How Sweet is the Terminal?

3.1

Cybersecurity

Terminal 101 Day 1

Class Objectives

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

- Explain why the command line is important for IT and security professionals.
- Use commands like `ls`, `cd`, `mkdir`, `touch`, `cp`, `mv`, `rm`, `rmdir`, and `cat` for basic file navigation.
- Navigate deeply nested folder structures using relative and absolute file paths.
- Use commands like `head`, `tail`, `more`, and `less` to preview files in the command line.
- Combine the above commands in sequence to accomplish relevant IT tasks.

Why the Terminal Matters

Graphical User Interfaces

We are used to Graphical User Interfaces (**GUIs**), with their familiar icons, windows, buttons, and mouse pointers.

Mac	PC

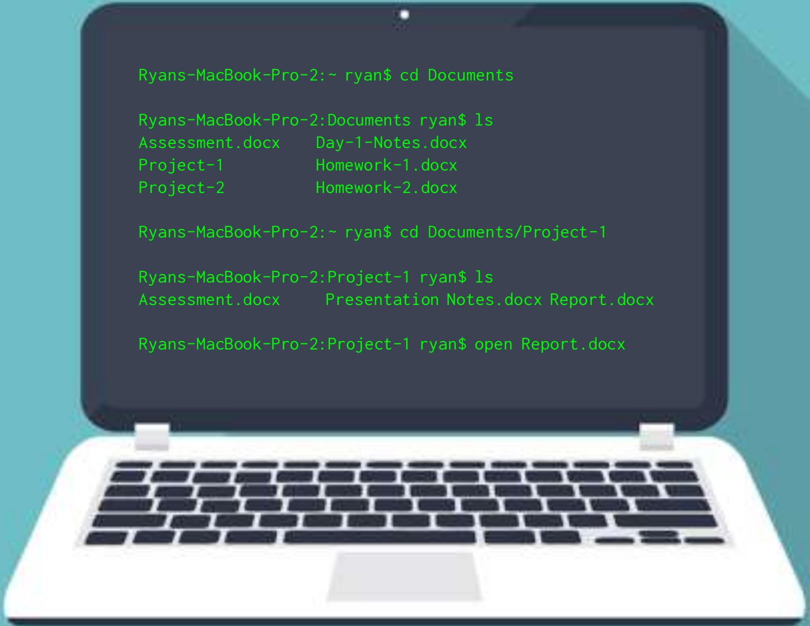
(MacOS; Windows)

What is the Command Line?

- Command line users don’t directly interact with icons and buttons on the screen.
- Users complete tasks on their computer by issuing commands with a line or lines of text in the terminal.

For example:

- Rather than clicking on a file to open it, we can type a command to open it.

