



# Commanding the Command Line

Cybersecurity

Terminal 101 Day 2



# Class Objective

---

By the end of today's class, you will be able to:



Identify and explain the structure of a terminal command.



Explain how options modify the default behavior of a terminal command.



Use the `man` command to list instructions and options for each command



Use the `find` command to locate files based on search parameters.



Use the `grep` command to search within the contents of files.



Use the `wc` command to count words and lines.



Combine multiple commands in sequence with pipes to accomplish intermediate IT tasks.

The background of the slide is dark gray with a pattern of diagonal lines that create a sense of depth and movement, resembling a series of overlapping planes or a stylized architectural design.

# Welcome Back to the Terminal

# Terminal Review

---

Commands for navigating a file directory:

<code>pwd</code>	
<code>ls</code>	
<code>cd</code>	
<code>cd ../</code>	
<code>clear</code>	

# Terminal Review

---

Commands for navigating a file directory:

<code>pwd</code>	Display the current working directory.
<code>ls</code>	List the directories and files in the current directory.
<code>cd</code>	Navigate into a directory.
<code>cd ../</code>	Navigate out of a directory.
<code>clear</code>	Clear the terminal history on the page.

# Terminal Review

---

Commands for making and removing files and directories:

<code>mkdir</code>	
<code>rmdir</code>	
<code>touch</code>	
<code>rm</code>	

# Terminal Review

---

Commands for making and removing files and directories:

<code>mkdir</code>	Create a new directory.
<code>rmdir</code>	Remove a directory.
<code>touch</code>	Create an empty file.
<code>rm</code>	Remove a file.

# Terminal Review

---

Commands for moving and copying files:

mv	
cp	



# Terminal Review

---

Commands for moving and copying files:

<code>mv</code>	Move files to a new location.
<code>cp</code>	Copy files into a new location.

# Terminal Review

---

Commands for previewing files:

more	
less	
head	
tail	

# Terminal Review

---

Commands for previewing files:

<code>more</code>	Display a file one page at a time.
<code>less</code>	Display a file, with the ability to scroll up and down.
<code>head</code>	Preview the top 10 lines of a file.
<code>tail</code>	Preview the bottom 10 lines of file.

# Terminal Review

---

Commands for concatenating and redirecting files:

cat	
>	
>>	

# Terminal Review

---

Commands for concatenating and redirecting files:

cat	Concatenate and combine files together.
>	Write to a file, overwriting the file if the file name already exists.
>>	Write to a file, appending the file if the file name already exists.



## Activity: Warm Up

In this activity, you will continue in your role as security analyst at Candy Corp.

- Local authorities found video evidence that Sugar Corp made a cash delivery to Candy Corp's back door on October 13th, 2019.
- You must gather physical access logs to prove Henry or Ruth opened the back door for the delivery.

Suggested Time:

15 Minutes



Time's Up! **Let's Review.**

# Questions?





# Command Line Structure

# Command Line Structure

---

So far, we are familiar with the following structure:

```
<command> <argument>
```

Example:

```
touch myfile
```

- The command is **touch**.
- The argument is **myfile**.

# Expanding the Command Line

---

IT professionals often need to run commands with more specific parameters than can be included in the command itself.



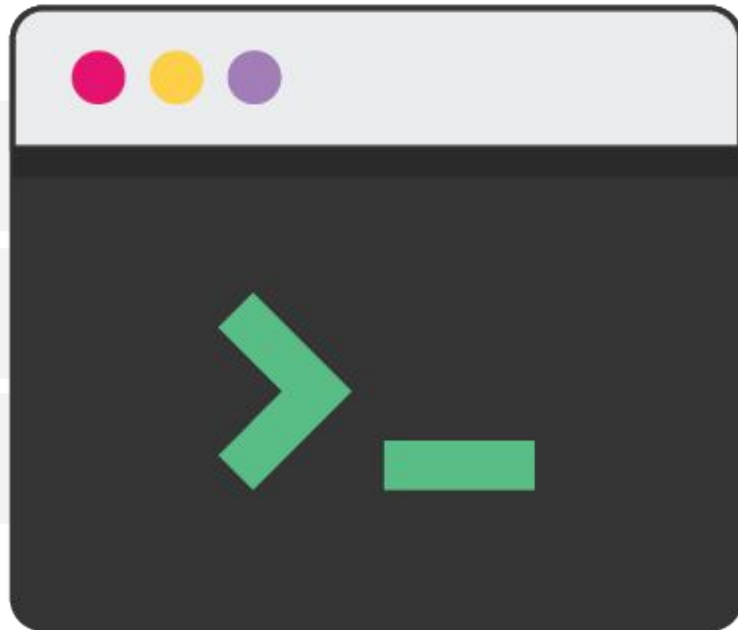
You need to clean up server space.



You want to list out files by size, so you can delete the largest files first.



