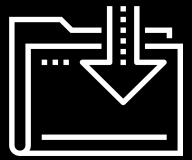
Advanced Bash

Cybersecurity

Bash Scripting and Programming Day 1



Today’s Objectives

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

- Construct compound commands using `&&`, `|` and file redirects.
- Create alias commands and save them to their `~/.bashrc` file.
- Edit your $PATH variable to include a custom `~/scripts` directory.
- Create simple bash scripts comprised of a list of commands.

Creating Compound Commands

Why Compound Commands?

Navigating Linux directories, quickly searching large log files, and writing small scripts to automate tasks will save you time and energy.

What are Compound Commands?

Compound Commands are several individual commands that we would originally run separately *linked together* to create a new command.

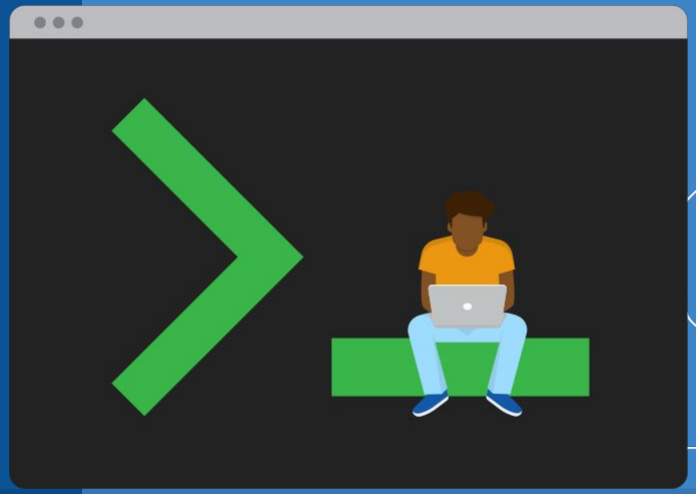
Syntax Breakdown

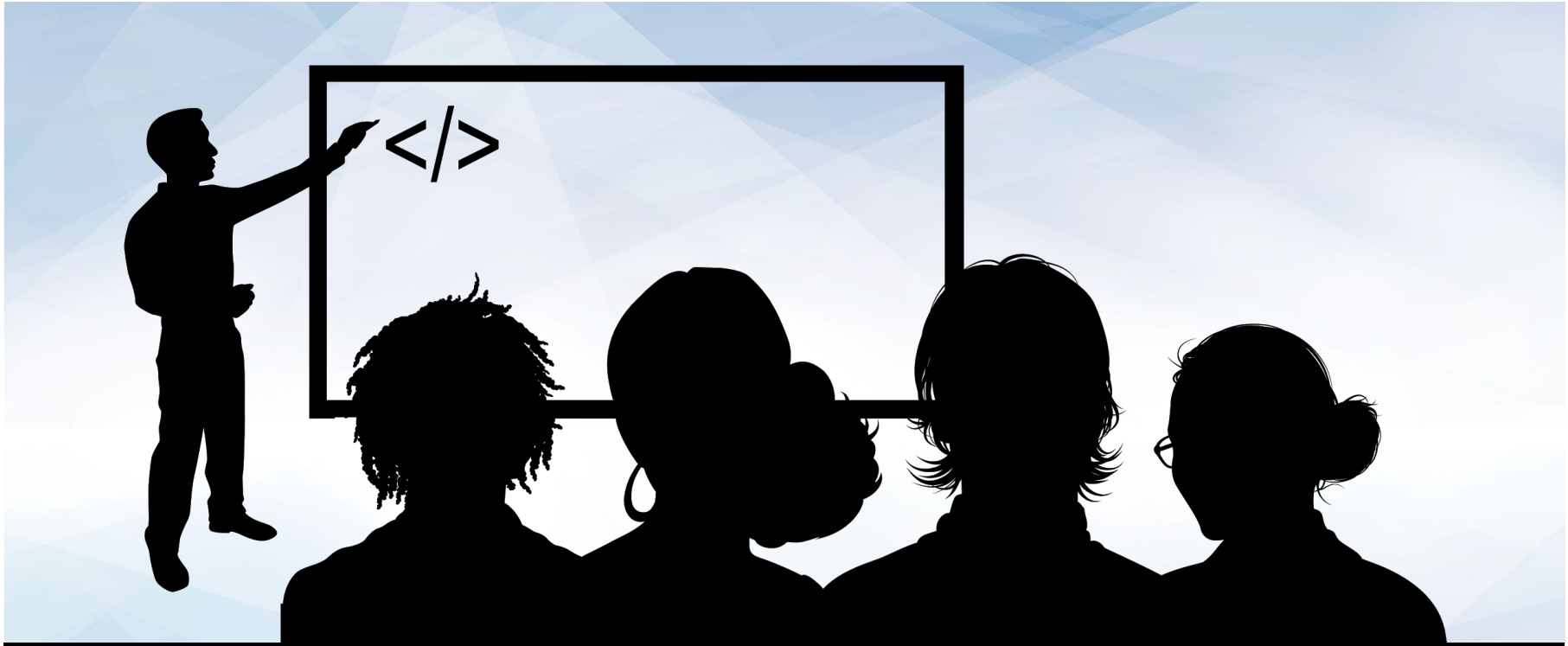
```
file $(find / -iname *.txt 2>/dev/null) > ~/Desktop/text_files ; tail ~/Desktop/text_files
```