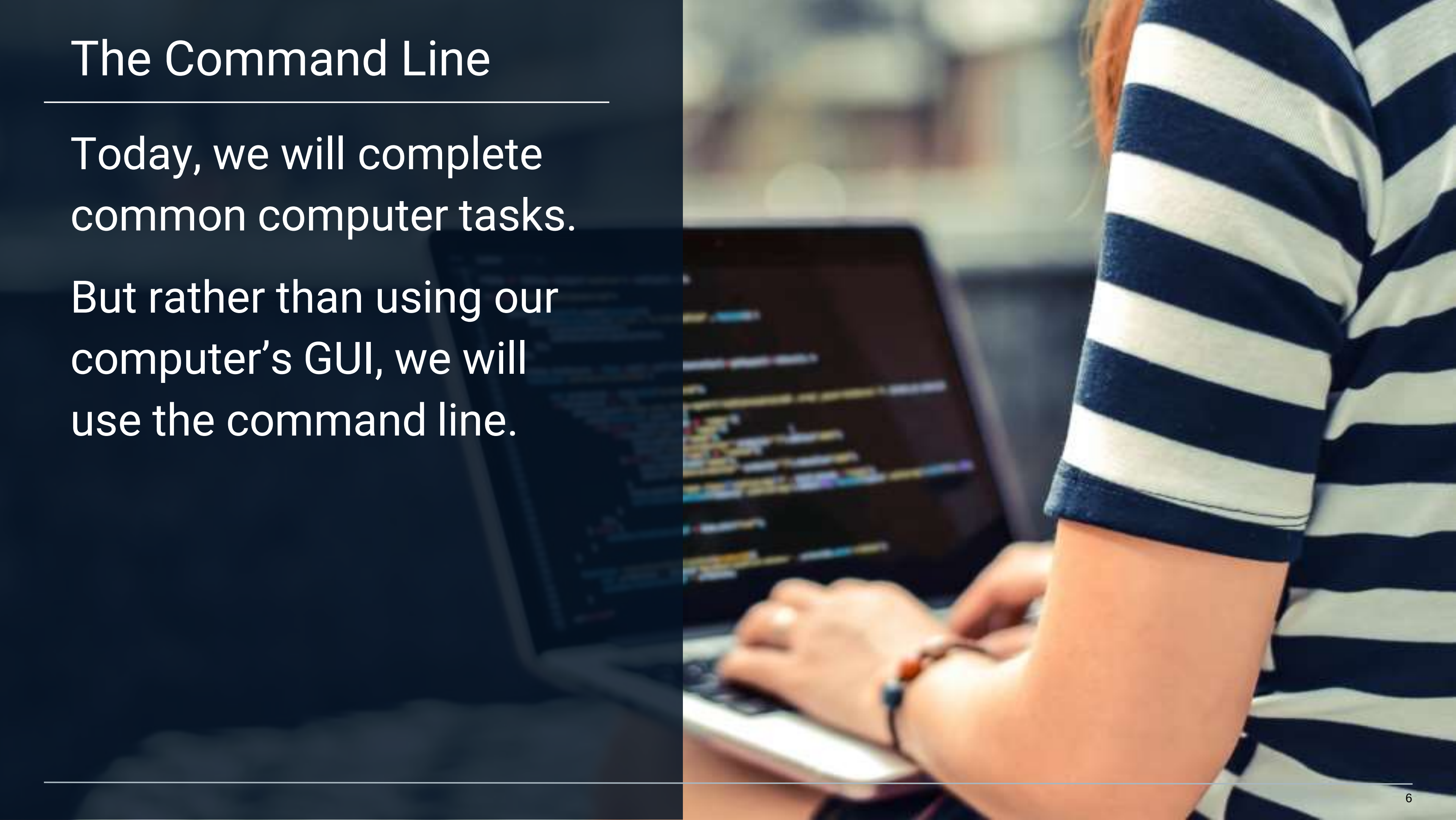


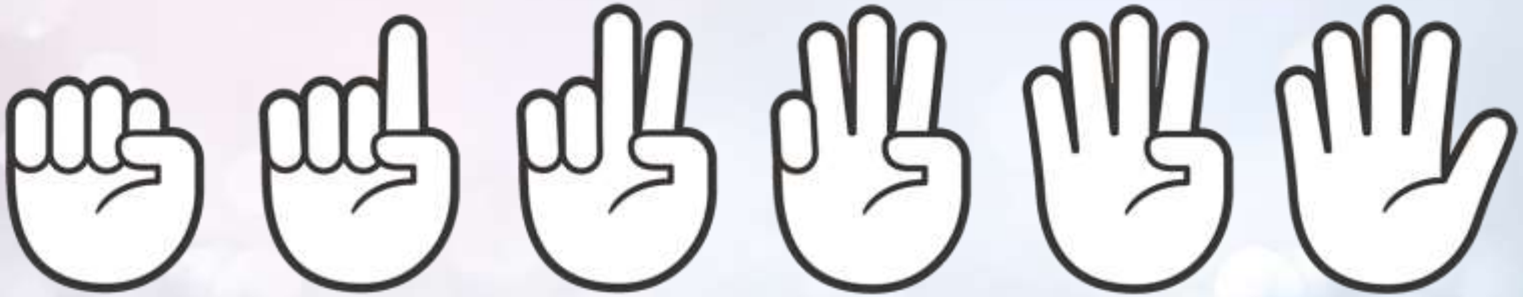
```
Ryans-MacBook-Pro-2:~ ryan$ cd Documents

Ryans-MacBook-Pro-2:Documents ryan$ ls
Assessment.docx    Day-1-Notes.docx
Project-1          Homework-1.docx
Project-2          Homework-2.docx

Ryans-MacBook-Pro-2:~ ryan$ cd Documents/Project-1

Ryans-MacBook-Pro-2:Project-1 ryan$ ls
Assessment.docx     Presentation Notes.docx Report.docx

Ryans-MacBook-Pro-2:Project-1 ryan$ open Report.docx
```

The Command Line

Today, we will complete common computer tasks.

But rather than using our computer’s GUI, we will use the command line.



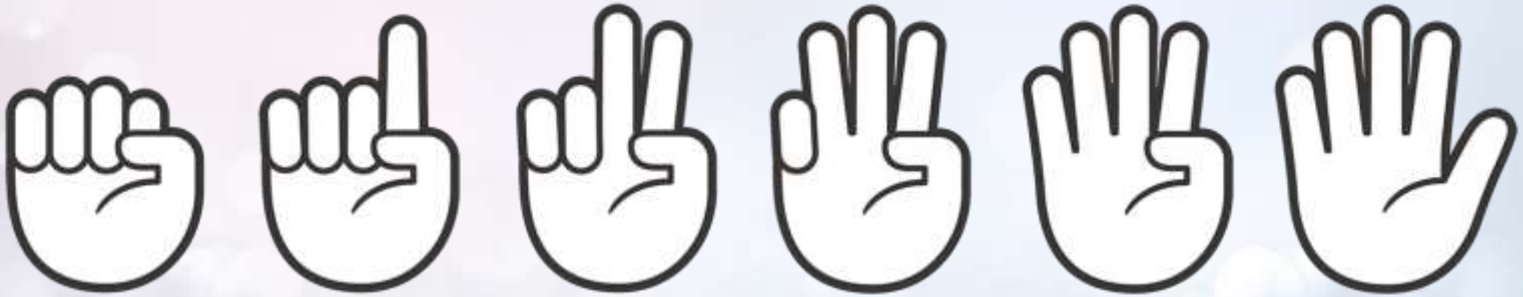


FIST TO FIVE:

Raise a Fist: If you’ve never worked with the command line and barely know what it is.

Raise a Five: If you work with the command line on a daily basis.

Raise One, Two, Three or Four: If you fall somewhere in between.



FIST TO FIVE:

For those who have used the command line: where have you used it and what for?

Why the Command Line?

Why the Command Line?

Using the command line is a critical skill. At different times, it may be:

01

The only way to achieve a desired outcome.

02

The fastest way to achieve a desired outcome.

03

The most flexible way to achieve a desired outcome.

Why the Command Line?

