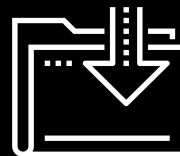







In and Out

Cybersecurity
Linux 2 Day 3



Today's Objectives

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

-  Explain the relationship between file descriptors and data streams
 -  Use redirection between file descriptors and data streams
 -  Perform the same tasks multiple times with for loops
-

I/O Data Streams, File Descriptors, and Redirection

I/O Data Streams

A **data stream** is a way to describe channels of data as they are processed and moved through a system.



Used in Linux to move data from one processor / program to another. Three most used data streams are:



stdin - used to stream input data



stdout - used to stream output data



stderr - used to stream error data

I/O Data Streams

A **data stream** is a way to describe channels of data as they are processed and moved through a system.



Used in Linux to move data from one processor / program to another.



Three most used data streams are:



stdin - used to stream input data



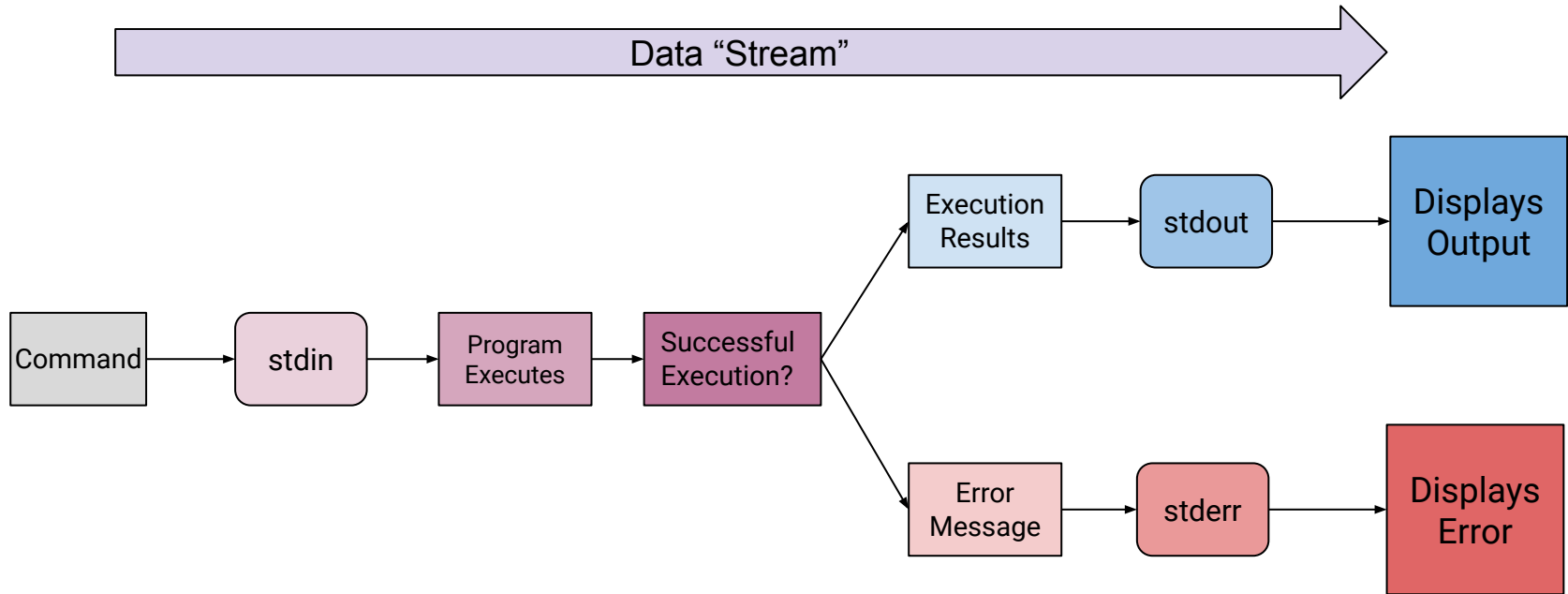
stdout - used to stream output data



stderr - used to stream error data

I/O Data Streams

A **data stream** is a way to describe channels of data as they are processed and moved through a system.





Instructor Demonstration

I/O Streams, File Descriptions and Redirections



Activity: Cat Whisperer

In this activity, you will use terminal to practice using `stdin` and `stdout` redirections with the `cat` command.

Instructions sent via Slack.

Suggested Time:
20 Minutes



Your Turn: Cat Whisperer

Instructions

Run the command `cat < /etc/passwd`.

- ☐ What do you see?
- ☐ How is this different from `cat /etc/passwd`?
- ☐ Rewrite this command using the file descriptor for `stdin`.

