resultaten zie je nog de details www-data.bak en coupons\_2013.md.bak staan waaruit je kan afleiden dat de aanvaller deze files probeerde te downloaden.

Afbeelding met tekst

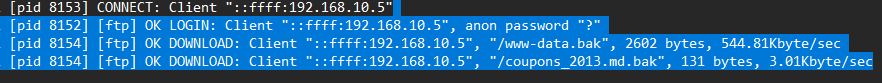
Automatisch gegenereerde beschrijving

#### Task 3.5

We hebben hier niets van gevonden in de file access.log dus hebben we eens gekeken in auth.log waar we enkel authenticatie gegevens terugvonden van user en wachtwoord en vonden we niets terug over de files uit de vorige vraag. In de file vsftpd.log kan je volgende code zien [anonymous] FAIL LOGIN: client waaruit we kunnen opmaken dat de aanvaller probeert om in te loggen met username anonymous. Nadat dit gelukt is zie je dat er terug gebruikt wordt gemaakt van ftp voor de bestanden uit vorige vraag.

Afbeelding met tekst

Automatisch gegenereerde beschrijving



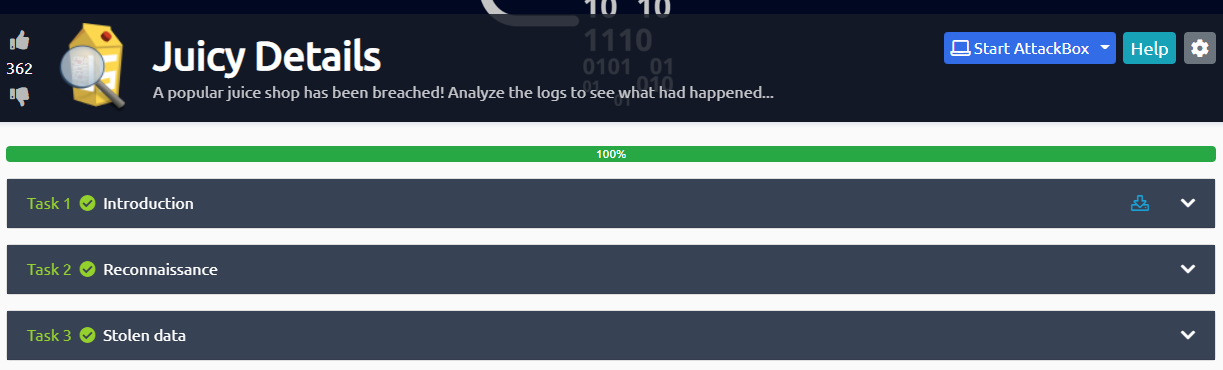
#### Task 3.6

In de file auth.log zie je op de eerste regel onmiddellijk Failed password for www-data from 192.168.10.5 port 40066 ssh2. Waaruit we kunnen afleiden dat het ingegeven passwoord voor www-data van ip adres 192.168.10.5 via poort 40066 over ssh2 mislukt is. Dus de aanvaller probeert het paswoord van account www-data via ssh verbinding te verkrijgen.

Afbeelding met tekst, oranje

Automatisch gegenereerde beschrijving

### Overzicht van afgewerkte room op TryHackMe



# **Yellow Team**

Iedere opdracht is terug te vinden als aparte branch op onze GitHub-repository. Dit om er zeker van te zijn dat reeds opgeloste oefeningen authentiek bleven.

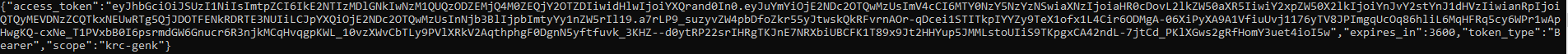
## Console Application Star | Mario Wiki | Fandom

In eerste instantie hebben we een eigen client toegevoegd aan de identity server met zijn eigen client\_secret.  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
  
Om een globale configuratie te creëren voor onze lokale applicatie hebben we deze credentials geplaatst in een aparte json file.   
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
  
We hebben daarna enkele methodes toegevoegd om allereerst het token op te gaan halen. Het eerste probleem waar we tegenaan liepen, was dat we onze baseURL hadden ingesteld als een secure URL, namelijk https. De error die hierdoor werd opgegooid was de volgende:  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
Aangezien het in deze opdracht nog niet direct de bedoeling was om de communicatie tussen de services via HTTPS te laten verlopen, hebben we er gewoon een simpel hypertekst transfer protocol van gemaakt.   
  