But the default behavior of the **ls** command does not list out the files by size.



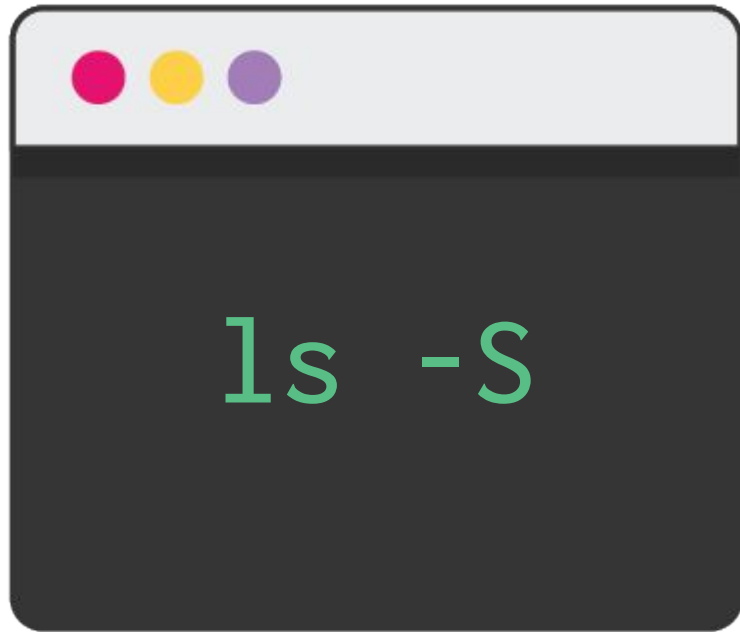
# Method One

---

Add an option to modify the command's default behavior.

We can use the command `ls -S` to list files by size.

- By default, `ls` simply lists out the files in the current directory. The `-S` option modifies that behavior to list by size, largest first.
- The syntax for the preceding command:
  - `ls` is the command.
  - `-S` is the option.



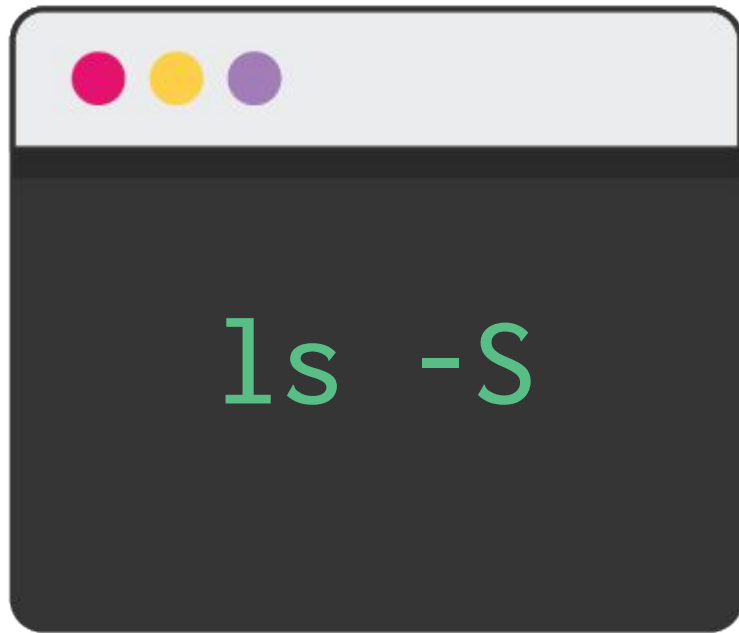
# Method One

---

## Keep in mind:

Add an option to modify the command's default behavior.

- Options always use a hyphen.
- Options, just like commands, are case sensitive:
  - `-s` provides a different result than `-S`.
  - The lowercase `-s` option prints the size of each file.
- Options have different uses for different commands.
  - `ls -s` will print the size of each file.
  - `cat -s` will suppress repeated empty output lines.

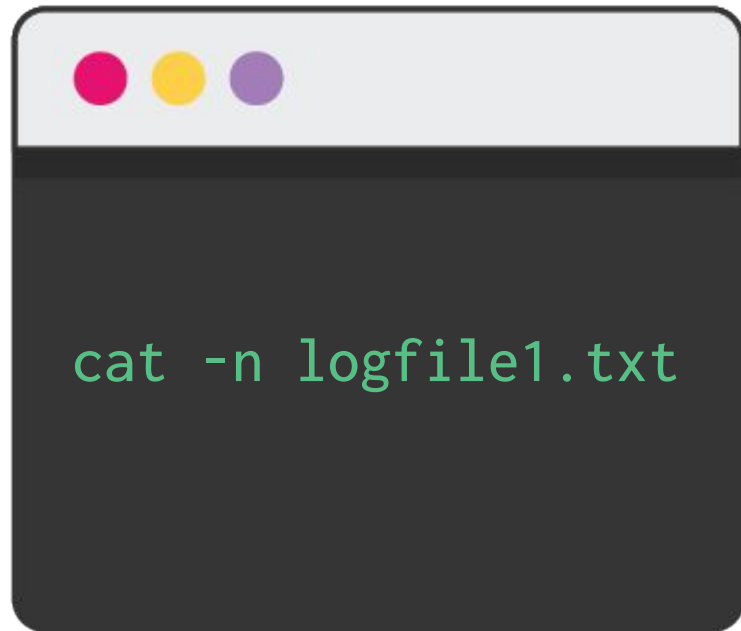


# Method Two

---

Add an option and an argument to modify the default behavior.

