

# TryHackMe — Linux Shells | Cyber Security 101 (THM)



Z3pH7 · [Follow](#)

9 min read · Oct 23, 2024



Listen



Share



More



**Hey everyone!** TryHackMe just announced the **NEW Cyber Security 101** learning path, and there are tons of giveaways this time! This article might help you out, but I've kept the summary short for easy understanding. Enjoy hacking!

## Introduction to Linux Shells

As regular users of operating systems, we all extensively use the Graphical User Interface (GUI) to carry out most operations. It takes a few clicks on different options, and your task is done. However, you can perform almost every task by writing commands in the CLI of your operating system rather than using the GUI. The shells give you some great features for the commands you write in your CLI. This way of interacting with the OS is more efficient and resource-friendly.

### Learning Objectives

- Learn interaction with Linux shell
- Use basic shell commands
- Explore the types of Linux shells available
- Write some shell scripts

Answer the questions below

---

*Who is the facilitator between the user and the OS?*

---

**Answer:** Shell

## How To Interact With a Shell?

Once the machine opens in the split-screen view, you will have the shell prompt ready to accept commands.

### LinuxShell

```
user@ubuntu:~$
```

Most Linux distributions use Bash (Bourne Again Shell) as their default shell. However, the default shell displayed when you open the terminal depends on your Linux distribution.

To see your current working directory, you can execute `pwd`, which stands for Print Working Directory, as shown in the terminal below:

## Check Current Working Directory

```
user@ubuntu:~$ pwd
/home/user
```

In the results of the above command, you can see that your current working directory is `/home/ubuntu`

However, you can change your directory as well. To do that, you can use `cd` (short for Change Directory), as shown in the terminal below:

## Change Directory

```
user@ubuntu:~$ cd Desktop
user@ubuntu:~/Desktop$
```

While using the GUI of an OS, you can see the contents of a directory on the screen. However, while using the shell, to see the contents of a directory, you must enter the following command:

## List Directory Contents

```
user@ubuntu:~$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos
```

If you want to read the contents of a file, you can type the following command in your shell:

## Displaying File Contents

```
user@ubuntu:~$ cat filename.txt  
this is a sample file  
this is the second line of the file
```

The grep command is a very popular command among Linux users. This powerful command can search for any word or pattern inside a file. Suppose you want to search for specific entries in a huge file. You can use the grep command along with the pattern of those entries, which will extract them for you. It also helps you to search for a specific keyword in a big file.

The following terminal shows us how to use the grep command to search for the word “THM” inside a big text file. The output displays the specific line of that text file containing this word.

### Searching a Word In File

```
user@ubuntu:~$ grep THM dictionary.txt  
The flag is THM
```

### Answer the questions below

*What is the default shell in most Linux distributions?*

**Answer:** Bash

*Which command utility is used to list down the contents of a directory?*

**Answer:** ls

*Which command utility can help you search for anything in a file?*

**Answer:** grep

### Types of Linux Shells

Linux distributions come with several types of shells, each with its own unique set of features. You can use the `echo $SHELL` command to see which shell you are

currently using. Additionally, the file `/etc/shells` lists all available shells on your system. Some popular shells include:

## Current Shell

```
user@ubuntu:~$ echo $SHELL
/bin/bash
```

You can also list down the available shells in your Linux OS. The file `/etc/shells` contains all the installed shells on a Linux system. You can list down the available shells in your Linux OS by typing `cat /etc/shells` in the terminal:

## Available Shells

```
user@ubuntu:~$ cat /etc/shells
# /etc/shells: valid login shells
/bin/sh
/bin/bash
/usr/bin/bash
/bin/rbash
/usr/bin/rbash
/bin/dash
/usr/bin/dash
/usr/bin/tmux
/usr/bin/screen
/bin/zsh
/usr/bin/zsh
```

To switch between these shells, you can type the shell name that is present on your OS, and it will open for you, as can be seen below:

## Switch Shell

```
user@ubuntu:~$ zsh
user@ubuntu ~ $
```

If you want to permanently change your default shell, you can use the command:  
`chsh -s /usr/bin/zsh`. This will make this shell as the default shell for your terminal.

There are many types of Linux shells. We will discuss a few of them and their features.

- **Bourne Again Shell (Bash):** Default in most Linux distributions, with powerful scripting capabilities and features like command history and tab completion.
- **Friendly Interactive Shell (Fish):** Focuses on ease of use with auto spell correction, customizable themes, and syntax highlighting.
- **Z Shell (Zsh):** Known for its extensive customization options and advanced tab completion features, but may run slightly slower due to its many features.