Next, run: `cat << EOF`

- ☐ After entering this command, you will get a prompt because the `cat` program is now waiting for your input.
- ☐ Enter a few lines of text, hitting the return key each time.
- ☐ After a few lines of text, type `EOF` and hit return. What happens?
- ☐ Explain what you think `EOF` means.

Bonus: now run: `cat << EOF > my_file.txt`

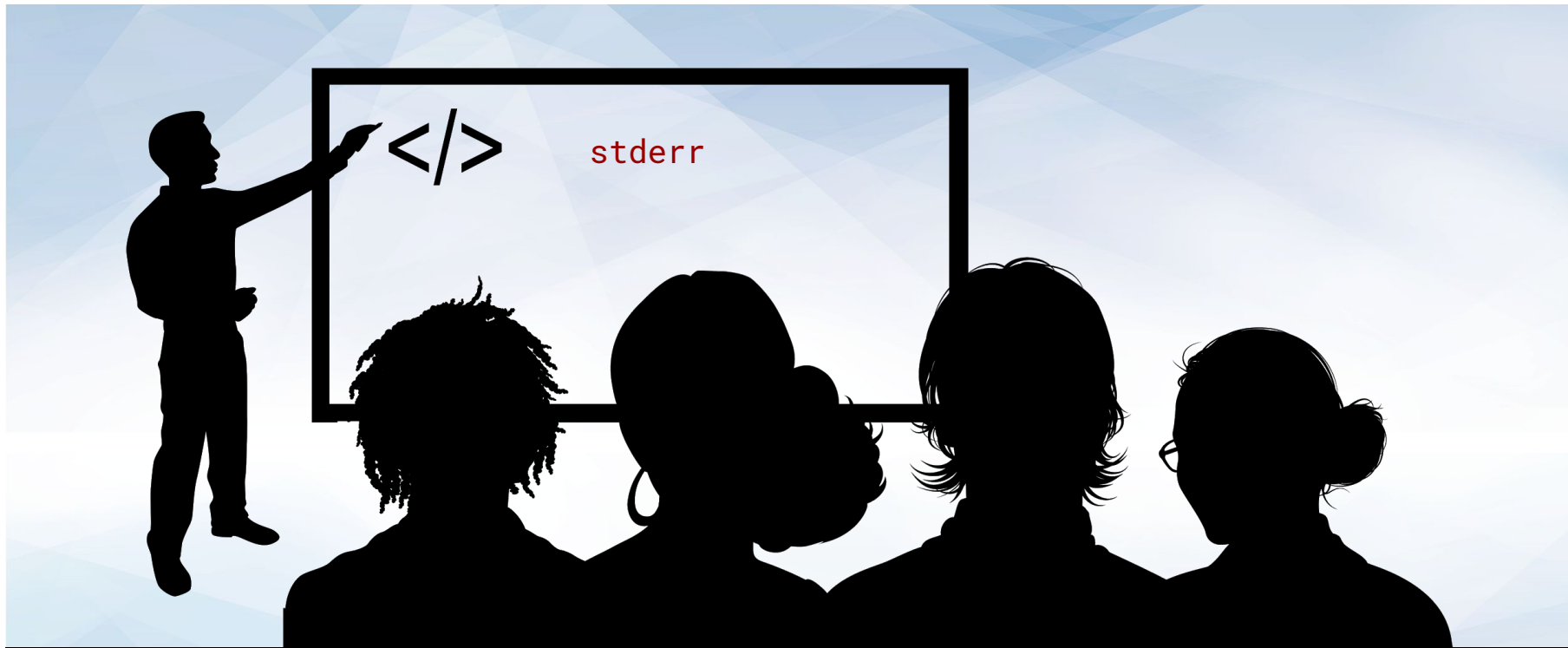
- ☐ How is this different from just `cat << EOF`
- ☐ Describe in detail what will happen when you complete this command.





Times Up! Let's Review.

Cat Whisper



Instructor Demonstration

Throw Err to the Wind

Throw Err Demo Summary

```
file $(find . -iname *.txt 2> /dev/null) > ~/Desktop/text\ files ; tail ~/Desktop/text\ files
```



\$ () for command expansion



. to reference your current location



* expands to any number of characters



2> redirects **stderr** to **dev/null** in order to hide errors



> redirects **stdout** to new file



~ expands to the home path of the user







\ escapes the space in the file name



; runs the tail command while the find command is still running

Today's Objectives

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

-  Explain the relationship between file descriptors and data streams
 -  Use redirection between file descriptors and data streams
 -  Perform the same tasks multiple times with for loops
 -  Collect user input through script arguments
-

Take a Break!





for loops



You'll often want to perform the same tasks multiple times within a script.

For loops

Want to perform the same task within a script numerous times?



