AstroJellies
Jeffrey Weng, Federico Tondolo, Allan Wang

APCS1 pd5
1/9/2017**UPDATED

Final Project Proposal

# Shall We Play A Game?

"Shall We Play A Game?" is a puzzle adventure styled game, using unix based commands plus custom in-game commands to hack your way stealthily into a secure research facility hidden behind multiple obstacles.

Several things come into play when you start the game. Using only your wits, a few hacking tools, and puzzle solving skills you will be tasked to hack into one of the securest facilities in the world, NZT, a leading research facility for the colonization of Mars. Your job after you get into their central network is to sabotage their plans and illegally launch their space shuttle, Terraferma. You'll be completing 3 puzzles/challenges, that are hidden throughout the terminal, finding hidden clues/secrets, to find out separate pieces of an overall LOCK. Each challenge can also provide tips/hints for other challenges.

Several basic tools and abilities have been given to the Hacker to use(Not Definite):

**The Decryptor:**

Where decrypted blocks of text can be decoded using either a brute force method or a KEY found somewhere in the game. Most likely will be encrypting messages using a Caesar Cipher. We'll be using basic: conditionals, loops, string manipulation, overloading methods, and reading of files to achieve this goal. Basic outline:

Decrypt Message: <Enter Message> (Message can be a file or the actual message)
Brute Force(1 ) or Specify Key(2)? <Input>

Brute Force(Uses up time, we may insert a timed events)

- Lists all 26 possible possibilities

Key

- Prints out the message using this key

**Network Manager:**

Our entire game would be based around a simple map, where connected computers/hidden computers are shown. You'll probably start off in the enemy network, trying to gain access to a computer that is securely locked. The network manager will use 2D array, that prints out a list of computers that can be accessed. You'll have to insert the specific coordinate (x[rows], y[columns]) to attempt to access. It'll then loop through the 2D array. We can also use ArrayLists if possible.

Basic Outline(0s and 1s are just aesthetics, can be used effectively later on, the random string of characters are the actual computers):

0 0 0 1 0 0 1 0 0 %$# 0 1 1
0 1 1 0 #@t 1 0 0 0 0 0 0 1 (Computer #@t, located at coordinate(2, 5), (2, 6),(2, 7))
0 0 0 0 0 0 0 0 &UR 0 0 0 0
1 1 1 1 0 0 0 0 1 0 1 0 1 0 0
0 0 0 0 0 0 0 0 &UR 0 0 0 0
0 0 0 1 0 0 1 0 0 %$# 0 1 1

**The Interface**

More specifically, it's an in-game file directory. Nothing super special here. Files, directories, and executables, each will have an ID assigned to the file. ID will be used to find out how to open. For example: all directories will have the same ID, which basically tells us, when we press this directory, we will open up folder into another layer of the interface. Different files can be opened in different ways, in general, you won't be able to open them unless you use some kind of unix command, but basic text files can be open. Executables are the same, but they don't have text, instead they launch programs, which can be harmful or not harmful.

We'll be using basic conditionals, while loops, calls to certain classes, and lots of printing.

**Virus Creator**

The ability to create viruses. We'll be setting up a basic blueprint for the creation of a virus in an interface which will hold all the basic methods of a virus. Further viruses will be created, implementing the blueprint class. Viruses will be used to gain access to certain systems and used to gain information. Can be deployed into locked systems by finding vulnerabilities.

**Tools**

A superclass called tools, that allows, in-game commands, however complex, to inherit basic properties from superclass Tools.

**MISC**

- Obtaining Keyboard input from the user constantly
- Maybe a File Reader?
- Tip Bot, so you won't get lonely :), and so you can get help of course
- Sorting Algorithms can be used to help implement tools stated above
- Special Parsing
    - Look through text,images(ASCII ART), links, etc to find clues
    - Special parsing, could showcase locations of weakness in file, highlight important info, easy navigation of file

***Aesthetics***

Pound sign loading bar, found this using homebrew....

To progress through the game, you'll need to use the tools provided and your ingenuity/creativity. When all the puzzles have been completed, you'll be directed towards an admin computer in the facility, and be prompted to use all the information you gathered to unlock the final lock. At this point, there will be no further guidelines nor help from the Tips Bot :(, the mission is all up to you the player.

In the case that you are discovered, you'll have to pass smaller quick-time events/challenges in order to continue hacking. We'll be applying some features that'll demonstrate actual hacking to obtain needed in-game information(password hints, plot information, and unprotected, easily accessible in-game computers in the network).

This game is based around a theme similar to games such as Hacknet, Hack run, and Welcome to the game(a game about surfing the deep web), where the player spends a considerable amount of time time looking through data to find essential tools for winning. This game is also based around CTF competitions, where you need to "capture the flag" hidden in websites, images, text, etc. Finally, some further research needs to be done on "actual" hacking and other puzzle elements that we want to include in our game.

The first step in this journey would be to create the basis of the code: the tutorial. The first so-called puzzle which will introduce the player to all the tools at his disposal and the aim and storyline of the game.The beginning of the tutorial would use ASCII messages about the geniuses behind the code and the title. Once the username is selected through simple text-input the user is informed of the backstory(through print statements), and is supplied with a terminal-like interface with which the player can run terminal commands such as "help" (to explain the tools he has at his disposal) and other unix commands such as ls, cd, cat, among others. This would include multiple conditional statements considering the textual input supplied by the user. After the tutorial, the game then proceeds to the first scene, where the end goal of the game will be displayed. The tutorial is the backbone of the game, all the rest of the parts of this "car" are based around the methods and algorithms utilised in the tutorial therefore finishing the tutorial first makes it infinitely easier to complete the rest of the project.