- This command displays the line numbers of `logfile1.txt`.
- By default, the `cat` command concatenates multiple files or displays the contents of a single file.
- Adding the option `-n` modifies the behavior by displaying the line numbers preceding each line.



# Method Three

---

Add options that require their own arguments, called **parameters**.

- This command previews the first four lines of `logfile1.txt`.
- By default, **head** displays the top ten lines of a file.
- The option `-n` changes the number of lines displayed.
  - `-n` requires a **parameter** specifying the number of lines.
  - Parameters provide additional details on how to modify a command's default behavior.

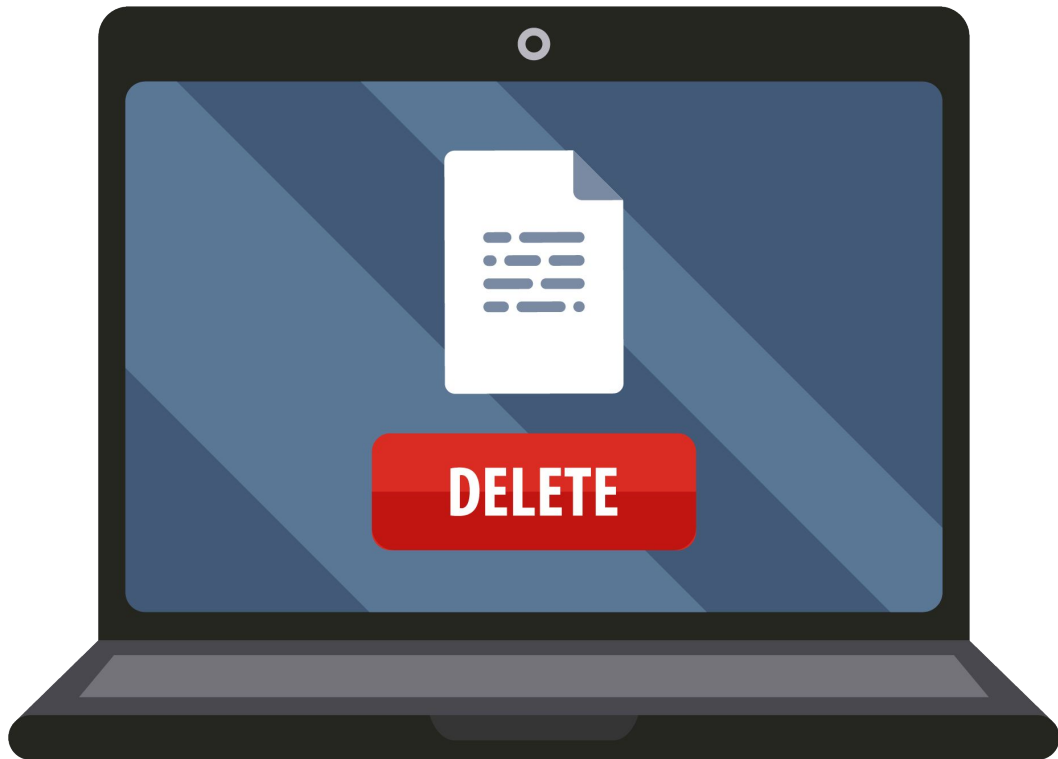


# Demo Scenario: Options

---

We are security analysts at ACME Corp. Our manager has tasked us with cleaning up some evidence files, as server space is getting low.

- We need to delete the three largest evidence files, as long as they don't contain the user **Sheila**, which will be needed for a future investigation.
- We've been told that the log files are not more than 40 lines.