De methode GetToken gaat voor ons een WebRequest uitvoeren. Hier stellen we het content type en de methode in. We gaan namelijk aan de hand van een form onze credentials meegeven en een post uitvoeren op de URL die ons een token uitleent waarbij we ons kunnen authenticeren aan onze API. Hier hebben we wat research moeten doen om de form data op een correcte manier mee te geven in onze request. Het probleem waarbij we hier tegenaan liepen, was voornamelijk de encodering die we hadden toegepast, namelijk de ASCII encodering. We hebben hierbij wat hulp op het internet gezocht om te bepalen hoe we ons form correct konden meegeven. Een werkende oplossing vonden we door officiële Microsoft docs te raadplegen, namelijk:  
<https://docs.microsoft.com/en-us/dotnet/framework/network-programming/how-to-send-data-using-the-webrequest-class>  
  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
  
Eens we deze problemen hadden opgelost, was het enkel nog een kwestie om ons token uit de response te halen. Dit hebben we kunnen bekomen door onderstaande methode te gebruiken om het token uit de response te parsen. Het token zit omvat in een object waarbij het token de value is van de key ‘access\_token’.  
  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
  
Het bearerToken wordt via ‘GetTokenFromResponse’ opgeslaan in de class variable ‘bearertoken’, die we dan makkelijk kunnen aanspreken in ons programma. Dan konden we uiteindelijk ons Main programma uitschrijven. In de Main halen we eerst het token op door de methode ‘GetToken’, waarbij we onze globale ingestelde configuratie (credentials) meegeven.   
  
We laten de user een stad / postcode ingeven. Deze input geven we mee als parameter in de request, het endpoint naar onze seatholders. Om deze request te laten slagen, geven we het bearerToken mee in de headers.  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
  
We lezen de response uit tot het einde, met onderstaande screenshot als uiteindelijke resultaat:  
Afbeelding met tekst

Automatisch gegenereerde beschrijving

## Gebruik Auth0 Star | Mario Wiki | Fandom

We hebben voor deze opdracht gebruik gemaakt van Auth0, als tussenliggende service voor het uitlenen van een token en te kunnen authenticeren aan onze API. De stappen hiervoor waren zeer duidelijk, binnen Auth0 konden we een nieuwe API aanmaken, waarbij we een naam moesten instellen en een identifier.  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
  
  
De identifier hierbij was wel zeer belangrijk. Dit is iets wat we initieel mis hadden gedaan. Zoals te zien in de placeholder is het natuurlijk belangrijk om als identifier jouw eigen API mee te geven die je wilt benaderen. Dus in ons geval was dat <http://localhost:5000/api/>. Wij hebben onze API dan ook de naam ‘localhost:5000’ gegeven.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Eens deze API was opgezet, konden we een test uitvoeren om Auth0 tokens op te halen voor onze applicatie. We hebben hierbij gebruik gemaakt van Postman om deze verbinding uit te testen.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Het enige verschil hierbij was dat we bovenop de client\_id, de client\_secret en het grant\_type ook ons audience moesten meegeven. De data moest op dezelfde manier worden doorgegeven, namelijk als x-www-form-urlencoded. We kregen zo als response het token terug, uitgeleend door Auth0.