- Searches the entire computer for files ending in .txt;
- Verifies that the files found are text files;
- Ignores any errors it comes across;
- Creates a list of all found files before saving the list to the desktop;
- Finally, it open the file and actively shows each line that is added.

Let’s Review

We’ve already chained commands using the following >, >>, and |.





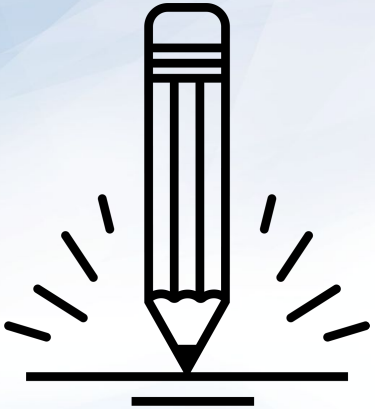
Instructor Demonstration

Chaining Commands Review

Combining Commands:

In the previous demo, we covered how to chain commands using the following:

>	`ls >list`	Takes the output of the ls command and sends it into a .txt file.
>>	`ls >> list.txt`	Takes the output of the ls command and sends it into a .txt file. If .txt does not exist, it will be created.
|	`ls -l | grep `*.txt` `	Pipes the output of one command and sends it as the input into the following commands.
;	`mkdir dir; cd dir; touch file; ls -l`	These commands run back to back, regardless of the outcome.
&&	`mkdir dir && cd dir && touch file && ls -l`	The next command is only run if the previous command was successful.



Activity: Compound Commands

In this activity, you will audit a new system. In order to simplify the process, we will combine several commands together.

Suggested Time:
15 minutes

Activity Instructions: Compound Commands

Create a research directory and copy all system logs, the shadow, passwd and host files in one command.	[redacted]
Using one command, create a list of all SUID files and save it to a text file in the research folder .	[redacted]
Create a list of top 10 most active processes. The list should only contain USER, PID, % CPU, %MEM and COMMAND.	[redacted]
Create a list of home folders along with user info from the passwd file. Only add the user info to your list if the UID is greater than 1000.	[redacted]

Suggested Time: 15 Minutes

Activity Review: Compound Commands

Create a research directory and copy all system logs, the shadow, passwd and host files in one command.	`mkdir ~/research && cp -r /var/log/* /etc/passwd /etc/shadow /etc/hosts ~/research`
Using one command, create a list of all SUID files and save it to a text file in the research folder .	`sudo find / -type f -perm /4000 > ~/research/suid_lst.txt`
Create a list of top 10 most active processes. The list should only contain USER, PID, % CPU, %MEM and COMMAND.	`ps aux --sort -%mem | awk {‘print $1, $2, $3, $11’} | head > ~/research/top_processes.txt`
Create a list of home folders along with user info from the passwd file. Only add the user info to your list if the UID is greater than 1000.	`ls home > ~/research/users.txt && cat /etc/passwd | awk -F “:” `{if ($3 >= 1000) print $0}` >> ~/research/users.txt`

Changing Aliases

What are Aliases?

While compounds are useful, they require a lot of typing.

We can use **aliases** as custom commands
that launch our compound command.