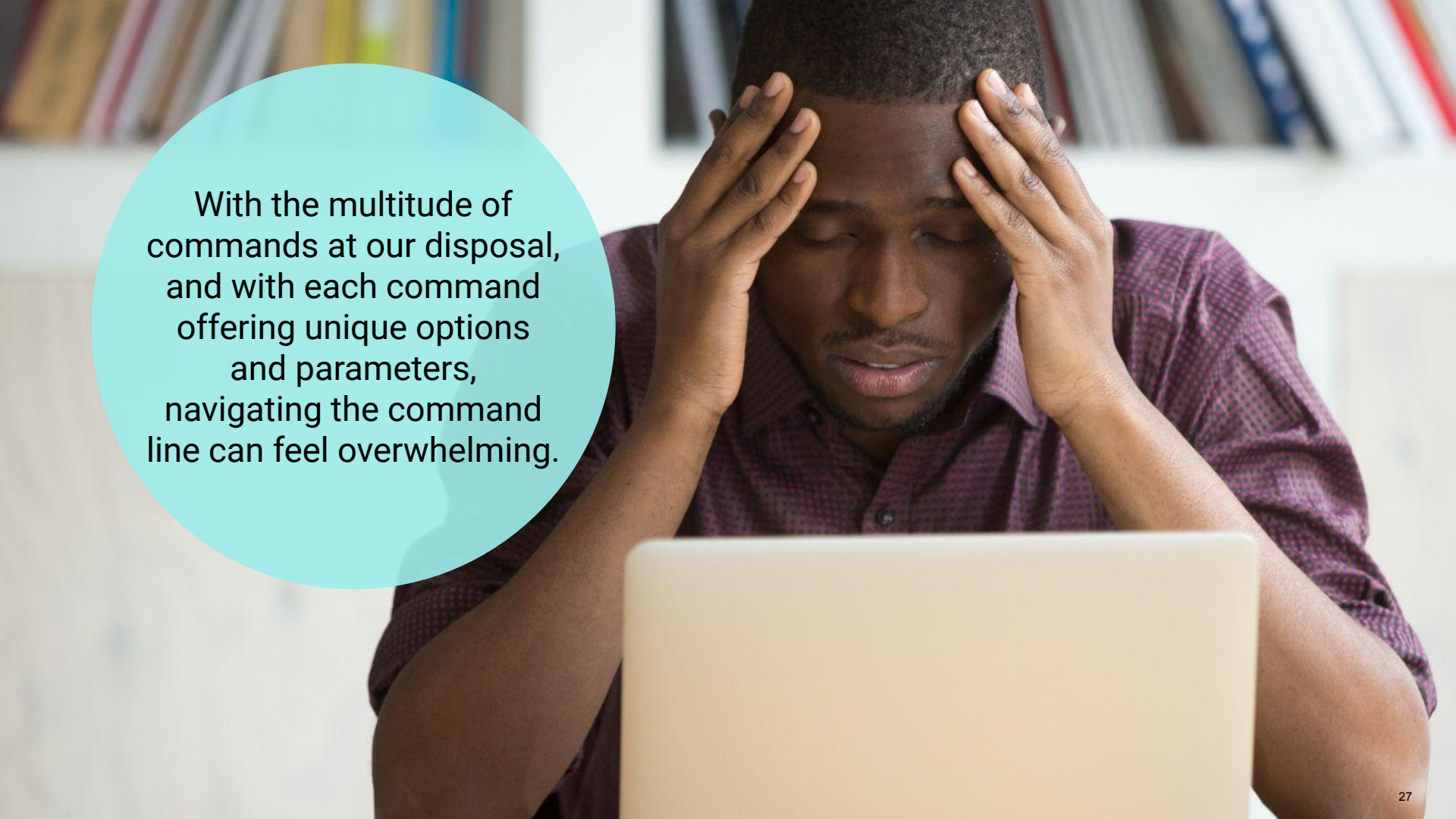


# Instructor Demonstration

---

## Options

# Manual Pages



With the multitude of  
commands at our disposal,  
and with each command  
offering unique options  
and parameters,  
navigating the command  
line can feel overwhelming.



IT and security professionals  
can learn and manage options  
with a valuable resource known  
as **manual (man) pages.**

# Welcome to Man Pages

**man** pages are built-in to the terminal and document the following for each command:

- Name of the command
- Synopsis (includes syntax)
- Description
- Options and option parameters

Access man pages with the following:

```
man <command>
```

For example:

```
man ls
```

LS(1)

BSD General Commands Manual

LS(1)

## NAME

**ls** -- list directory contents

## SYNOPSIS

**ls** [-ABCFGHLOPRSTUW@abcdefghijklmnopqrstuvwxyz] [file ...]

## DESCRIPTION

For each operand that names a file of a type other than directory, **ls** displays its name as well as any requested, associated information. For each operand that names a file of type directory, **ls** displays the names of files contained within that directory, as well as any requested, associated information.

If no operands are given, the contents of the current directory are displayed. If more than one operand is given, non-directory operands are displayed first; directory and non-directory operands are sorted separately and in lexicographical order.

The following options are available:

- e Display extended attribute keys and sizes in long (-l) output.
- 1 (The numeric digit ``one''.) Force output to be one entry per line. This is the default when output is not to a terminal.
- A List all entries except for . and ... Always set for the super-user.
- a Include directory entries whose names begin with a dot (.).
- B Force printing of non-printable characters (as defined by ctype(3) and current locale settings) in file names as \xxx, where xxx is the numeric value of the character in octal.



