ESCAPE ROOM

Written and designed by:

Taylor Greenblatt, Kade Griffon, and Kaleb Smith

Based on an original idea by:

Kaleb Smith

https://github.com/TippJar175/Escape-Room

tgreenb04@gmail.com

kadegriffon@yahoo.com

kalebsmith2551@yahoo.com

FADE IN

INT. ZOOM MEETING ROOM - DAY

Taylor Greenblatt, Kade Griffon, and Kaleb Smith sit idle in a Zoom meeting room. An introduction slide appears on screen showing the project name, group members, and a link to the Escape Room github.

KALEB
Welcome to the Escape Room Project by Taylor Greenblatt, Kade Griffon, and myself, Kaleb Smith. The idea of this project was to create a fun, fantasy-themed, CTF-style game, where users log in remotely via SSH and gather several clues that allow the users to progress through the game and make their escape. We built this project in the latest version of Kali Linux using Python coding and user-access restrictions within this Linux environment. In order to play the game, users will have to download the OVA image that we supplied on the project github, which can be found on the first and last slides of this presentation.

The slide changes to a screenshot of the project README file.

KALEB (CONT'D)
Once they have downloaded the OVA, the user will have to spin up the machine and their own version of Kali Linux and run an NMAP scan to find the IP address of the Escape Room machine. We enabled the ssh service to start automatically, so the user should be able to log into the machine via ssh with the log in credentials provided in the project README file, which can also be found on the project github, and as you can see on the screen is "guest" for both username and password. The user may also have to change their settings on the new OVA to "host only" to successfully log in.

The slideshow disappears, and Kaleb becomes full screen.

KALEB (CONT'D)
And with that introduction, I'll hand it off to Kade to provide a birds-eye view of the game objectives.

The video changes to Kade.

KADE
Thank you Kaleb.

A slide appears on screen showing a screen capture of Taylor running the ssh command and then the ls command in the /home/guest directory, which shows the Key_Master.py executable and Welcome to the Dungeon text file.

KADE (CONT'D)
Great, so once the user logs in, they will be in the home/guest directory, which contains two files: An executable called key master and a text file called welcome to the dungeon. They won't be able to see these files by default, so they'll have to run the ls command to display them. The text file includes some general instructions about the game and the executable prompts the user to choose one of three cells.

The slide changes to a screen capture of Taylor running the Key_Master file.

KADE (CONT'D)
All the cells contain a Welcome file and should be read first. In order to escape, the user will have to find a passphrase hidden within each cell, and each cell is located in a different directory. However, the user will not be able to enter any cell without knowing the passphrase for that cell, and the passphrase for each cell is hidden within each previous cell, so the cells must be unlocked chronologically. The first passphrase is hidden somewhere in the home/guest directory, which we call the dungeon.

The slide changes to a screenshot of Cell 1 showing the Solitary_Confinement Directory and Welcome file.

KADE (CONT'D)
Once the user finds the first passphrase, they will be provided with the location of the first cell. As you can see, this cell contains a child directory and the ubiquitous welcome file. Hidden somewhere within the Solitary_Confinement directory is another executable file that the user must find and run to obtain the passphrase for cell 2. After locating the hidden file, the user will return back to the dungeon and provide the Keymaster the passphrase to unlock the second cell.

The slide changes to a screenshot of Cell 2 showing the High_Valyrian, The_Citadel, and the Welcome file.

KADE (CONT'D)
The objective of the second cell is to decrypt a hashed word that we placed in the file called High_Valyrian using a very specific tool provided by Kali Linux. Once the word has been decrypted, the user will be able to input it into the citadel program. They will then be provided the passphrase to unlock the third and final cell.

The slide changes to a screenshot of Cell 3 showing the Cyber_Shark, Shark_byte.pcap And Welcome file.

KADE (CONT'D)
And this is where things get really dicey. The objective of this challenge is to perform a packet capture analysis on a pcap file called shark byte to find a password that was sent in the clear over the network. The problem for the user, however, is that the tool necessary to perform the analysis doesn't work over ssh. So the users will have to figure out a way to perform that operation outside of this controlled environment. I assure you, it can be done.
(MORE)

KADE (CONT'D)
They will be able use the Cyber shark executable to help them on their quest, and once the correct password from the pcap has been located, they will be able to run a new executable in the dungeon to escape.

The slide changes to a screenshot of the home/guest directory that shows the new executable file.

KADE (CONT'D)
There it is. Escape dot p-y.

The slide changes to a screenshot of the home/guest directory that shows the new executable file after Taylor runs it.

KADE (CONT'D)
There is one final challenge for the user, and solving it will provide a congratulatory message before killing the ssh connection, but that's how the game is played. And just for fun, we also coded a short cut into each executable that allows the user to skip to the end of the game. Good luck with that.

The last slide disappears, and Kade changes to fullscreen.

And with that, I'll hand it off to Taylor for closing remarks.

Kade disappears, and Taylor comes into view.

TAYLOR
Thanks Kade.

The first slide showing the project name comes back into view.

TAYLOR (CONT'D)
So what did we learn? This project was challenging because it included a lot of moving parts within each python executable and the user-access restrictions proved to be quite tricky and problematic. However, we were able to solve most of the issues we faced, and the ones we couldn't solve, we were able to incorporate into the game and make them a challenge that the user would have to figure out.
(MORE)

TAYLOR (CONT'D)
For instance, going in, we didn't know that our packet capture analysis tool that we want the user to access in Cell three doesn't work over ssh. But we did find a way to make it work using a different method outside of the escape room environment, which we thought would be an interesting challenge for the users to figure out on their own. It also made the last cell significantly more difficult to solve, which, progressive difficulty is the standard operating procedure for every game ever made. So it was kind of a blessing in disguise. Aside from that, I think we definitely improved our knowledge of Python and least privilege. For example, none of us knew how to call linux commands within a python script. Now we know, and it's very simple. But not every command can be called, and we now know of a few that can and a few that cannot, like changing directories. We also got a nice crash course in how to change and manipulate different user, group, and world permissions so that only certain users could access different commands, tools, and directories. It was a fun challenge, and we look forward to people playing the game, and we hope they enjoy playing it as much as we enjoyed building it. Good luck everyone. And thanks for tuning in.

CUT TO BLACK

THE END