Syntax Breakdown

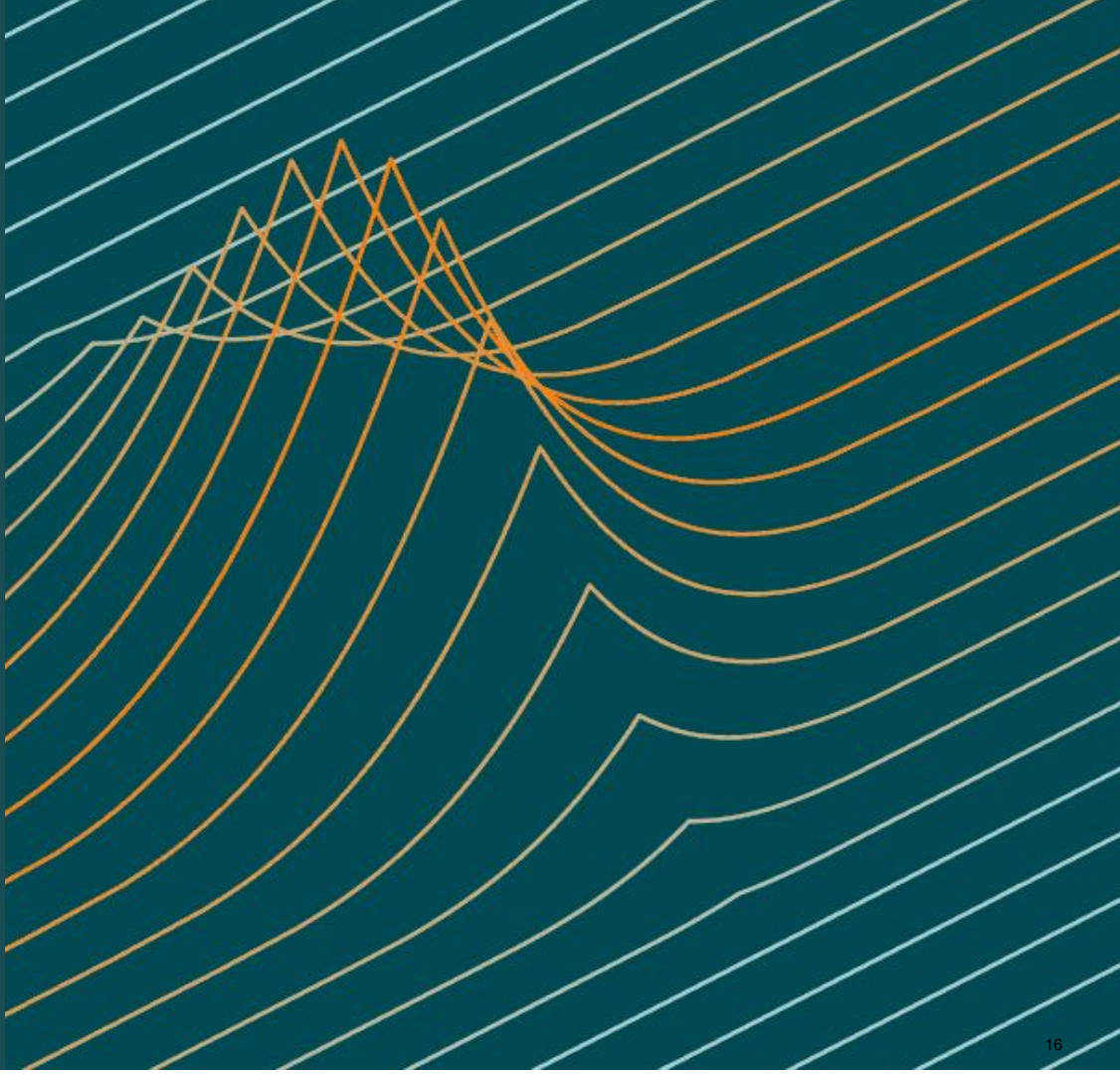
```
alias lh='ls -lah
```

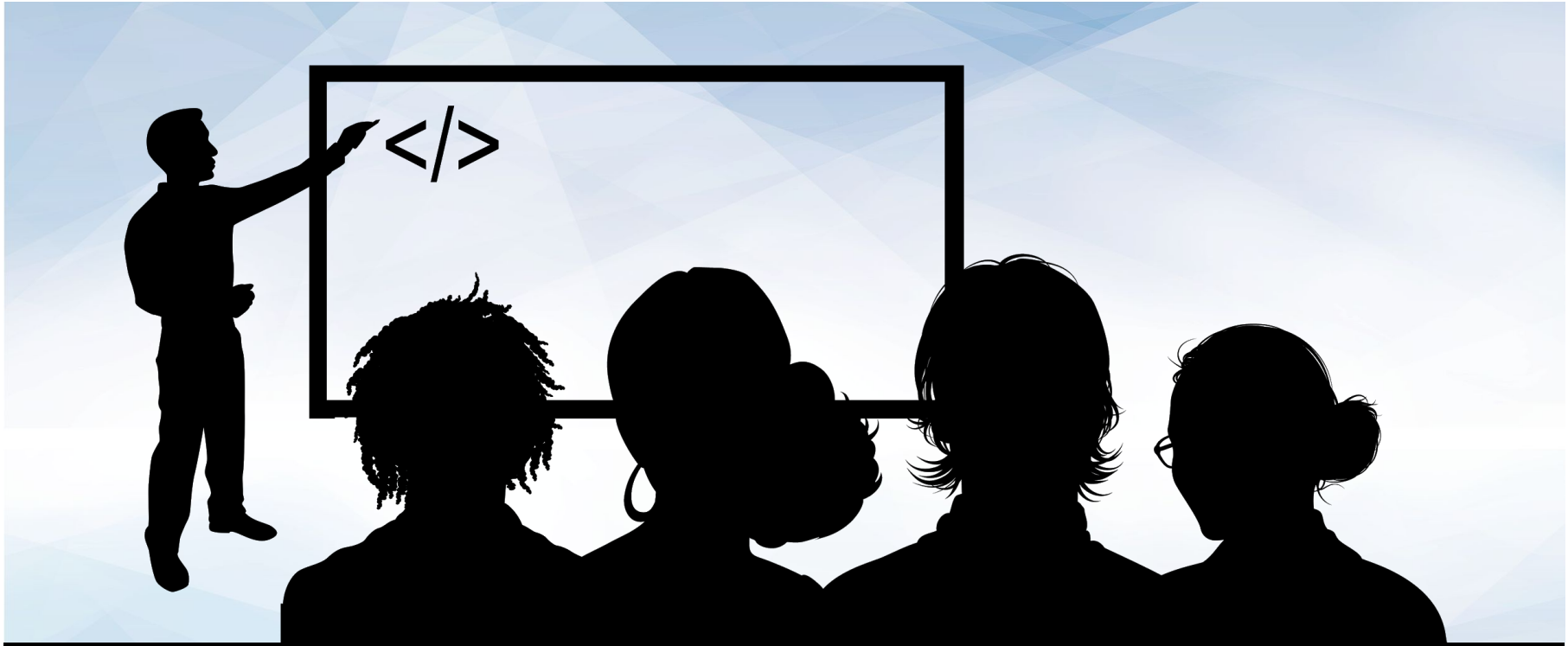
- `alias` indicates we are creating an alias.
- `lh` is our custom command.
- `ls -lah` is the command that runs when we use out alias `lh`.

In the next demo, we will create custom commands using aliases and save them.

Alias Demo Set Up

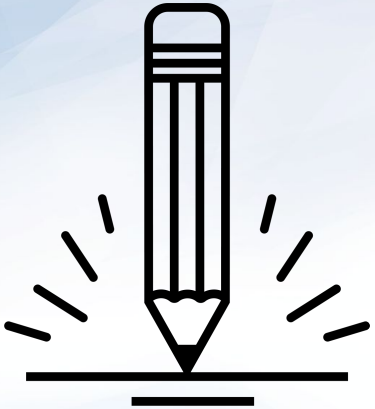
In the next demo, we will create custom commands using aliases and save the configuration file so we can use them again whenever we login.





Instructor Demonstration

Creating Aliases



Activity: Creating Aliases

In this activity, you will create several aliases and save them to your `~/.bashrc` file.

Suggested Time:
10 minutes

Activity Instructions: Creating Aliases

Create aliases for the following commands in your `~/.bashrc` file:

`ls -la`	
`cd ~/Documents`	
`cd ~/Downloads`	
`cd etc`	
`nano ~/.bashrc`	

Suggested Time: 10 Minutes

Activity Review: Creating Aliases

Create aliases for the following commands in your `~/.bashrc` file:

`ls -la`	`echo “alias lsa=’ls -a’” >> ~/.bashrc`
`cd ~/Documents`	`echo "alias docs='cd ~/Documents'" >> ~/.bashrc`
`cd ~/Downloads`	`echo "alias dwn='cd ~/Downloads'" >> ~/.bashrc`
`cd etc`	`echo "alias etc='cd /etc'" >> ~/.bashrc`
`nano ~/.bashrc`	`echo “alias rc=’nano ~/.bashrc” >> ~/.bashrc`

Review continued on next slide.