Answer the questions below

*Which shell comes with syntax highlighting as an out-of-the-box feature?*

**Answer:** Fish

*Which shell does not have auto spell correction?*

**Answer:** Bash

*Which command displays all the previously executed commands of the current session?*

**Answer:** history

## Shell Scripting and Components

Shell scripting allows users to automate repetitive tasks by combining several commands into a single executable file, which can be run at once. A basic shell script includes several key components:

**Shebang ( #! ):** Defines the interpreter used to run the script (e.g., `#!/bin/bash` for Bash).

1. **Variables:** Allow storing values that can be used throughout the script.
2. **Loops:** Repeatedly execute a set of commands.

### 3. Conditional Statements: Execute different code depending on certain conditions.

```
user@ubuntu:~$ nano first_script.sh
```

first\_script.sh

```
#!/bin/bash
```

We are all set to write our first script now. There are some fundamental building blocks of a script that together make an efficient script. Let's learn and utilize these script constructs to write one script ourselves.

#### Variables

A variable stores a value inside it. Suppose you need to use some complex values, like a URL, a file path, etc., several times in your script. Instead of memorizing and writing them repeatedly, you can store them in a variable and use the variable name wherever you need it.

The script below displays a string on the screen: "Hey, what's your name?" This is done by `echo` command. The second line of the script contains the code `read name`. `read` is used to take input from the user, and `name` is the variable in which the input would be stored. The last line uses `echo` to display the welcome line for the user, along with its name stored in the variable.

```
# Defining the Interpreter
#!/bin/bash
echo "Hey, what's your name?"
read name
echo "Welcome, $name"
```

To execute the script, we first need to make sure that the script has execution permissions. To give these permissions to the script, we can type the following

command in our terminal:

## Execution Permission to Script

```
user@ubuntu:~$ chmod +x variable_script.sh
```

## Script Execution

```
user@ubuntu:~$ ./variable_script.sh
Hey, What's your name?
John
Welcome, John
```

## Loops

```
# Defining the Interpreter
#!/bin/bash
for i in {1..10};
do
echo $i
done
```

The first line has the variable `i` that will iterate from 1 to 10 and execute the below code every time. `do` indicates the start of the loop code, and `done` indicates the end. In between them, the code we want to execute in the loop is to be written. The `for` loop will take each number in the brackets and assign it to the variable `i` in each iteration. The `echo $i` will display this variable's value every iteration.

Now, let's execute the script after giving it the execution permission.

## Script Execution

```
user@ubuntu:~$ ./loop_script.sh
1
```



2  
3

## Conditional Statements

Conditional statements are an essential part of scripting. They help you execute a specific code only when a condition is satisfied; otherwise, you can execute another code. Suppose you want to make a script that shows the user a secret. However, you want it to be shown to only some users, only to the high-authority user. You will create a conditional statement that will first ask the user their name, and if that name matches the high authority user's name, it will display the secret.

```
# Defining the Interpreter
#!/bin/bash
echo "Please enter your name first:"
read name
if [ "$name" = "Stewart" ]; then
    echo "Welcome Stewart! Here is the secret: THM_Script"
else
    echo "Sorry! You are not authorized to access the secret."
fi
```

Following is the terminal showing the script execution when the user name matches the authorized one defined in the script:

conditional\_script.sh

```
user@ubuntu:~$ ./conditional_script.sh
Please enter your name first:
Stewart
Welcome, Stewart! Here is the secret: THM_Script
```

However, the following terminal shows the script execution when the user name does not match the authorized one defined in the script:

conditional\_script.sh

```
user@ubuntu:~$ ./conditional_script.sh
Please enter your name first:
Alex
Sorry! You are not authorized to access the secret.
```

## Comments

```
# Defining the Interpreter
#!/bin/bash
```

```
# Asking the user to enter a value.
echo "Please enter your name first:"

# Storing the user input value in a variable.
read name

# Checking if the name the user entered is equal to our required
name.
if [ "$name" = "Stewart" ]; then

# If it equals the required name, the following line will be
displayed.
echo "Welcome Stewart! Here is the secret: THM_Script"

# Defining the sentence to be displayed if the condition fails.
else
    echo "Sorry! You are not authorized to access the secret."
fi
```

**Note:** Other types of variables, loops, and conditional statements can also be used to achieve different tasks. Moreover, multiple lines of comments can also be added within a single comment. However, it is not the scope of this room.

