**LARK PROJECT**

*Name*:

*Narrative*: So, it’s present day, and you are working at home on your computer, and you get a suspicious email. The email contains a message that states that a series of missiles will go kaboom! in twenty-four hours from now if you do not enter the correct code to stop it. You think that the email is real but decide to ask trusted individuals for their two senses. They tell you that the email is just malware and to not worry about it. However, you don’t want to be responsible for the world ending so you decided to do as the email says. But since you took time to ask around now there is only 15 minutes left until the missiles are scheduled to go off. You download putty and log into the ember lab and continue to follow the instructions in the email.

*Purpose*: The purpose of the game is to stop the supposed “missile strike” in 15 minutes.

*Topics*:

* Vim editing a file
* Cat
* How to run a bash script
* Echo
* Grep
* ls -a
* gawk
* chmod
* head/tail
* man
* /dev/null

*Final Code:* MELISSA

*File Names:*

* File 1 (grep) = “firstClue”
* File 2 (tail) = “linux”
* File 3 (bash script with argument) = “e4g62”
* File 4 (chmod) = “gnd53”
* File 5 (vim) = “fm219”
* File 6 (gawk) = “editor”
* File 7 (/dev/null) = “someFile”

*Instructions*:

* Make a game directory that has readme.txt file and game files.
* Include in the readme.txt basic commands instructions (ls,cat,cd, man)
* Instruct the user to have a certain size window.
* Instruct user to run the script so the clock starts, and an echo states the instructions again
* 1. Instructs the user to find file “.firstClue” using the man pages to use the -a option. In the “.firstClue” file there is a lot of random stuff but the user has to enter a tail command to see the instructions for the name of the second clue file and the first letter of the final code.
* 2. The second clue file will include a lot of random stuff and the user needs to grep for the answer to the question “The Linux operating system originated from research and has roots in this operating system?” and provide a hint. Once they grep for the file, they will see the instructions to the name of the third clue file and the second letter of the code.
* 3. The third clue file will include the instructions to determine the code and run the script with the code as the argument. Once the script is run it will produce the name of the fourth clue file and the third letter of the code. (Demonstrates how the final code will work)
* 4. The fourth clue file will include the name of a script they need to run but they won’t be able to because they need to use chmod. Give instructions for chmod and how to view the file permissions. Once the script is run it will produce the name of the fifth clue file and the fourth letter of code.
* 5. The fifth clue file will be a vim file with a scramble of letters where the user needs to use insert mode and use ‘x’ to delete the unnecessary letters. The answer will be the name of the sixth clue file and the fifth letter of code will be in the vim file. (do not use “!wq”)
* 6. The sixth clue file will be a gawk program where the user will need to use gawk and the print utility to get the name of the seventh file and the sixth letter of the code. (example: $gawk ‘{print $3, $1}’)
* 7. The seventh and final clue final will be a script that tests if a file is equal to /dev/null and the user will need to edit the file to make the file equal to /dev/null to get the seventh letter of the code.

Brayden: file 1 (letter: M), file 4 (letter: I), file 6 (letter: S), file 7 (letter:A)

Justine: Readme file, file 2 (letter: E), file 3 (letter: L), file 5 (letter: S)

*Need to Do:*

* Create a script with a timer and when the timer goes off the lose result displays and returns the user to the beginning of the game. (chmod needs to reset for file4)
* Create a win script that can be accessed at any time and displays win when they enter the correct code and displays lose when they enter the incorrect code and returns the user to the beginning of the game. (chmod needs to reset for file4)