01

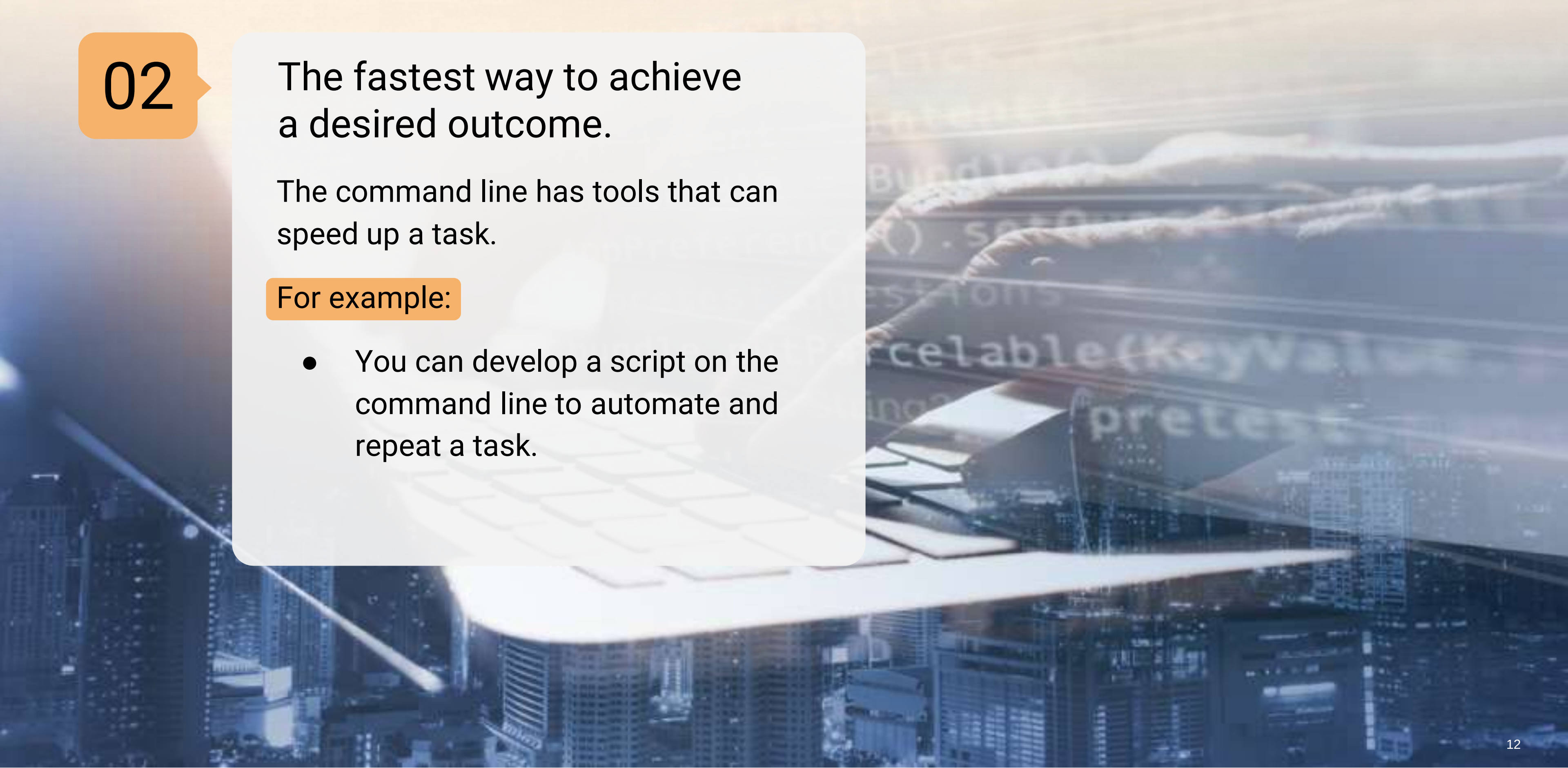
The only way to achieve a desired outcome.

You may have to work with systems and tools that have no GUI interface.

For example:

- You need to configure a system that does not have a GUI. The command line is your only mode for configuration.

Why the Command Line?