Activity Review: Creating Aliases

Create aliases for the four previous exercises:

`mkdir ~/research && cp /var/logs/* /etc/passwd /etc/shadow /etc/hosts ~/research`	[redacted]
`sudo find / -type f -perm /4000 > ~/research/suid_lst.txt`	[redacted]
`ps aux -m | awk {'print $1, $2, $3, $4, $11'} | head> ~/research/top_processes.txt`	[redacted]
`ls home > ~/research/users.txt && cat /etc/passwd | awk -F ":" '{if ($3 >= 1000) print $0}' >> ~/research/users.txt`	[redacted]

Review continued on next slide.

Activity Review: Creating Aliases

Create aliases for the four previous exercises:

`mkdir ~/research && cp /var/logs/* /etc/passwd /etc/shadow /etc/hosts ~/research`	`echo "alias logs='mkdir ~/research && cp /var/logs/* /etc/passwd /etc/shadow /etc/hosts ~/research'" >> ~/.bashrc`
`sudo find / -type f -perm /4000 > ~/research/suid_lst.txt`	`echo "alias suid='sudo find / -type f -perm /4000 > ~/research/suid_lst.txt'" >> ~/.bashrc`
`ps aux -m | awk {'print $1, $2, $3, $4, $11'} | head> ~/research/top_processes.txt`	`echo "alias aux='ps aux --sort -%mem | awk {'print $1, $2, $3, $4, $11'} | head > ~/research/top_processes.txt'" >> ~/.bashrc`
`ls home > ~/research/users.txt && cat /etc/passwd | awk -F ":" '{if ($3 >= 1000) print $0}' >> ~/research/users.txt`	`echo "alias users='ls home > ~/research/users.txt && cat /etc/passwd | awk -F ":" '{if ($3 >= 1000) print $0}' >> ~/research/users.txt'" >> ~/.bashrc`

Custom Commands

Now, we will create a custom command that runs our script.

- In order to do this, we'll have to look under the hood of what happens when we run commands.
- We'll also look a built-in variable known as the PATH variable

Countdown timer

15:00

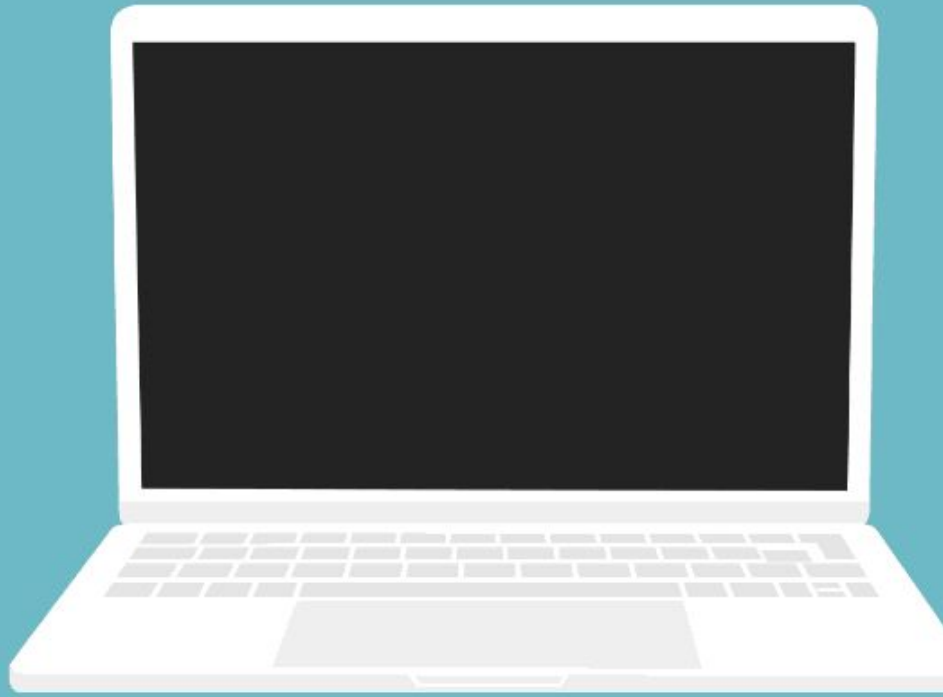
(with alarm)

My First Bash Script

Variables

A bash script is an executable file that contains a series of commands.

