**ESCAPE ROOM IDEAS**

**Overview**

When a user logs in, he is greeted with a text file and a binary file:

A txt file called Welcome to the Dungeon that includes a passphrase to unlock cell 1.

A binary file written in Python that gives the user the option to choose cell 1, 2, or 3.

Each cell will have to be completed in numerical order to escape the dungeon. User will realize this in short order. A shortcut to the ending could be obtained by typing “no” instead of a cell option.

Each cell asks the user for a passphrase. The first passphrase to enter cell 1 is in the txt file.

The second passphrase is a username in cell 1. The third passphrase is a password in cell 2.

The second and third passphrase used together allows the user to switch to a more privileged user.

The escape key is located in cell three. The user can use the password from cell 2 to enter cell 3 but must switch users to access the clues in cell 3.

If possible, there may be a countdown timer that kicks the user out of the dungeon if not completed in time.

**Cell 1**

Upon entering the first passphrase, the user will be sent to a different directory called Cell\_1.

There will be a readme file that explains the objective.

The directory includes a compressed directory that the user must decompress.

The decompression reveals a massive directory tree with a binary file hidden within it.

There is also a hidden file called .hint that contains a command that would move user to the location of the binary file if executed on the command line.

The hint command would be encoded in hexadecimal and then in base64.

When the binary file is finally executed, it will reveal a double encoded username and the user will be brought back to the original directory.

User will have to use the decoded username to enter cell 2.

**Cell 2**

After decoding the username, the user will run the binary again and choose cell 2.

He will then be prompted to enter the decoded username from cell 1.

When correctly entered, the user is sent to a new directory called Cell\_2.

This directory contains a txt objective, a readable and executable Python file, and an unreadable txt file.

In order to read the unreadable file, the user must run the Python file with the unreadable txt as a sys.argv argument.

The user will only realize this if he opens and reads the Python file or gets lucky.

Executing the Python file with txt file as the argument will reveal a hashed password and the path to a wordlist.

User will have to use john the ripper and the wordlist to crack the password.

Once the password is cracked, the user will be sent back to the main directory.

User will have to convert the password to leet speak and rerun the binary and choose Cell 3.

When prompted for the passphrase, the user will input the leetspeak version of the password.

**Cell 3**

Upon entering the correct passphrase, the user will be sent to a Directory called Cell\_3.

Any attempt to run or execute any files in this directory as user:Guest will be denied.

Guest user will have to switch user with username from Cell 1 and password from Cell 2.

Only the new user will be able to execute commands in this directory.

After switching users, user will be able to LS the directory to see the files.

The directory will contain one file with instructions to download another file using wget.

Using wget and the url in the file, the user will download a zipped file.

Unzipping the file will reveal a binary file, a pcap file and a txt file with additional instructions including a hashed password.

User will have to use wireshark to run the pcap file.

User will have to search the pcap file for packets that contains passwords in the clear.

User will have to hash all the passwords from the pcap to find the one that matches the hash from the instructions.

After user finds the correct hashed password, he will be instructed to run the binary file.

The binary will first prompt user to input the source IP for the password. If correct, the user will then be prompted for the username associated with the password.

If user gets both correct, the escape key will be displayed and he will be sent back to the home directory where a new binary file called Escape will be available to execute.

Running the Escape binary will prompt the user to enter the escape key.

Upon entering the correct escape key, User will be met with a congratulatory message and the option to enter his hacker handle to be included on the Wall of Fame.

**To Do List**

**Staging Area**

Download a kali box and configure it so random users can ssh into it. (The Hardest Part)

Change root password.

As root, create a user: guest, password: guest.

Change guest /Home directory name to /Dungeon.

In guest Dungeon directory, create a txt file called Welcome\_to\_the\_Dungeon with only read permissions for guest.

In guest Dungeon directory, create a python file that can be executed from the command line by guest but no other permissions.

The python script will give guest the option to choose cell 1,2, or 3, followed by a prompt for a passphrase. An incorrect passphrase will prompt guest to try again and tell them where the passphrase for that cell number is located. Typing “no” instead of a cell number will reveal the escape key and the final binary file needed to escape the dungeon without the prerequisite passphrases.

When guest chooses a cell number with the correct passphrase, the script will change guests permissions and cd into a directory called Cell\_#.

Remove all guest permissions except the ability to cat Welcome\_to\_the\_Dungeon and execute the first binary file.

**Cell 1**

As root, create a directory called Cell\_1 in the root /Home directory.

Make sure the passphrase from the txt file works with the binary.

Create a txt file with only read permissions for guest that explains the objective of this cell.

Create a hidden txt file called .hint with the command line tool needed to find a binary file hidden within the directory. The actual message will be encoded with hex and then base64.

In /Cell\_1, use httrack to download a deep directory website to hide a binary file.

Give guest permissions to search and traverse the directories of the file freely.

In the page 6 directory, create a python executable file called New\_User.

Make sure guest has permission to execute the file.

When executed, the display shows a base64 encoded message and plain text message that tells guest that this is the passphase for Cell\_2. It also suggests that the passphrase may not work as written, and finally sends guest back to the /Dungeon directory.

Compress the httracked directory with tar and gz.

**Cell 2**

Create a directory in root home directory called/Cell\_2.

Make sure the passphrase from Cell\_1 works with the /Dungeon binary and changes guest permissions to have access to Cell\_2.

Create a txt file with only read permissions for guest that explains the objective of this cell.

Create a txt file with no guest permissions. Guest should not be able to cat this file.

The no permission txt file will contain a message for the guest that this file contains a hashed password and a path to a wordlist. It should also give a hint to use john the ripper. Finally it should state that the guest should run the binary with both the txt file and the cracked password as an argument.

Make sure guest has permission to access the wordlist.

Create python file that is executable for guest.

The python code, when run with the unreadable txt file as an argument, should display all lines except the first line of the txt file.

If the code is run without and argument, it should state that the code needs at least one argument to run properly.

If the code is run with both the txt file and the cracked password, it should display a message that says the password must be changed to leetspeak to open Cell\_3 and send the user back to /Dungeon.

Finally, running the code with “no” as an argument will be another shortcut to the escape key.

**Cell\_3**

Create a directory in root home directory called/Cell\_3.

Make sure the passphrase from Cell\_2 works with the /Dungeon binary and changes guest permissions to have access to Cell\_2.

Guest should only have X permissions for Cell\_3 and no permissions at all for the files in Cell\_3.

Create a user called mad\_king, password: qwerty or qw3r+y.

mad\_king is the username from Cell\_1 and qwerty is the password from Cell\_2.

Mad\_king should have owner or group permissions for all files in Cell\_3.

Create wireshark group, and add mad\_king to wireshark group.