Afbeelding met tekst

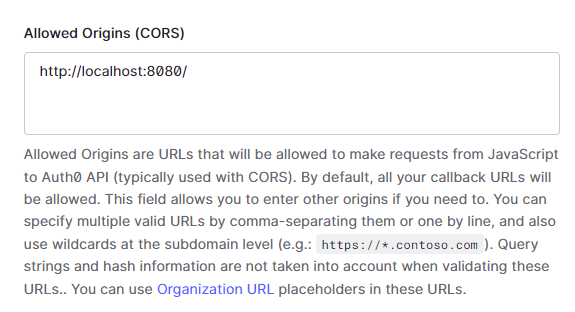
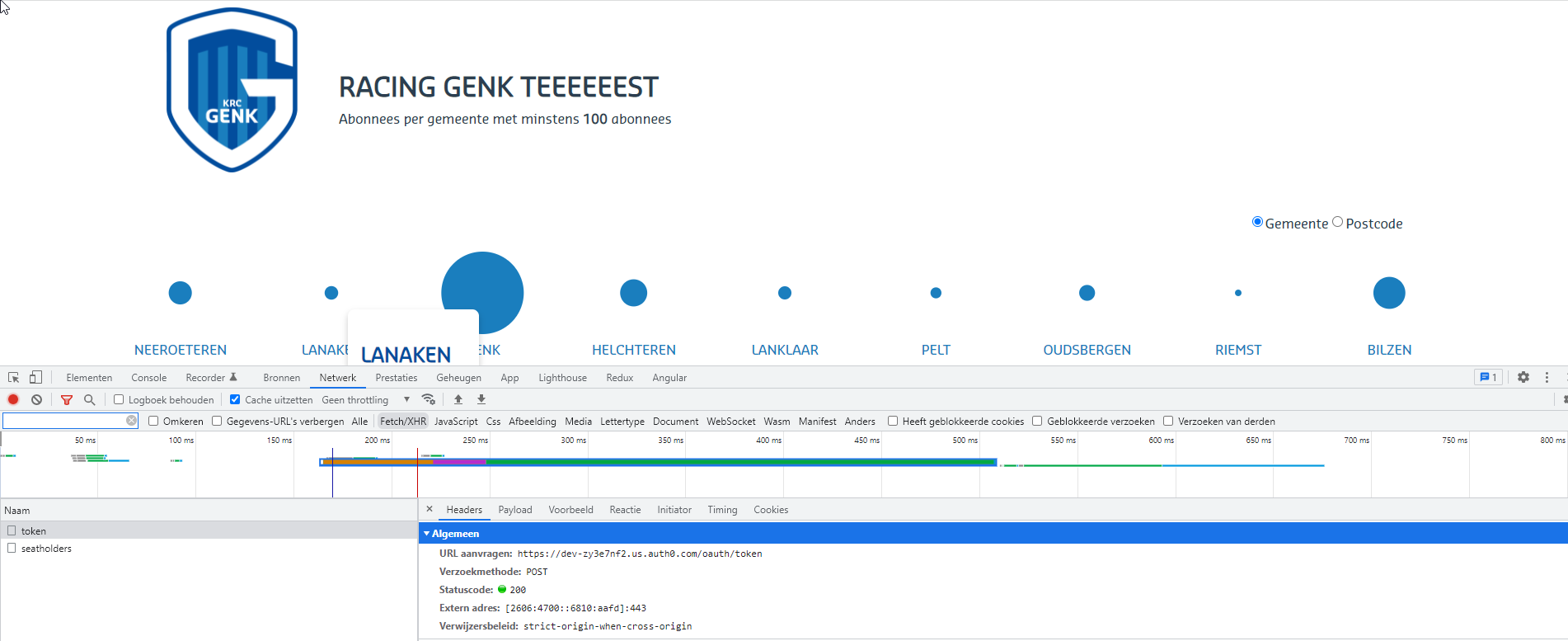
Automatisch gegenereerde beschrijving  
De eerste stappen op het platform van Auth0 waren zeer duidelijk en verliepen vlot. Maar eens we dit gingen implementeren in ons project, merkten we toch dat we enkele zaken nog moesten aanpassen. We hebben onder andere in Startup.cs de ‘ConfigureServices’ moeten aanpassen om nu gebruik te maken van de Auth0 Client Credentials Flow, in plaats van de flow van de identityserver. We hebben hierbij de manier van authentiseren moeten veranderen, alsook het authenticatieschema. Deze informatie konden we ook terugvinden onder het kopje ‘quickstart’ van onze API. Maar dit vergde toch wat meer uitzoekwerk.  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
Onze lokale console applicatie was vrij gemakkelijk om aan te passen, hierbij moesten we niet veel veranderen. Ons username en password werden nu respectievelijk het client\_id en het client\_secret van Auth0. De baseUrl moesten we ook nog veranderen naar het baseUrl van Auth0.  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
  
In de form data moesten we nog een extra veld toevoegen, namelijk het audience veld. Ons API endpoint van de resources die we graag willen benaderen. Afbeelding met tekst

Automatisch gegenereerde beschrijving

Het gebruik maken van de Auth0 Client Credentials Flow in onze console applicatie ging vrij vlot:  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
  
Maar de front-end werkte precies nog niet zo goed mee. Na wat opzoekwerk op het internet vonden we dat we binnen onze applicaties op het Auth0 platform ook Allowed Origins kunnen toevoegen. Eens dit in orde was, werkte de front-end ook vlot met deze flow.  
  