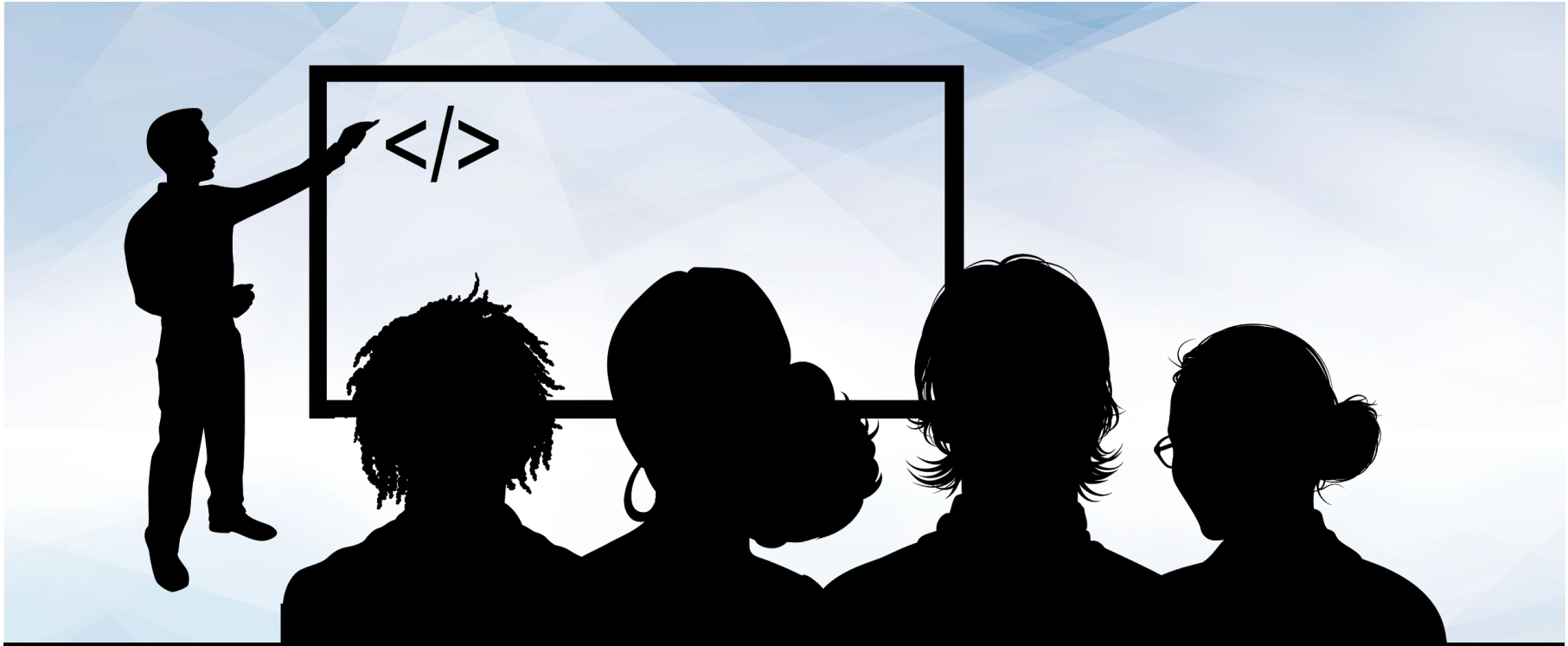
- When the script is executed, these commands will run one by one until they are all executed.
- A fundamental system administrator skill is creating a bash script and then scheduling it to run at a regular time using cron.



Variable Demo

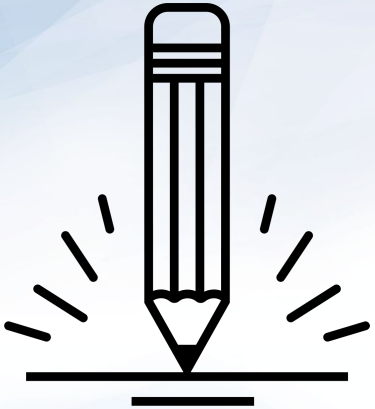
In the following demo, we will use:

- Basic Variables
- Built-In Variables
- Common expansion
- Variables in Scripts



Instructor Demonstration

My First Bash Script



Activity: My First Bash Script

In this activity, you will work in groups of two to create a script that completes several system audits steps automatically.

Suggested Time:
20 minutes

Activity Instructions: My First Bash Script

Complete the following set-up:

- Create a new script file called `sys_info.sh`.
- Change the permissions on the file to make it executable.
- Open the file with nano.
- Add a top hashbang line to make this a bash script.