# Instructor Demonstration

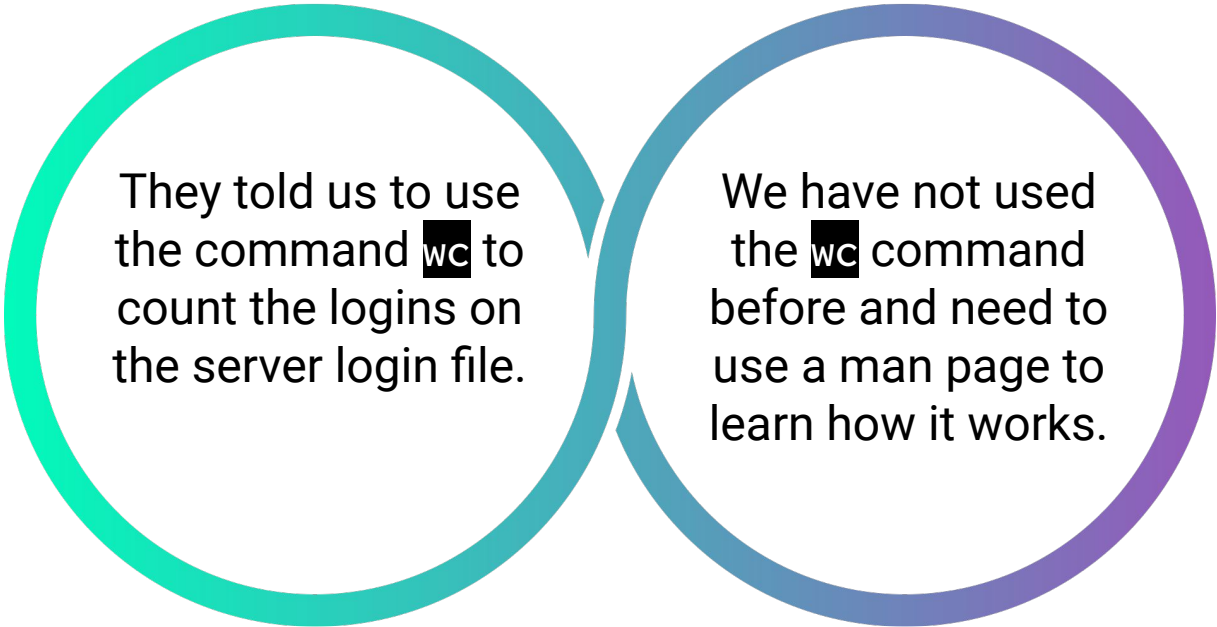
---

## Man Pages

# Demo Scenario: Man Pages

---

In this demo, we are a security analyst at ACME Corp. Your manager has tasked you with counting the number of logins on a server on the day of October 13, 2019.



They told us to use the command `wc` to count the logins on the server login file.

We have not used the `wc` command before and need to use a man page to learn how it works.



## Activity: Learning New Commands

You continue in the role of security analyst at Candy Corp.

- There has been a network attack on Candy Corp's websites and management needs your help determining which website was the main target.
- To determine which website was the main target, you will count the IP addresses in log files provided by your manager.

Suggested Time:

---

18 Minutes





Time's Up! **Let's Review.**

# Questions?



# The `find` Command

# find

---

We have previously navigated in and out of multiple directories to find files or directories.

This isn't always the best practice.

## For example:

- You might be tasked with finding access logs on a server that you're not familiar with.
- Security professionals are often not provided the exact location of a file, so you may have to navigate through hundreds of directories to find the access logs.

# Introducing the `find` Command

---

The `find` command searches for files and directories with one command.



**WILL**

By default, `find` will search through the current directory and the subdirectories within that current directory.



**WILL NOT**


However, `find` does not review the contents within a file, only the file name or directory name.

# find Syntax

---

This command finds **all** files in our current directory and its subdirectories.

```
find -type f
```



We use the option **-type** and the required parameter **f** to indicate that we are searching for files.

# find Syntax

---

This example will find a **specific** file.

```
find -type f -name log.txt
```



We use the option **-name** to search for an exact match of the specified parameter, **log.txt**.

# find Syntax

---

To find a specific file with case insensitivity, we change the `-name` option to `-iname`.

```
find -type f -iname log.txt
```

This example will find the files called **log.txt** (lowercase) or **LOG.TXT** (uppercase) in our current directory and its subdirectories.



# find Syntax

---

This example uses a symbol known as a **wildcard** to search for all files ending with `.txt`.

The `*` wildcard symbol indicates that any file ending with `.txt` will be displayed, regardless of what comes before `.txt`.

```
find -type f -iname `*.txt`
```

Using wildcards with `find` is called a **wildcard search**.