Zoals te zien in onderstaande screenshot maakt de front-end nu ook gebruik van de Auth0 Client Credentials Flow:   
  
We merkten naderhand dat de identityserver volledig overtollig was geworden. We hebben dan ook de docker-compose.yml naar behoren aangepast, zodat deze service niet meer hoefde mee te draaien in onze applicatie.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

## Eenvoudige MVC met Auth0 Star | Mario Wiki | FandomStar | Mario Wiki | Fandom

Dit was voor ons de meest uitdagende van de opdrachten binnen Yellow Team. Er waren hierbij twee grote struikelblokken die we ondervonden tijdens het uitwerken van deze opdracht:

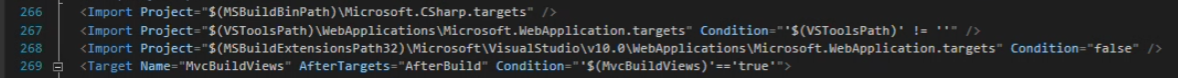
1. De verscheidenheid aan opties om gebruik te maken van OpenID Connect in combinatie met Auth0 en .NET.
2. Het meedraaien van de MVC app in docker.

We hadden besloten om verder gebruik te maken van Auth0 en een oplossing te zoeken om binnen Auth0 gebruik te maken van OpenID Connect binnen onze MVC app. We kwamen hierbij uit op verschillende bronnen en verschillende manier om dit te implementeren:  
  
a) <https://auth0.com/docs/authenticate/identity-providers/enterprise-identity-providers/oidc>  
b) <https://auth0.com/docs/get-started/applications/configure-applications-with-oidc-discovery>  
c) <https://auth0.com/docs/quickstart/webapp/aspnet-owin/01-login>  
d) <https://auth0.com/docs/quickstart/webapp/aspnet-core-2/03-authorization>  
  
We hebben testjes opgezet aan de hand van al deze bronnen. We hebben allereerst een Open ID Connect verbinding aangemaakt op het Auth0 platform (bron a):  
Afbeelding met tekst

Automatisch gegenereerde beschrijving

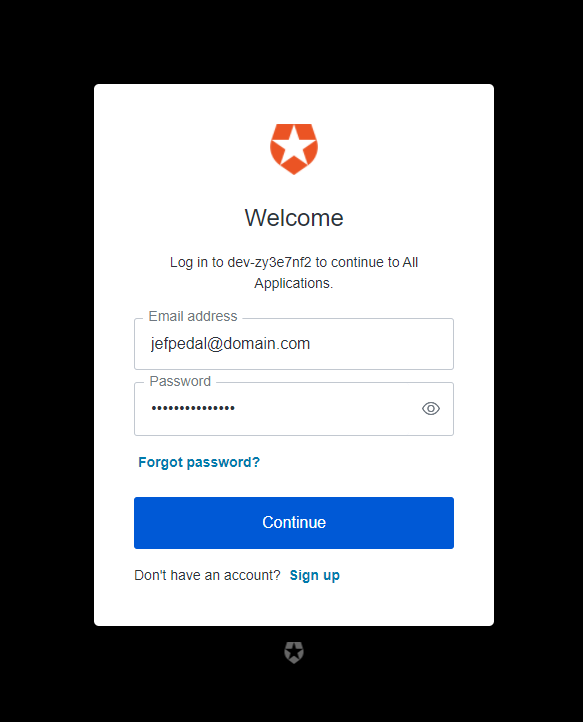
In een poging om deze OpenID connection te gebruiken in een MVC App hebben we de documentatie gevolgd in bron b. Aangezien deze documentatie het voorbeeld toont van een App\_Start\Startup.Auth.cs file, namen we aan dat dit moest geïmplementeerd worden in een .NET Entity Framework project. Tijdens het opzetten van deze test kwamen we dan uit bij de grote uitdaging om dit project laten mee te draaien in Docker.Afbeelding met tekst

Automatisch gegenereerde beschrijving

Het probleem hierbij was dat hij binnen de linux container ging zoeken naar een Microsoft bestand, namelijk Microsoft.WebApplication.targets. Zoals vermeld in de csproj file van dit Entity Framework project.