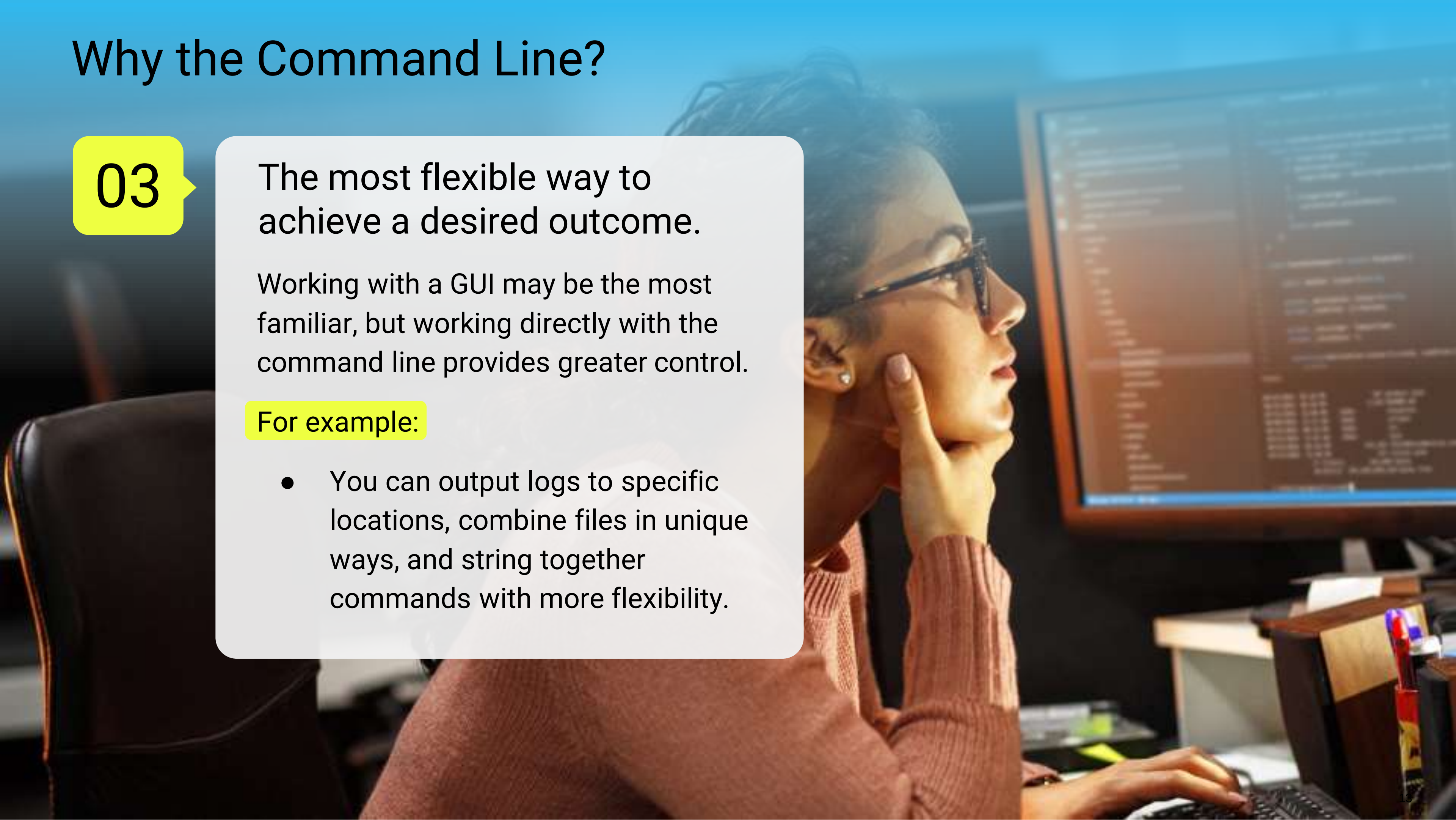
02

The fastest way to achieve a desired outcome.

The command line has tools that can speed up a task.

For example:

- You can develop a script on the command line to automate and repeat a task.



Why the Command Line?

03

The most flexible way to achieve a desired outcome.

Working with a GUI may be the most familiar, but working directly with the command line provides greater control.

For example:

- You can output logs to specific locations, combine files in unique ways, and string together commands with more flexibility.

Words to the Wise

A few things to remember.

- The best way to learn the command line is to use it.
- This week will be heavily focused on hands-on activities.
- Take notes during each demo and try your best on activities.
- **Remember:** Practice makes perfect!

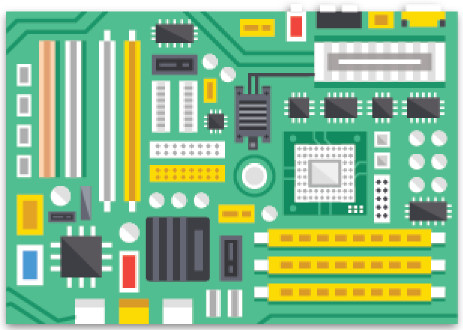
Virtual Machines

Physical Machines

To most people, a computer means a physical laptop or desktop computer, made up of hardware, such as:

Monitors

Graphic Cards



Hard Drives

etc.

Virtual Machines

As we learn how to work on the command line, we will be using a new tool called **virtual machines**.

Virtual Machines

Virtual machines are programs that simulate entire computers.

- Virtual machines (VMs) can operate as entirely different computers than the one they are running on.
- We can run many different VMs on a single physical computer.

Physical vs. Virtual Machines

Virtual

vs.

- Easy and inexpensive (or free) to set up.
- Easily distributed. In this class, we'll be distributing VMs so each student is running the exact same setup.
- Multiple VMs can be placed on a single physical machine.