## Answer the questions below

*What is the shebang used in a Bash script?*

**Answer:** `#!/bin/bash`

*Which command gives executable permissions to a script?*

**Answer:** `chmod +x`

*Which scripting functionality helps us configure iterative tasks?*

**Answer:** Loops

## The Locker Script

In the previous task, we studied variables, loops, and conditional statements in shell scripting. Let's use that knowledge to create a shell script that utilizes all these components.

### Requirement

A user has a locker in a bank. To secure the locker, we have to have a script in place that verifies the user before opening it. When executed, the script should ask the user for their name, company name, and PIN. If the user enters the following details, they should be allowed to enter, or else they should be denied access.

- Username: John
- Company name: Tryhackme
- PIN: 7385

### Script

```
# Defining the Interpreter
#!/bin/bash
# Defining the variables
username=""
companyname=""
pin=""
# Defining the loop
for i in {1..3}; do
# Defining the conditional statements
    if [ "$i" -eq 1 ]; then
        echo "Enter your Username:"
        read username
    elif [ "$i" -eq 2 ]; then
        echo "Enter your Company name:"
        read companyname
    else
        echo "Enter your PIN:"
        read pin
    fi
done
# Checking if the user entered the correct details
if [ "$username" = "John" ] && [ "$companyname" = "Tryhackme" ] && [ "$pin" = "7385" ]
    echo "Authentication Successful. You can now access your locker, John."
```

```
else
    echo "Authentication Denied!!"
fi
```

## Script Execution

### Executing the Locker Script

```
user@ubuntu:~$ ./locker_script.sh
Enter your Username:
John
Enter your Company name:
Tryhackme
Enter your PIN:
1349
Authentication Denied!!
```

## Answer the questions below

*What would be the correct PIN to authenticate in the locker script?*

**Answer:** 7385

## Practical Exercise

We have placed a script on the default user directory `/home/user` of the attached Ubuntu machine. This script searches for a specific keyword in all the files (with `.log` extension) in a specific directory.

**Note:** Some changes are required inside the script file before you execute it. When you open the machine according to the instructions in task #2, you will be able to gain the session as a normal user. However, we recommend you to become the root user in order to search for the flag in all the files of the given directory. To become one, you only need to type the following command and enter the password of the user:

### Become Root User

```
user@ubuntu:~$ sudo su
[sudo] password for user:
root@tryhackme:/home/user#
```

You can make the changes in the script file by keeping in view the following details:

- **Flag:** thm-flag01-script
- **Directory:** /var/log

**Hint:** Look for empty double quotes " " inside the script file and fill them. Make sure not to leave any space between them.

**Answer the questions below**

*Which file has the keyword?*

**Answer:** authentication.log

*Where is the cat sleeping?*

**Answer:** under the table

```
#!/bin/bash
# nano flag_hunt.sh
# Defining the directory to search our flag
directory="/var/log"

# Defining the flag to search
flag="thm-flag01-script"

echo "Flag search in directory: $directory in progress..."

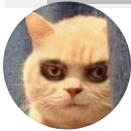
# Defining for loop to iterate over all the files with .log extension in the de
for file in "$directory"/*.log; do
    # Check if the file contains the flag
    if grep -q "$flag" "$file"; then
        # Print the filename
        echo "Flag found in: $(basename "$file")"
```

```
fi  
done
```

```
./flag_hunt.sh
```

```
grep "cat" /var/log/authentication.log
```

Thank you!

[Tryhackme](#)[Tryhackme Walkthrough](#)[Cyber Security 101](#)[Linux](#)[Shell](#)[Follow](#)

## Written by Z3pH7

234 Followers · 7 Following

Cybersecurity | Pentester | Student

## Responses (4)



What are your thoughts?

[Respond](#)



Samar

2 months ago



What is the default shell in most Linux distributions?

> Bash



1



1 reply

Reply



Nsikakabasi David Eteudoh

2 months ago



Thank you very much. You helped me with solving the Practical exercise. I appreciate it.



1

Reply



Miac

about 2 months ago



Dumb question, what the pass for user at Practical Exercise?