Write a script that does each task explicitly?

Copy and paste whole blocks of code?

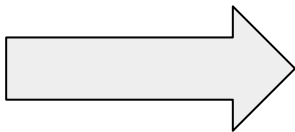
These solutions are not maintainable, they don't scale well, and they're error prone.



for loops allow you to perform the same tasks multiple times without repeating a block of code.

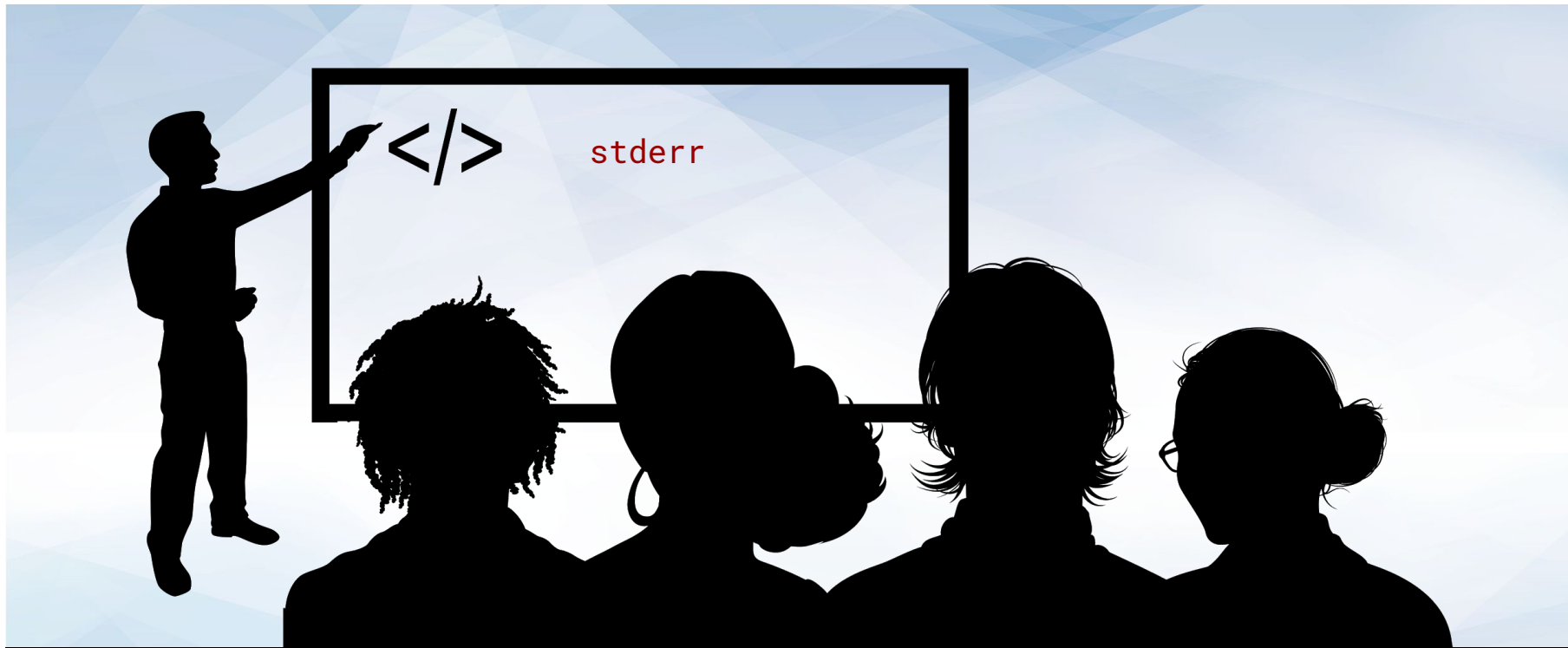
For loops syntax

```
echo "Hello, Moe!"  
echo "Hello, Larry!"  
echo "Hello, Curly!"
```



```
for NAME in "Moe" "Larry" "Curly"  
do  
    echo $NAME  
done
```

- > Begins with **VAR_NAME in LIST**
 - > The list can be a series of strings
 - > The keywords **do** and **done** mark the beginning/end of the for loop
 - > Everything in between **do** and **done** runs once for each element of the list
 - > Each time the code block runs, **NAME** changes to the next name in the list
-



Instructor Demonstration
for loops



Activity: First Steps with For

In this activity, you will use for loops to automatically create a directory tree.

Instructions sent via Slack.

Suggested Time:
20 Minutes








Times Up! Let's Review.

First Steps with For

Today's Objectives

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

-  Explain the relationship between file descriptors and data streams
-  Use redirection between file descriptors and data streams
-  Perform the same tasks multiple times with for loops