Physical

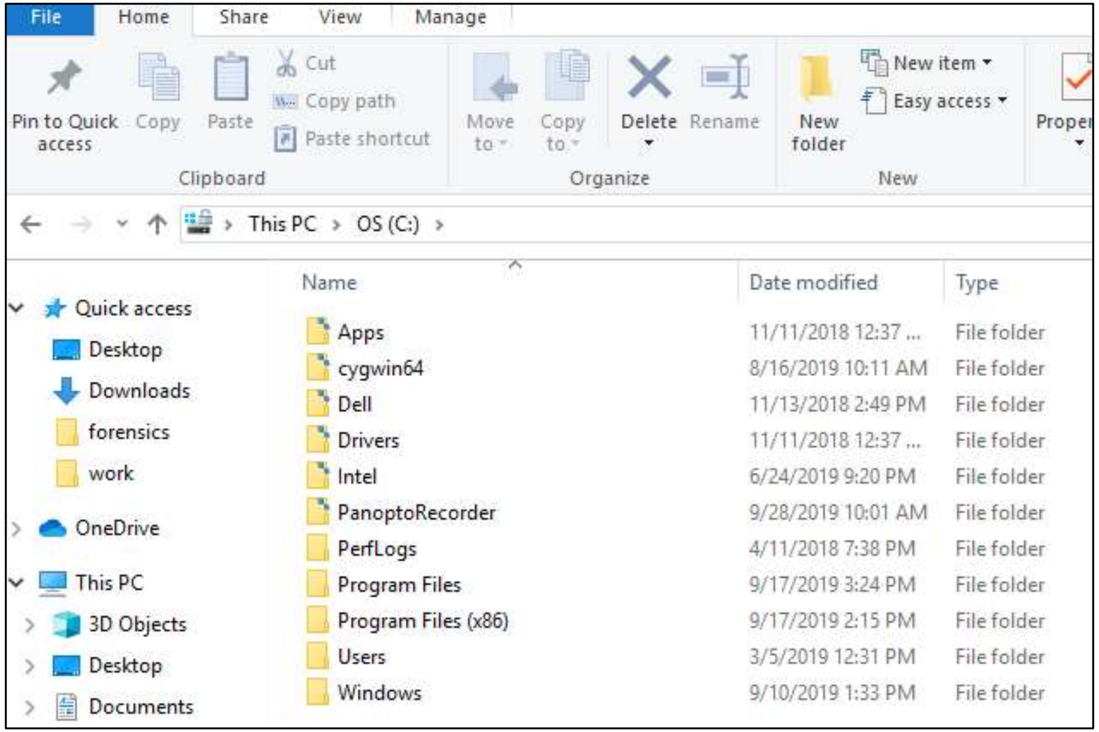
- Typically more efficient because they have direct access to hardware components.

Working on the Command Line

Within this Windows Explorer, we can complete the following tasks:

- View current directory location.
- Navigate through directories and files.
- Create new folders.

In the next demo, we'll complete these tasks on the command line.



Name	Date modified	Type
Apps	11/11/2018 12:37 ...	File folder
cygwin64	8/16/2019 10:11 AM	File folder
Dell	11/13/2018 2:49 PM	File folder
Drivers	11/11/2018 12:37 ...	File folder
Intel	6/24/2019 9:20 PM	File folder
PanoptoRecorder	9/28/2019 10:01 AM	File folder
PerfLogs	4/11/2018 7:38 PM	File folder
Program Files	9/17/2019 3:24 PM	File folder
Program Files (x86)	9/17/2019 2:15 PM	File folder
Users	3/5/2019 12:31 PM	File folder
Windows	9/10/2019 1:33 PM	File folder

Basic Terminal Commands

Basic Commands

In the next demo, we'll cover the following commands:

`pwd`	Display the current working directory (*print working directory*).
`ls`	List directories and files in the current directory.
`cd`	Navigate into a directory (*change directory*).
`mkdir`	Make a directory.
`rmdir`	Remove a directory.
`touch`	Create an empty file.
`rm`	Remove a file.
`clear`	Clear the terminal history of a page.

Demonstration Scenario

We are security analysts at ACME Corp. Our manager assigned us several investigations and gave us a computer in the evidence room. We are asked to:

01

Create several directories to organize our investigations: `Case1` and `Case2`.

02

Put our directories in the already existing folder `security_evidence`.

03

Put an empty file in the `Case1` folder titled `case1_evidence`.

Demonstration Review

What does each of the commands do?

`pwd`	
`ls`	
`cd`	
`mkdir`	
`rmdir`	
`touch`	
`rm`	
`clear`	

Demonstration Review

What does each of the commands do?

`pwd`	Display the current working directory (*print working directory*)
`ls`	List directories and files in the current directory
`cd`	Navigate into a directory (*change directory*)
`mkdir`	Make a directory
`rmdir`	Remove a directory
`touch`	Create an empty file
`rm`	Remove a file
`clear`	Clear the terminal history of a page

Activity: Take Five and Practice the Command Line

In this activity, you are a security analyst at Wonka Corp. Wonka Corp believes an employee is selling secret recipes. You must create evidence directories for `email`, `logs`, and `web_access` to organize your investigation of the rogue employee.

Suggested Time:
7 minutes

Times Up! Let’s Review.

Command Line Review

Completing this activity required the following steps:

01 Make a directory.

02 Navigate into the directory.

03 Print the working directory.

04 Create empty files.

05 Delete files.

06 List files.

07 Clear the terminal window.

File Paths

File Paths

A file path is the unique location in the directory where a file is located.

The **`cat.jpg`** image file is located in the **`my_images`** directory, which is in the **`desktop`** directory, which is in the **`root`** directory.

```
/root/desktop/my_images/cat.jpg
```

The directory path is: **`/root/desktop/my_images/`**.

The file is an image called: **`cat.jpg`**.

File Paths in IT

- You are a security administrator and need to access a network log for an investigation.
- You might ask the network team for the location or path of the file. The network team would provide you with a file path such as:

```
/root/var/logfile1.txt
```

There are two types of file paths: **absolute paths** and **relative paths**.

Absolute Paths

```
root/Desktop/Sallysfiles/textfiles/Sallyfile.txt
```

An **absolute path** indicates the location of a file regardless of your current location in the file directory.

- If you're in the `Applications` directory and need to access a file in the `Desktop` directory, you can use an absolute path to get the file without navigating to the `Desktop` directory.

Absolute Paths

We want to access a file located at:

```
/root/Desktop/Sallysfiles/textfiles/Sallyfile.txt
```

- If we are located in a different directory, such as,

```
/root/rootfiles/file_logs/
```

- We can use the absolute path,

```
/root/Desktop/Sallysfiles/textfiles/Sallyfile.txt
```

- To access

```
Sallyfile.txt
```

Relative Paths

```
/Sallysfiles/textfiles/Sallyfile.txt
```

A **relative path**, starts at your current location in the directory and doesn't require typing out the root directory.