1 reply

Reply

[See all responses](#)

## More from Z3pH7

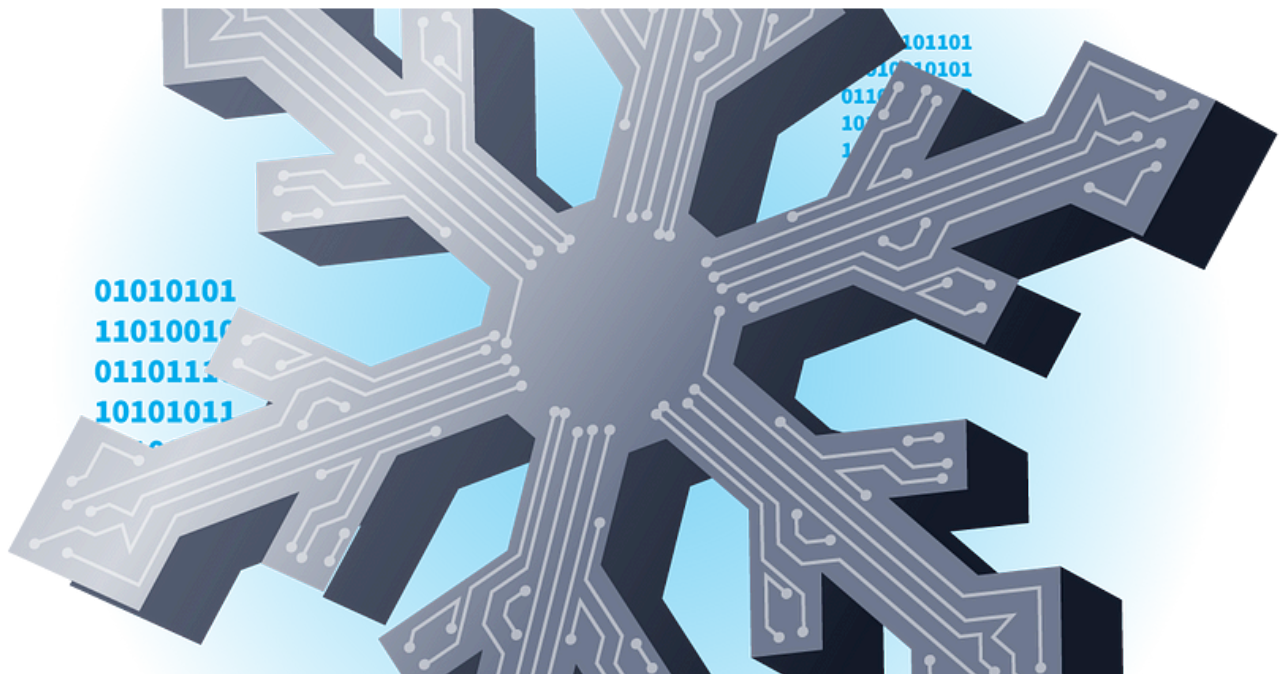


 Z3pH7

## TryHackMe—Windows PowerShell | Cyber Security 101 (THM)

Hey everyone! TryHackMe just announced the NEW Cyber Security 101 learning path, and there are tons of giveaways this time! This article...

Oct 23, 2024  102



 Z3pH7

## TryHackMe—Hashing Basics | Cyber Security 101 (THM)

Hey everyone! TryHackMe just announced the NEW Cyber Security 101 learning path, and there are tons of giveaways this time! This article...



Oct 29, 2024 🖱 101 💬 1



Z3pH7

## TryHackMe—Networking Core Protocols | Cyber Security 101 (THM)

Hey everyone! TryHackMe just announced the NEW Cyber Security 101 learning path, and there are tons of giveaways this time! This article...