Na talloze uren zoeken en uittesten om toch te trachten dit project laten mee te draaien in Docker. Besloten we om opnieuw te beginnen en nieuwe testjes op te zetten. Zo kwamen we uit bij de laatste twee bronnen, waarbij bron d het dichtst aanleunde bij wat we nodig hadden. Dit voorbeeld toonde een simpele MVC app die gebruik ging maken van Auth0 OpenID Connect.  
  
We hadden ook reeds vastgesteld dat we binnen Auth0 users konden toevoegen aan een database die Auth0 voor ons beschikbaar stelt. Met behulp van deze users konden we ook authenticeren aan de hand van een Username-Password-Authentication, die standaard beschikbaar is binnen Auth0.

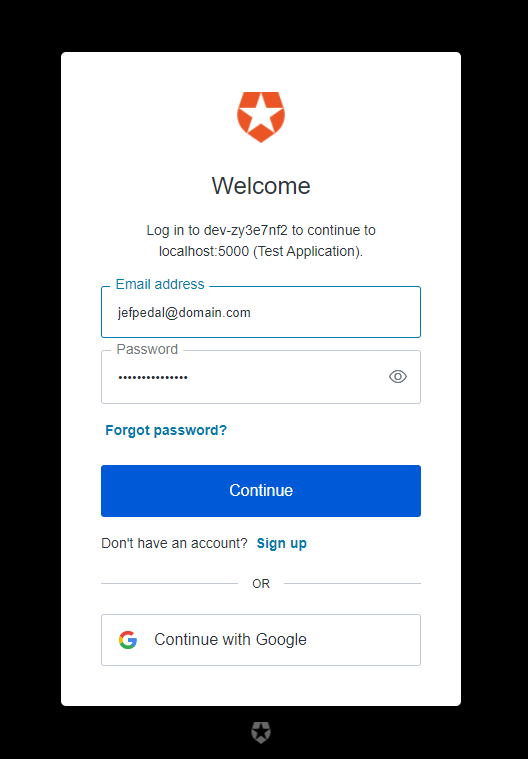
Afbeelding met tekst

Automatisch gegenereerde beschrijving

We merkten op dat dit project, in het voorbeeld van bron d precies gebruik maakte van deze openID connectie.

Afbeelding met tekst

Automatisch gegenereerde beschrijvingDoor deze connectie te implementeren in onze MVC applicatie, kwam hetzelfde inlogscherm tevoorschijn om te kunnen authentiseren met onze app.

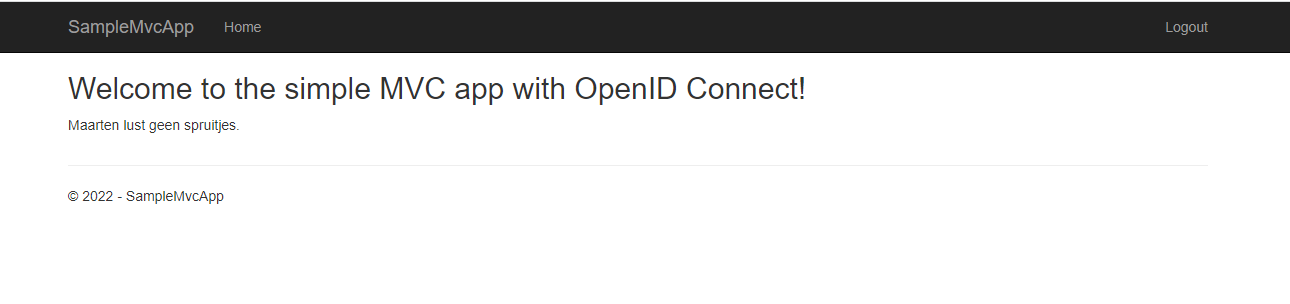


Pas na het inloggen, werden we verder doorgeleid naar onze applicatie (zoals ook te zien in de afbeelding: Log in to … to continue to localhost:5000 (onze Applicatie).  
  