Your script should output the following data:

- A title and today's date.
- The uname info for the machine.
- The machine's IP address. (Narrow this output down to one line.)
- The Hostname.
- The DNS info.
- The Memory info.
- The CPU info.
- The Disk usage.
- The currently logged on users.

Run your script using `./` notation.

Suggested Time: 20 Minutes

Activity Review: My First Bash Script

Set-up:

Create a new script file.	`touch sys_info.sh`
Change the permissions on the file to make it executable.	`chmod +x sys_info.sh`
Open the file with nano.	`nano sys_info.sh`
Add a top hashbang line to make this a bash script.	`#!/bin/bash`

Review continued on next slide.

Activity Review: My First Bash Script

Add the following to your script:

Create a new script file.	`touch sys_info.sh`
Change the permissions on the file to make it executable.	`chmod +x sys_info.sh`
Open the file with nano.	`nano sys_info.sh`
Add a top hashbanf line to make this a bash script.	`#!/bin/bash`

Review continued on next slide.

Activity Review: My First Bash Script

Add the following to your script:

A title	`touch sys_info.sh`
Today's date	`chmod +x sys_info.sh`
The machine's type	`nano sys_info.sh`
The uname info for the machine	`#!/bin/bash`
The machine's IP address	`echo -e "IP Info: $(ip addr | head -9 | tail -1) \n”`
The Hostname	`echo "Hostname: $(hostname -s) "`
The DNS info	`echo "DNS Servers: "` `cat /etc/resolv.conf`

Review continued on next slide.

Activity Review: My First Bash Script

Add the following to your script:

A title	`echo “A Quick System Audit Script”`
Today’s date	`date`
The machine’s type	`echo “Machine Type Info:” echo $MACHTYPE`
The uname info for the machine	`echo -e “Uname info: $(uname -a) \n”`
The machine’s IP address	`echo -e “IP Info: $(ip addr | head-9 | tail -1)\n”`

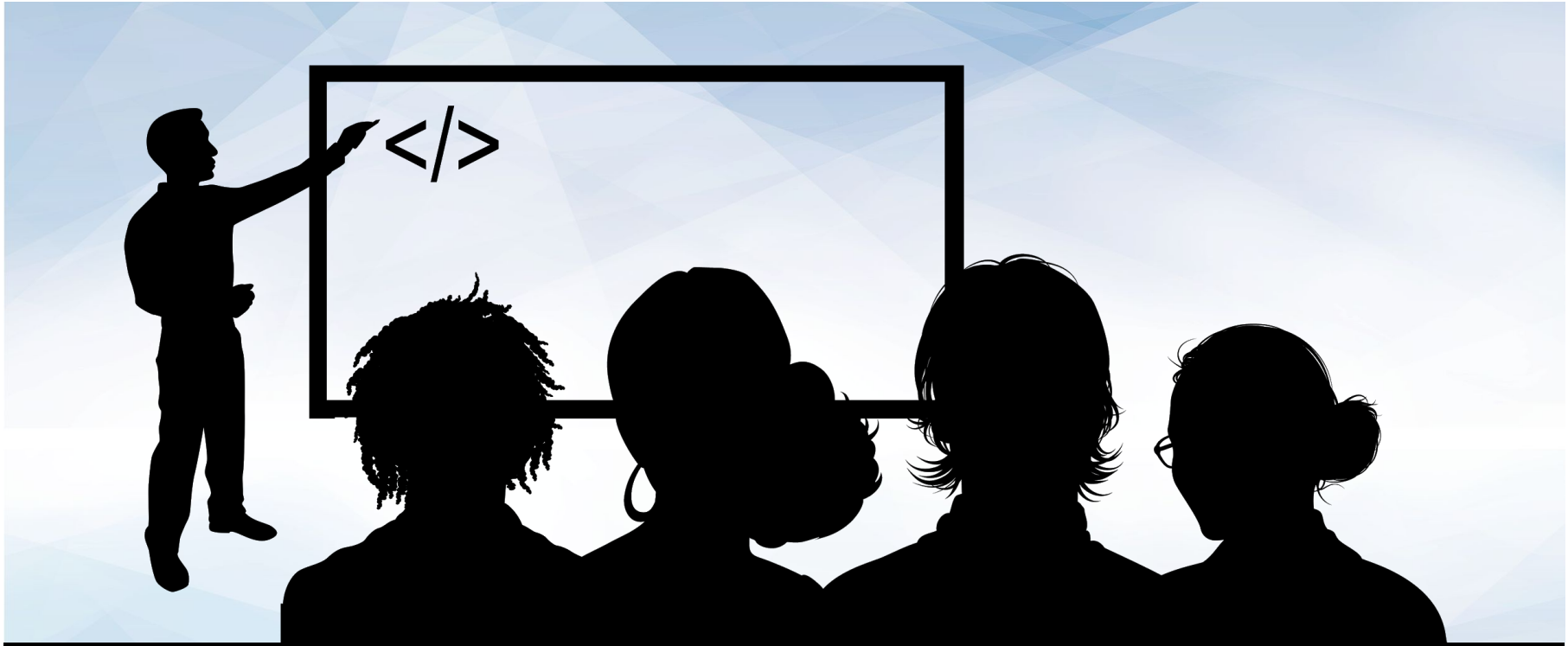
Review continued on next slide.