Oct 25, 2024 🖱 11



Z3pH7

# TryHackMe—Public Key Cryptography Basics | Cyber Security 101 (THM)

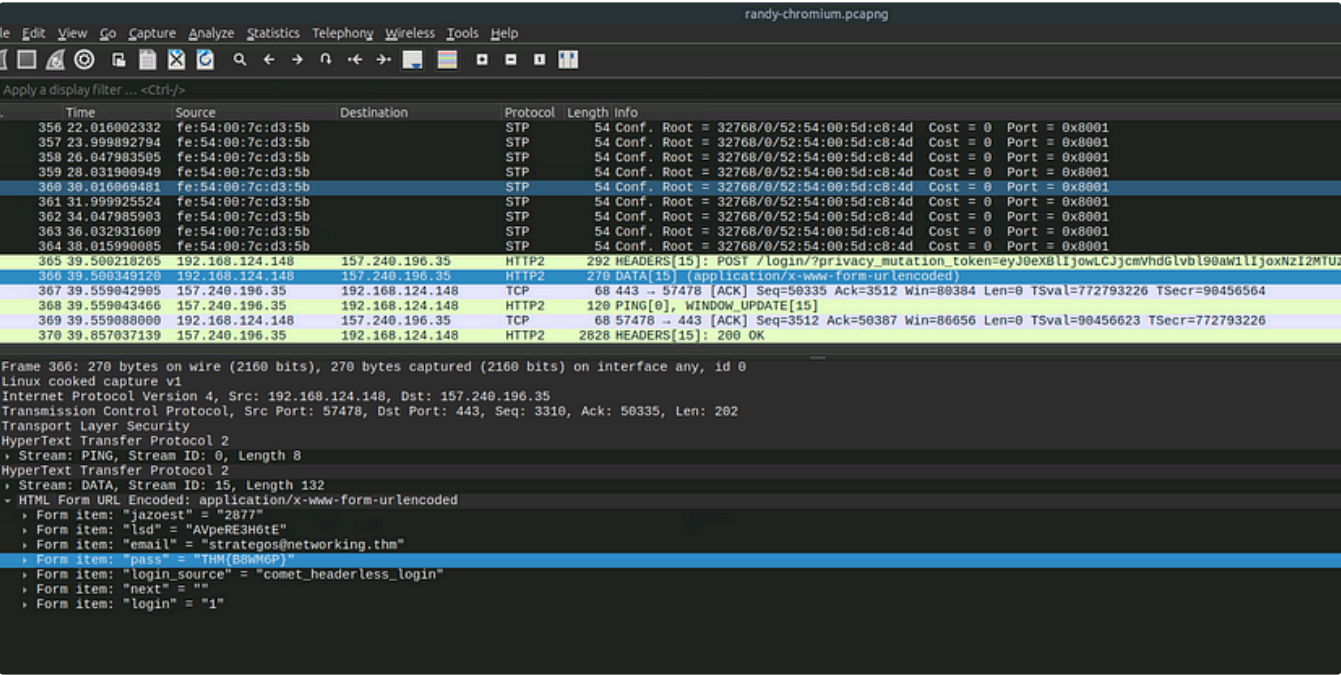
Hey everyone! TryHackMe just announced the NEW Cyber Security 101 learning path, and there are tons of giveaways this time! This article...