With the addition of `-iname`, this example finds all files that end with `.txt` or `.TXT` in your current directory and its subdirectories.

# find Syntax

---

In the final example, we're using `find` to search for a file located in **another** directory.

Specifically, we're looking for the case insensitive `log.txt` in the `/root/desktop` directory.

```
find /root/desktop -type f -iname `log.txt`
```

We place the desired directory after the `find` command and before the the `-type` option.

# Syntax for Finding Directories

---

We find directories with the same syntax, but add `-d` as the `type` parameter.

```
find -type d
```

```
find -type d -name logs
```

```
find -type d -iname logs
```

```
find -type d -iname '*1013*'
```

```
find /root/desktop -type d -iname logs
```

# find Demo Setup

---

In the next demonstration, we'll use the following scenario:



Your manager at ACME has tasked you with finding logs for a certain type of web server called **Apache**, for the date of **October 13th**.



They told you that the directory should be named **apache** and the log files should have the date noted as **1013** in their file names.



Since there are many directories, you will use the **find** command to complete these tasks.

---



# Instructor Demonstration

---

find

# Demo Review: **find**ing your Way

---

The previous demo covered the following concepts:

<b>find:</b>	Command-line command used to locate a file or a directory.
<b>type f:</b>	An option used to find files.
<b>type d:</b>	An option used to find directories.
<b>name:</b>	Additional option used to find specific file of directory names.
<b>iname:</b>	Additional option used to find case-insensitive names.



## Activity: **finding** Your Way

Your manager at Candy Corp has tasked you with searching through the **PeanutButtery.net** server's files and file directories to uncover secret recipes they believe are hidden in the file system.

Suggested Time:

---

18 Minutes



Time's Up! **Let's Review.**



# Questions?





 Countdown timer

15:00

(with alarm)

grep

# Limitations of find

---

The `find` command only searches for the names of files, *not* the contents within.

However, security professionals are often tasked with searching for specific data inside a file.

## For example:

You might be asked to check if a specific user logged in on a specific day.

You would first find the access log file for that day and then need to verify if that user appeared in the log file.

# grep

---

We've used preview commands such as `head`, `more`, `tail`, and `less` to display the contents of a file. But these commands are limiting.



Large files take a lot of time to scan for data.



If you have more than one file to scan, it can take a lot of time to preview all of them.



Manually previewing and scanning files invites human error.

---

# grep

---

We can use **grep** to search a file or multiple files for a specific data point.

01

**grep** (*global regular expression print*) is a command to search for data inside of files.

02

**grep** by default returns the entire line that the desired data is found in.

03

**grep** by default will only search for data in the current directory, not subdirectories.

---

# grep Syntax

---

In this example, we are using `grep` to find the lines in which user **bob** is mentioned in the file `log1.txt`.

```
<grep data_point File_to_search_inside>
```

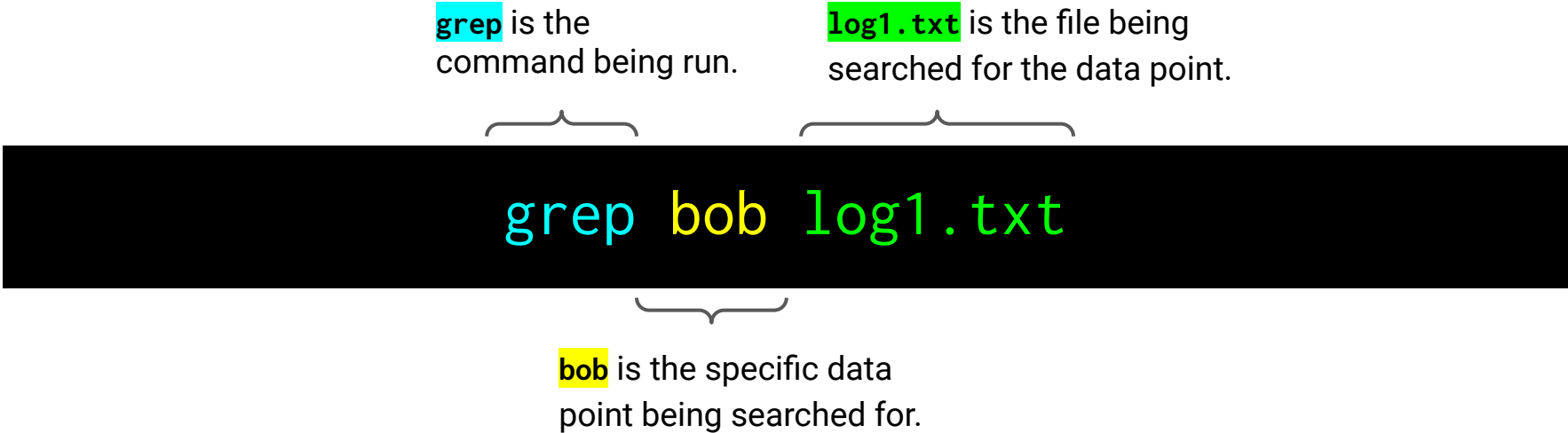
# grep Syntax

---

In this example, we are using `grep` to find a specific data point within a single file. The syntax is:

`grep` is the  
command being run.