- If you need to access a file in the `Desktop` directory and are already in `Desktop`, create a path from your current point.
- In the above example, the user is in the `Desktop` directory, so `root/Desktop/` is assumed and not written.

File Paths on the Command Line

Relative and absolute paths function as helpful shortcuts on the command line.

```
cd Desktop
cd textfiles
cd Sallysfiles
```

```
cd Desktop/textfiles/Sallysfiles/
```

Demonstration Scenario

Our manager at ACME Corp. needs us to create several additional evidence files for our investigation. They provided the following instructions:

01 Navigate to the directory `/home/instructor/Cybersecurity-Lesson-Plans/03-Terminal/instructor/pathnav_demonstration/security_evidence/Case2/`. Create an empty file called `case2_evidence`.

02 Navigate to the directory `Case1` and create an empty file called `Web_logs`.

03 Navigate back to the root folder.

Instructor Demonstration

Relative vs. Absolute Paths and the `cp` and `mv` Commands

Copying and Moving Files

Copying Files

Security and IT professionals often *copy* files from one location to another.

For example:

- You're asked to make a copy of a log file and put it in a separate evidence folder.

The original file stays in its original location.

cp *(copy)* creates a copy of the file and places it in a specified location.

The original file remains. A copy is created in the new location, where it can be edited as needed.

Moving Files

Security and IT professionals often *move* files from one location to another.

For example:

- An image file is accidentally placed in the email directory. You have to move the file out of the email directory and into the correct image directory.

`mv` (*move*), relocates the file from its current location to a new, specified location.

Unlike **`cp`**, the original file doesn't remain.

The file is moved into the new location. (Similar to "cut and paste.")

cp and mv Syntax

cp and mv have a similar syntax.

```
mv cat.jpg /dirA
```

- This command moves the file `cat.jpg` into the directory `dirA`.
- The `<file>` or `<location>` can be indicated with an absolute or relative path if the file is not in your current location.

```
mv dirA/dirB/dirC/cat.jpg /dirA/dirB
```

- This command moves the file `cat.jpg` from the dirC directory, over to the `dirB` directory.

Copy and Move Demo Scenario

Our manager made a mistake. They realize they only need web activity for Case 2.

- We need to move the `Web_logs` file from the `Case1` directory to the `Case2` directory.
- After we move the file into the `Case2` directory, we must
  - make a copy of the `Web_logs` file and
  - put it in the `security_evidence` folder.

Let’s Review

What does each of these do?

`file path`
`absolute path`
`relative path`
`cp`
`mv`

Let’s Review

What does each of these do?

`file path`	Identifies a unique location in a file system.
`absolute path`	Starts the root of the file system.
`relative path`	
`cp`	
`mv`	

Let’s Review

What does each of these do?

`file path`	Identifies a unique location in a file system.
`absolute path`	Starts the root of the file system.
`relative path`	Begins from the current directory.
cp	
mv	

Let’s Review

What does each of these do?

`file path`	Identifies a unique location in a file system.
`absolute path`	Starts the root of the file system.
`relative path`	Begins from the current directory.
`cp`	Copies files.
`mv`	

Activity: Finding Your Milky Way

You continue in your role as security analyst at Wonka Corp. Your manager now believes there is a second employee working with Slugworth Corp.

To continue the investigation, you've been asked to create an additional directory, and copy and move several of the evidence files.

Suggested Time:
15 minutes

Times Up! Let’s Review.

Finding Your Milky Way Review

Completing this activity requires the following steps:

- Make an additional directory.
- Use the *move* command with an absolute path.
- Use the *copy* command with an absolute path.
- Navigate to the directories with absolute or relative paths.
- Confirm the files are moved and copied correctly.

Finding Your Milky Way Review

To move `email_evidence`

from `Internal_Investigation_Employee_A`
to `Internal_Investigation_Employee_B`

```
mv/home/sysadmin/Cybersecurity-Lesson-Plans/03-
Terminal/student/take_5/Internal_Investigation_Employee_A/email_evi
dence /home/sysadmin/Cybersecurity-Lesson-Plans/03-
Terminal/student/take_5/Internal_Investigation_Employee_B/
```

Finding Your Milky Way Review

To copy `log_evidence`

from `Internal_Investigation_Employee_A`
to `Internal_Investigation_Employee_B`

```
cp/home/sysadmin/Cybersecurity-Lesson-Plans/03-
Terminal/student/take_5/Internal_Investigation_Employee_A/log_evidence
/home/sysadmin/Cybersecurity-Lesson-Plans/03-
Terminal/student/take_5/Internal_Investigation_Employee_B/
```

Take a Break!

Preview Commands

Previewing Files

Security and IT professionals often need to *preview* the contents of a file.

For example:

A security administrator needs to preview the top 10 lines of an email file to figure out who it was sent to.

+

A network administrator needs to preview the bottom of a log file to see the last timestamp captured.

`More Less Head` and `Tail`

To preview and scroll through a whole file, use the **`more`** or **`less`** commands.

`more`	View a file one page at a time. Space bar moves you to the next page.
`less`	Similar to `more`, but it allows you to scroll up and down a file.

To preview a file by a certain number of lines, use the **`head`** or **`tail`** commands.

`head`	Displays the top 10 lines of a file.
`tail`	Displays the bottom 10 lines of a file.