Oct 28, 2024 75



See all from Z3pH7

## Recommended from Medium



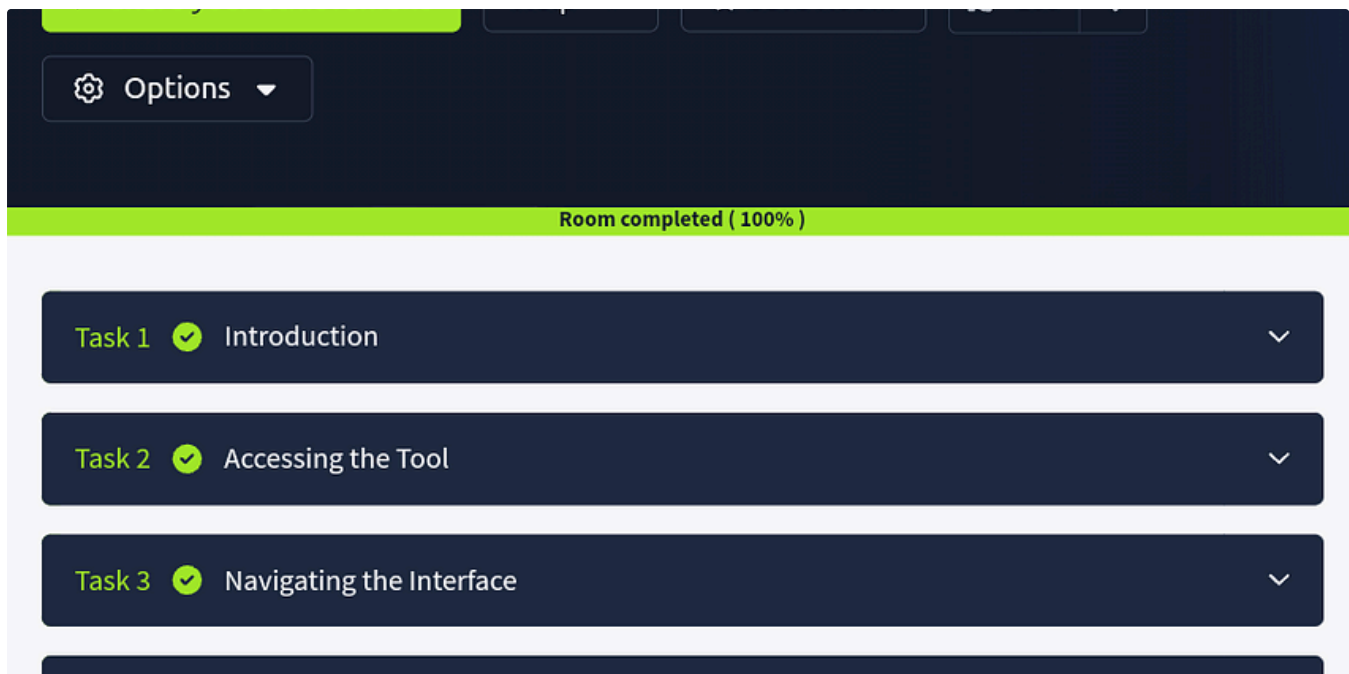
embossdotar


## TryHackMe—Networking Secure Protocols—Writeup

Key points: TLS | SSH | VPN | Network traffic | HTTPS | POP3S | IMAPS | SMTPS. Networking Secure Protocols by awesome TryHackMe! 🌈

Oct 22, 2024 3





 Jawstar

## CyberChef: The Basics Tryhackme Write up

Tryhackme

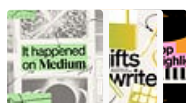
★ Nov 7, 2024 🖱 8



Open in app ↗


# Medium

🔍 Search



### Staff picks

791 stories · 1538 saves

 Z3pH7

## TryHackMe—Windows PowerShell | Cyber Security 101 (THM)

Hey everyone! TryHackMe just announced the NEW Cyber Security 101 learning path, and there are tons of giveaways this time! This article...

Oct 23, 2024  102

 IritT

## Networking Core Protocols—Cyber Security 101—Networking—TryHackMe Walkthrough

Learn about the core TCP/IP protocols.

Oct 26, 2024



Nikhil Bhandari

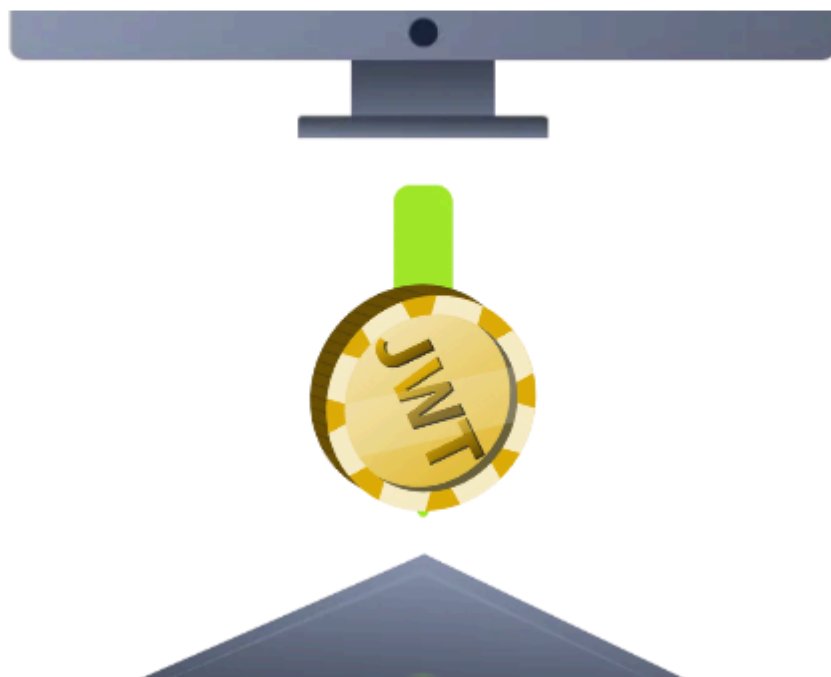
## TryHackMe — Networking Concepts | Cyber Security 101 (THM)

Hey everyone, Nikhil Bhandari here! TryHackMe has just launched their NEW Cyber Security 101 learning path, and they've got plenty of...

Oct 23, 2024



19



Sudarshan Patel

## Tryhackme | JWT Security | Writeup

Learn about JWTs, where they are used, and how they need to be secured.

Oct 14, 2024  11



See more recommendations