`log1.txt` is the file being  
searched for the data point.



```
grep bob log1.txt
```

The diagram illustrates the syntax of the `grep` command. It features a black rectangular box containing the command `grep bob log1.txt`. Above the box, two annotations with curly braces identify the components: `grep` is the command being run, and `log1.txt` is the file being searched. Below the box, a third annotation with a curly brace identifies `bob` as the specific data point being searched for.

`bob` is the specific data  
point being searched for.

This command will display all the lines where the data point `bob` was found inside of the file `log1.txt`. If no matches of `bob` are found in the file, nothing will be returned.



# grep Syntax

---

In this example, we are using **grep** to find a specific data point within **multiple** files.

```
grep bob *.txt
```

We are using **grep** to find where **bob** exists within in all **.txt** files.

- **bob** is the specific data point being searched for.
- **\*.txt** is the wildcard. **\*** indicates that it will search through all files that end with **.txt**.

The command will display all the **.txt** files where the value of **bob** was found, followed by the lines where it was found.



Can anyone tell us what this  
command does?

```
grep -i bob *.txt
```

# grep Syntax

---

This `grep` command is used to find a **case-insensitive** specific data point.

```
grep -i bob *.txt
```

Specifically, this command finds the lines where the user **bob** or **BOB** exists within all `.txt` files.

# grep Syntax

---

This example uses `grep` to indicate that it should display the *names* of the files that contain the specified data point. When the command is run, it will only display the names of the file that contain **bob**.

```
grep -il bob *.txt
```

Specifically, this command outputs the file names of `.txt` files containing the user **bob** or **BOB**. It will only display the *name*.

**Note:** `-il` are two separate options.

- `i` is an option for `grep` indicating case insensitivity.
- `l` is an additional option indicating to only return the file name.
- `i` and `l` use a single hyphen.

# grep Demo Setup

---

In the next demonstration, we'll use the following scenario:



Your manager has now asked for your help with a security investigation into an illegal money transfer that took place on May 17.



The suspect, Sally Stealer, stated that she has never logged in to the company's banking website and that she definitely did not transfer any money on **0517**.



You must use the **grep** command to search the application logs to see if Sally Stealer logged in on that day and, if so, when she transferred funds.

---



# Instructor Demonstration

---

grep

# Demo Review: `grep`

---

The previous demo covered the following concepts:



`grep` is a command-line command to find a data point **inside** of a file.



The basic syntax is: `<grep data_point File_to_search_inside>`



`grep` by default will return the **whole line** on which it finds the data point.



The `i` option will search for the data point with case insensitivity.



The `l` option will return the file names of the file containing the data point.

---



## Activity: `grep`

Sugar Corp recently made a large purchase of guavaberries.

- Your manager suspects Sugar Corp is trying to reproduce some of Candy Corp's secret recipes that use guavaberries.
- Your task is to determine which of the secret recipes contain guavaberries in their ingredient list.

Suggested Time:

---

15 Minutes





Time's Up! Let's Review.

# Questions?




# Piping

# So Far...

---

We've covered many commands line tools commonly used by IT and security professionals:



**find** to search  
for file names  
or directories.

**grep** to search  
for data points  
inside of files.

**wc** to count  
lines or words  
inside of files.

# Combining Commands

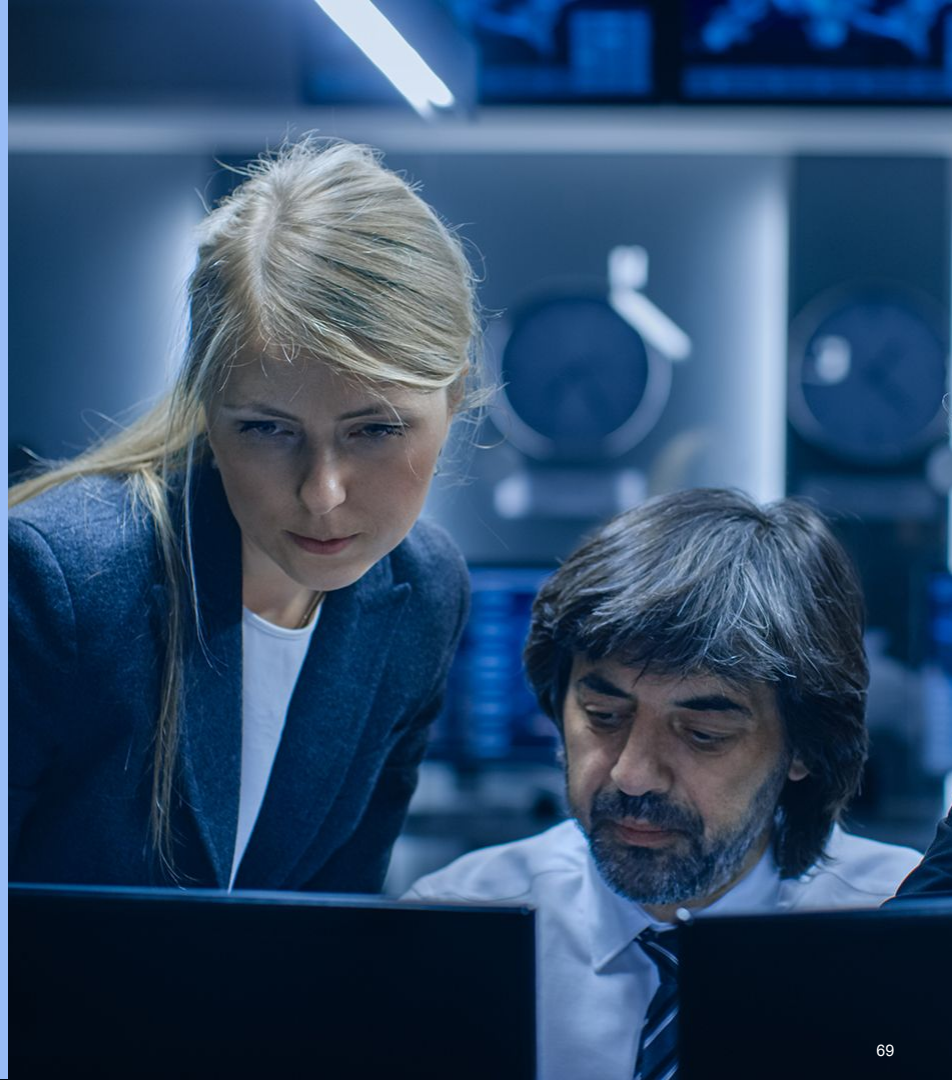
---

It's common practice to combine commands in order to complete certain tasks.

## For example:

A security analyst might be tasked with determining if a user exists in a log file, and how many times that user appears.

- We can use the **grep** command to see if a user appears, by redirecting the results into an output file.
- We can use the **wc -l** command to determine how many times the user appears in the file by counting the results of the output file.



# Pipes

---

We can use pipes to combine commands in a single line.



Pipes redirect the output of one command to another, to complete additional tasks.



A pipe is designated with the following symbol: |



Multiple pipes can be used in a single command.



Pipes are unidirectional, meaning processing of data flows from left to right.

# Pipes Demo Setup

---

In the next demonstration, we'll use the following scenario:



Our manager at ACME Corp has tasked us with continuing the previous investigation against Sally Stealer. They believe she may have transferred other large amounts of money.



Our manager created a single file, `largetransfers.txt`, containing all transfers over one million dollars.



We must count how many of those transfers belong to Sally Stealer.

---



# Instructor Demonstration

---

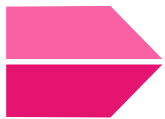
## Pipes



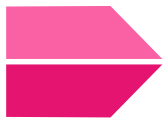
# Demo Review: Pipes

---

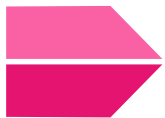
In the previous demo, we covered the following concepts:



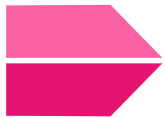
A pipe is used to take the output of one command and redirect it to another, in order to complete an additional task on the output.



A pipe is designated with the following symbol : |



Multiple pipes can be used in a single command.



Pipes are unidirectional, meaning the processing of the data flows from left to right through the pipeline.



## Activity: Gathering Evidence

Candy Corp believes they have enough evidence to send to the authorities to have Sugar Corp charged with a cyber crime.

Your task is to gather several points of evidence from your file systems to provide to the authorities to prove Sugar Corp is stealing data.

Suggested Time:

---

15 Minutes



Time's Up! **Let's Review.**

# Questions?

An illustration of seven diverse hands raised against a dark background with diagonal lines. The hands are of various skin tones (red, orange, brown) and are wearing different colored sleeves (red and white stripes, blue, blue with black and red patterns, blue with yellow patterns, red, purple, blue). Some hands have accessories like a watch, rings, and a bracelet. The word "Questions?" is written in large white text at the top.



**Which command do we use  
to access information about a  
specific command?**

man

---



**Which command do we use  
to find a file or directory?**

find

---





**Which command do we use to find a specific data point in a file?**

grep

---



**How do we combine commands?**

pipes

---

# Questions?



*The  
End*