Om deze verbinding volledig correct te leggen moesten we binnen Auth0 ook nog enkele zaken instellen binnen onze applicatie, zoals bijvoorbeeld de allowed callback URLs en allowed logout URLs. Aangezien onze MVC app draait op localhost:3000, moesten we deze URLs als callback en logout toevoegen.  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
Om de autorisatie volledig compleet te maken moesten we dan nog het Authorize attribuut toevoegen aan de HomeController, anders werd er niet gevraagd naar het externe inlogscherm van Auth0, omdat authenticatie natuurlijk niet nodig zou zijn.  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
  
Na het succesvol inloggen met behulp van het tussenliggend Auth0 inlogscherm konden we dan terug worden omgeleid naar onze MVC app, zoals gedefinieerd in de callback URL, en de message zien op het home scherm : ‘Maarten lust geen spruitjes.’

  
Daar bovenop draaide dit project ook meteen vlot mee in Docker.  
  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
  
Omwille van deze redenen hebben we uiteindelijk besloten om verder te gaan met de documentatie uit bron d.

## Uitbreiding MVC basic / advanced Star | Mario Wiki | FandomStar | Mario Wiki | FandomStar | Mario Wiki | Fandom

Om deze ster uit te voeren, konden we verder gaan op het voorbeeld uit bron d (ster 3).  
  
We hebben allereerst permissies toegevoegd aan onze Auth0 API:  
**Afbeelding met tekst

Automatisch gegenereerde beschrijving**  
  
We moesten ook deze instellingen aanzetten binnen onze API, om role based access control te activeren:  
**Afbeelding met tekst

Automatisch gegenereerde beschrijving**  
Omdat we enkel door deze stappen uit te voeren, er niet in slaagden om de permissies mee te sturen in het token besloten we om verder te werken met de rollen zoals vermeld in de documentatie.   
  
We maakten de rollen aan binnen Auth0 en wezen deze toe aan onze users:  
**Afbeelding met tekst

Automatisch gegenereerde beschrijving**  
  
Binnen de rollen kenden we de eerder aangemaakte permissies toe die we nodig hadden. Zoals we dit ook vaak zien in de praktijk. Autorisatie gebeurt vaak op basis van rollen, die eventueel ook geneste permissies kunnen bevatten, dus vonden we het een goed idee om hiermee verder te gaan.  
**Afbeelding met tekst

Automatisch gegenereerde beschrijving**  
**Afbeelding met tekst

Automatisch gegenereerde beschrijving**  
Jef Pedal kreeg de admin role toegewezen, met de advanced permission:  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
John Doe kreeg de User role toegewezen, met de basic permission:  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
  
Zo gingen we verder met de documentatie uit bron d, die ons vertelde dat we een rule moesten toevoegen binnen Auth0 zodat de rollen werden toegevoegd aan het Id Token om deze beschikbaar te stellen in onze backend.  
**Afbeelding met tekst

Automatisch gegenereerde beschrijving**  
  
Het was voor ons eerst onduidelijk waar we precies een rule konden aanmaken, dus volgden we ook nog volgende documentatie om ons even wegwijs te maken:  
<https://auth0.com/docs/customize/rules/create-rules>  
  
Daarna volstond het om gewoon een empty rule toe te voegen met de javascript code zoals vermeld in de documentatie:  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
  
We moesten hierbij in onze backend ook nog de OIDC authentication handler registration aanpassen om deze te informeren welke claim in het Token de informatie voor de rollen bevat:  
  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
  
  
  
Vervolgens was het enkel nog een kwestie om de Authorize attributen aan te passen in de HomeController en in de nieuw aangemaakt AboutController.  
  
De HomeController werd toegankelijk gemaakt voor iedereen, en de AboutController enkel voor Jef Pedal (de admin).  
  
Afbeelding met tekst, oranje

Automatisch gegenereerde beschrijving  
  
Afbeelding met tekst, oranje, schermafbeelding

Automatisch gegenereerde beschrijving  
  
Afbeelding met tekst

Automatisch gegenereerde beschrijving  
  
  
Afbeelding met tekst, schermafbeelding, monitor, elektronica

Automatisch gegenereerde beschrijving  
  
Afbeelding met tekst, schermafbeelding, elektronica, scherm

Automatisch gegenereerde beschrijving