Activity Review: My First Bash Script

Add the following to your script:

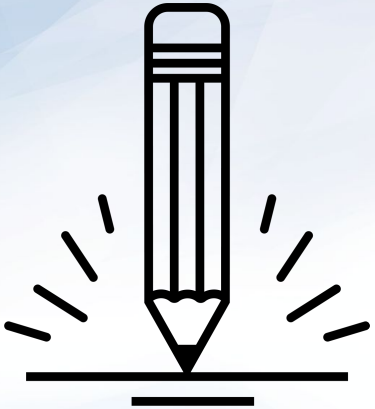
The Hostname	`echo -e “Hostname: $(hostname -s)”`
The DNS info	`echo “DNS Servers:” cat /etc/resolv.conf`
The Memory info	`echo “Memory Info:” free`
The CPU info	`echo “\nCPU Info:” lscpu | grep CPU`
Disk Usage	`echo “\nDisk Usage:” df -H | head -2`
Currently logged on Users	`echo “\nWho is logged in: \n $(who -a)\n”`

Custom Commands



Instructor Demonstration

PATH



Activity: Custom Commands

In this activity, you will continue to add more commands into your script. Then, you will save the script to a directory which will be added to your $PATH.

Suggested Time:
15 minutes

Activity Instructions: Custom Commands

Complete the following inside your script.

- Add the command for creating a `~/research` directory to your script.
- Add the command for finding `SUID` files to your script.
- Add the command for finding the Top 10 processes to your script.
- Modify each command of the script so that it writes all output to a file called `~/research/sys_info.txt`.

Complete the following in your command line environment:

- Manually create a `~/scripts` directory and save your script there. (This should not be part of your script).
- Add your `~/scripts` directory to your `$PATH`.
- Reload your bashrc file.
- Run your script.
- Open `~/research/sys_info.txt` and verify it has the desired output.

Suggested Time: 15 Minutes

Activity Review: Custom Commands

Inside your script:

Add the command for creating a `~/research` directory to your script.	[redacted]
Add the command for finding `SUID` files to your script.	[redacted]
Add the command for finding the top 10 processes to your script.	[redacted]
Modify each command of the script so that it writes all output to a file called `~/research/sys_info.txt`.	[redacted]

Review continued on next slide.

Activity Review: Custom Commands

Inside your script:

Add the command for creating a `~/research` directory to your script.	`mkdir ~/research 2> /dev/null`
Add the command for finding `SUID` files to your script.	`echo "\nSUID Files:" >> ~/research/sys_info.txt` `find / -type f -perm /4000 >> ~/research/sys_info.txt`
Add the command for finding the top 10 processes to your script.	`echo "\nTop 10 Processes" >> ~/research/sys_info.txt ps aux -m | awk {'print $1, $2, $3, $4, $11'} | head >> ~/research/sys_info.txt`
Modify each command of the script so that it writes all output to a file called `~/research/sys_info.txt`.	`>> ~/research/sys_info.txt to each line of your script.`

Review continued on next slide.

Activity Review: Custom Commands

Inside your script:

Manually create a `~/scripts` directory and save your scripts there.	[redacted]
Add your `~/scripts` directory to your `$PATH`.	[redacted]
Reload your bashrc file.	[redacted]
Run your script.	[redacted]

Review continued on next slide.

Activity Review: Custom Commands

Inside your script:

Manually create a `~/scripts` directory and save your scripts there.	`mkdir ~/scripts && cp sys_info.sh ~/scripts`
Add your `~/scripts` directory to your `$PATH`.	`echo “export PATH=$PATH:~/scripts” >> ~/.bashrc`
Reload your bashrc file.	`source ~/.bashrc`
Run your script.	`sys_info.sh`

Questions?