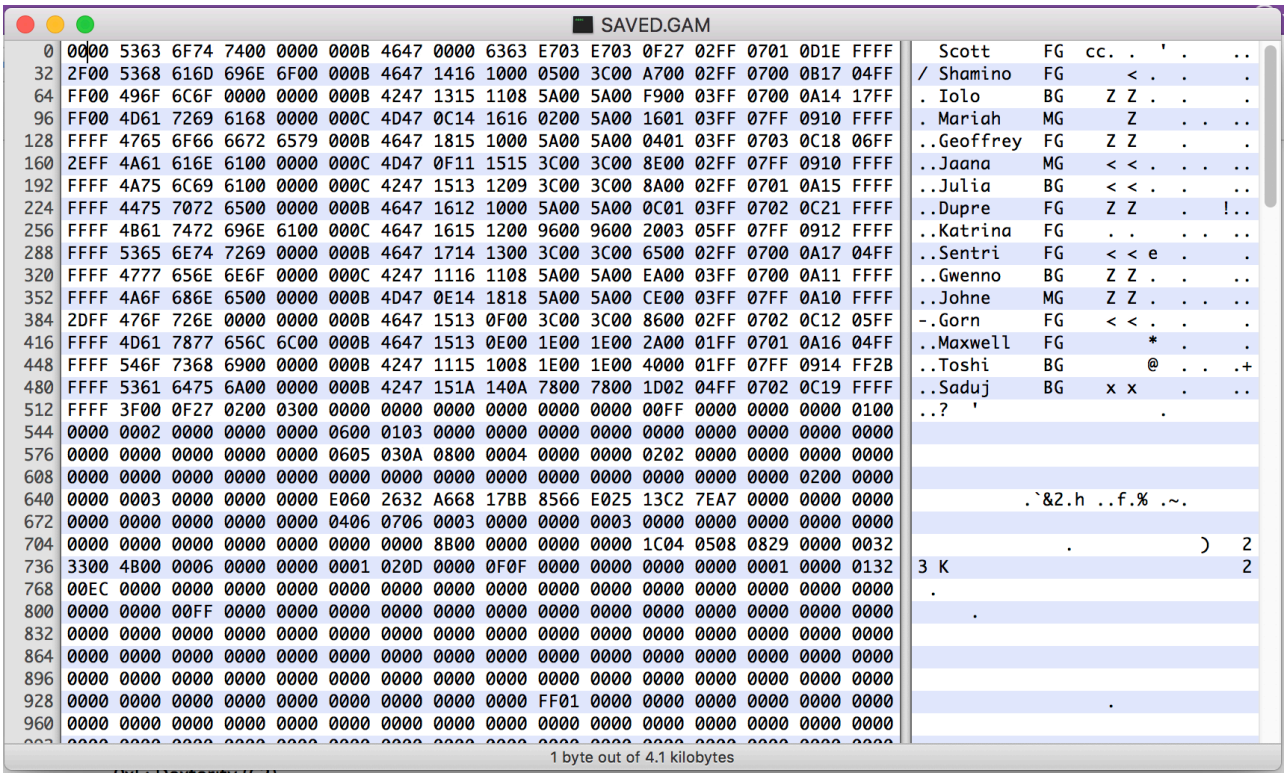
Ultima_5 Write Up

Monday, March 5, 2018 4:12 PM

Use CTL-E to exit from game to DOS

1. INIT.GAM has all the character names from start
2. BRIT.OOL was wiped after:
    a. Torch item use
    b. Save game
3. SAVED.GAM changes after:
    a. Making a full new character and returning to the start menu
    b. Pressing 'Q' in game