Modifying the Preview

`head` and `tail` preview 10 lines by default. This number can be changed easily.

- The syntax for `head` is `head -number file`.
    - For example:
      `head-50 logfile.txt` shows the top 50 lines in `logfile.txt`.
- The same syntax applies for the `tail` command.

Demo Scenario

At ACME Corp, we've been asked to examine several potential evidence files.

Our manager provided a directory, `evidence_directory`, containing four files: `File1`, `File2`, `File3`, `File4`.

These files were pulled from a secret computer at Wonka. They should be access logs showing who has logged in to that computer.

We'll use the preview commands to determine which files are actually access logs.

Once we determine which are access logs, we'll document the last timestamp in the file.

Let’s Review

What does each command do?

`more`	
`less`	
`head`	
`tail`	
`-number`	

Activity: Oh Henry, What Did You Do?

You continue in your role as security analyst at Wonka Corp.

- Your manager identified two employees who may be working with Slugworth—Henry and Ruth—and quickly pulled some files from their computers.
- You must preview the files to determine which have readable text data that can be used for analysis.

Suggested Time:
5 Minutes

Time’s Up! Let’s Review.

Oh Henry Review

Completing this activity required the following steps:

01

Using the **`more`**, **`less`**, and **`head`** commands to preview content from several files.

02

Using the **`tail`** command to view data at the bottom of a file.

03

Using the **`move`** command to rename a file.

Data Streams and the `cat` Command

The **`cat`** Command

Security and IT professionals often have to combine files.

For example:

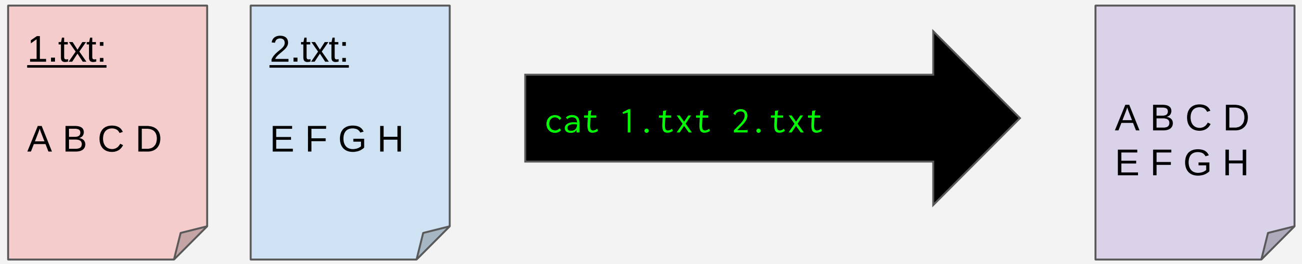
- A security professional needs to combine several of the same type of log file into a single log file.

  We can use the **`cat`** command to complete this task.

The `cat` Command

`cat` (*concatenate*): combines files and displays the results.

- To combine `1.txt` and `2.txt`, run `cat 1.txt 2.txt`.
- This displays the combined results of `1.txt` and `2.txt` — but doesn't save the results beyond the command line.



```
cat 1.txt 2.txt
```

This command does not *save* the results.
To save results of the **`cat`** command,
you have to redirect the **data stream**.

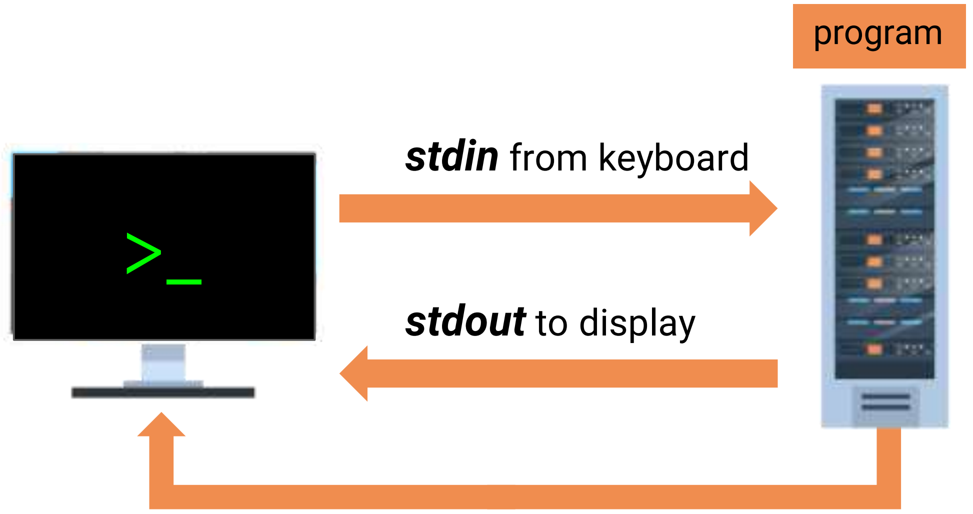
Data Streams

- A **data stream** describes channels of data as they are processed and moved through a system.
- Data streams are used to move data from one process or program to another.

stdin and ***stdout***

The most common data streams are ***stdin*** and ***stdout***, pronounced "*standard in*" and "*standard out*".

- Input data is usually the *argument* given to the program. Output data is the information returned by the program when the command succeeds.
- Input data is streamed using the data stream ***stdin***.
- Output data is streamed using the data stream ***stdout***.



Back to Saving Results with `cat`

To save the results of `cat`, you have to use either > or `>>`.