For this picture: Byte Offsets

2/3: represent beginning of the name

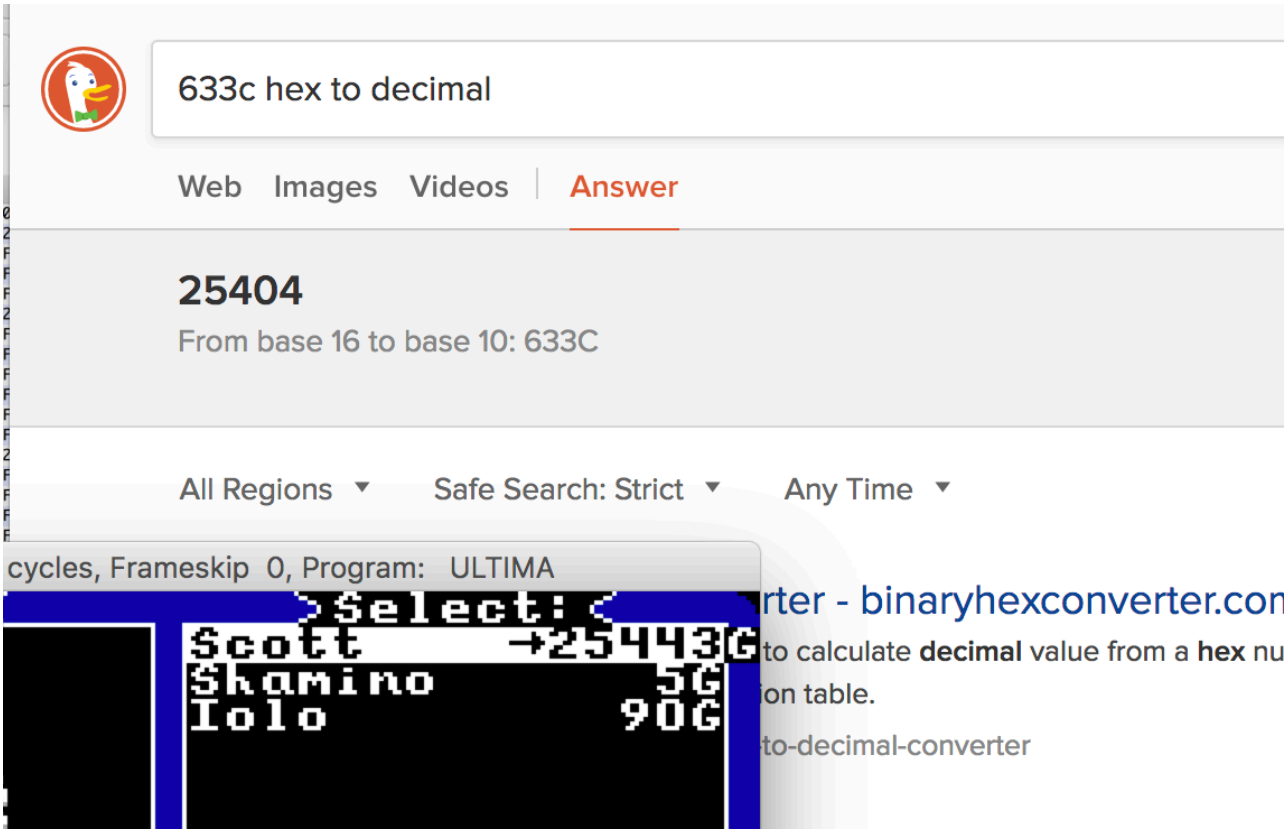
4/5/6/7: Name

8/9/10: Name

0xB: has 0B or 0C for some reason (probably some terminator)

12/13: Character class
14: Strength (63)
15: Dexterity (63)
16: Intelligence (63)
17: Magic (63)
18: HP1 (E7)
19: HP2 (03)
20: HM1 (E7)
21: HM2 (03)
22: EXP1 (0F)
23: EXP2 (27)
516: Gold1 (0F)
517: Gold (27)

For calculating hp… I entered 63633C on the internet and compared it to 63633C for Ultima_5



The system is big endian: (check in the health point bytes to confirm)
6300 is 99 (base 10)
0063 is 25344 (base 10)

E703 is 999 (base 10)

Python Malware Attempt:

To simplify the file manipulation, I kept a Clean and Manip version of Ultima_5.

Script Methods:

- **isIdenticalFiles():** The script detects when files have changed by comparing the Clean and Manip directories. This was the script I used to find the changed files when I messed with the saves. The only important file that we need to manipulate is the **SAVED.GAM**
- **openTempFile():** If you want to work off a temporary file in the manipulated folder, this allows you to open the SAVED.GAM file and create a temporary files.
- **modMainCharacterStats(decList):** This works directly with the Main Character's offsets to manipulate the decimal values provided by decList.
- **modSaveState():** Opens, reads, manipulates, writes and closes the SAVED.GAM file. This is the method used when running the program. Mocks a "Main" method.

Shell Attempt:

My idea of a traditional malware uses the computer's shell commands to run the program without having to use input/output readers.

I went through the shell commands with you and they worked in MacOS terminal. When I tried to run the commands through a Python script, the files would not change when I would write a new file.

I/O Finish:

I took the easy way out and read the file through a simple file open. The hard part was converting the file to a mutable format. I chose decimal because the values are converted to decimal when you append the bytes to a list.

Then, to write the files back, you need to change a decimal list to a byte array. You can use bytes(list) to do this.