>

overwrites a file

or

>>

writes or appends to a file

`cat` with `>`

```
cat 1.txt 2.txt > combined_1&2.txt
```

- The above command will combine two separate files, `1.txt` and `2.txt`, into a single file called `combined_1&2.txt`.
- If `combined_1&2.txt` already exists, it will **overwrite** the file with the combined data from `1.txt` and `2.txt`.

`cat` with >>

```
cat 3.txt 4.txt >> combined_3&4.txt
```

- The above command will combine two separate files, `3.txt` and `4.txt`, into a single file called `combined_3&4.txt`.
- If it already exists, it will **append** to the bottom of the file with the combined data from `3.txt` and `4.txt`.

`cat` and Data Streams:

```
cat 1.txt 2.txt > combined_1&2.txt
```

- `cat` takes data directly from ***stdin*** and directs it to ***stdout***.
- `1.txt` and `2.txt` is sent to ***stdin***.
- `cat` is taking ***stdin***, concatenating the data, and sending it to ***stdout***.
- The results of combining `1.txt` and `2.txt` is ***stdout*** .
- ***stdout*** is redirected and written to a file called `combined_1&2.txt`.

Demonstration Set Up

At ACME Corp, you are asked to combine several evidence log files.

Your manager provided:

A directory, called `Logfile_evidence_directory` containing four files: `LogFile1`, `LogFile2`, `LogFile3`, `LogFile4`.

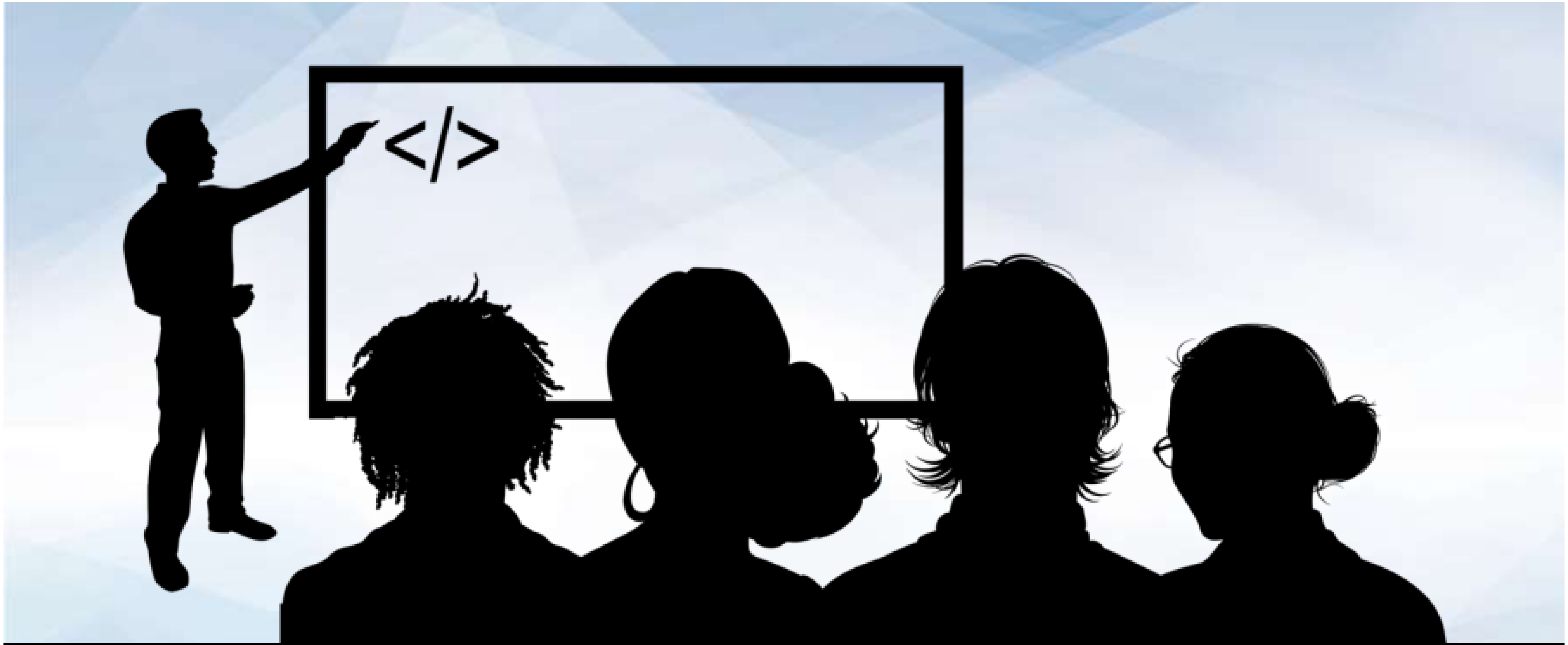
Each file has logs from a different day.

Your manager explained:

That your forensic team will be doing an analysis of these logs. Their job will be easier if you can combine the log files into a single file.

You are tasked with:

Using the `cat` command to combine the logs into a single file called `rogue_employee_log_evidence`.



Instructor Demonstration

Data Streams and the `cat` Command

Activity: Internal Investigation: Finding the Kit `cat` Burglar

You continue in your role as security analyst at Wonka Corp.

- Your manager believes Henry and Ruth are sharing secret recipes and collected some files from their computers to help you build your case.
- You must analyze this additional data and prepare a combined evidence file for the local authorities.

Suggested Time:
5 minutes

Times Up! Let’s Review.

Lesson Recap

Which command do we use to retrieve a list of all files in a folder?

Which command do we use to create a folder?

Which command do we use to combine two files?

How do we save the output?

Questions?

The End