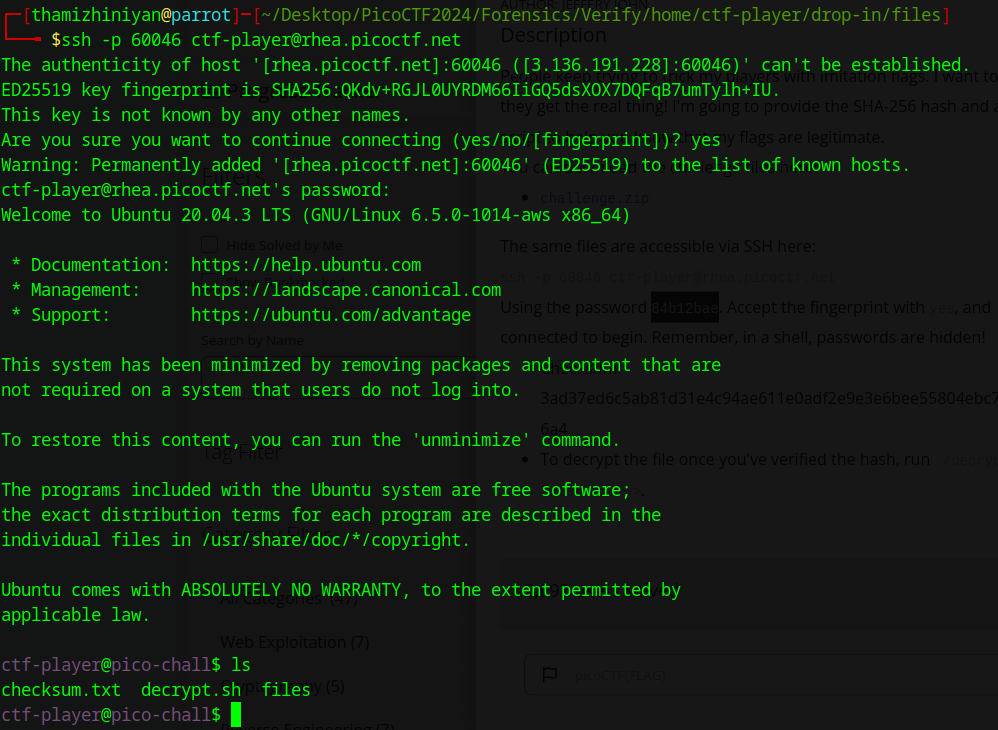
Challenge Description

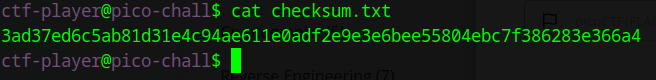
People keep trying to trick my players with imitation flags. I want to make sure they get the real thing! I'm going to provide the SHA-256 hash and a decrypt script to help you know that my flags are legitimate.

Solution

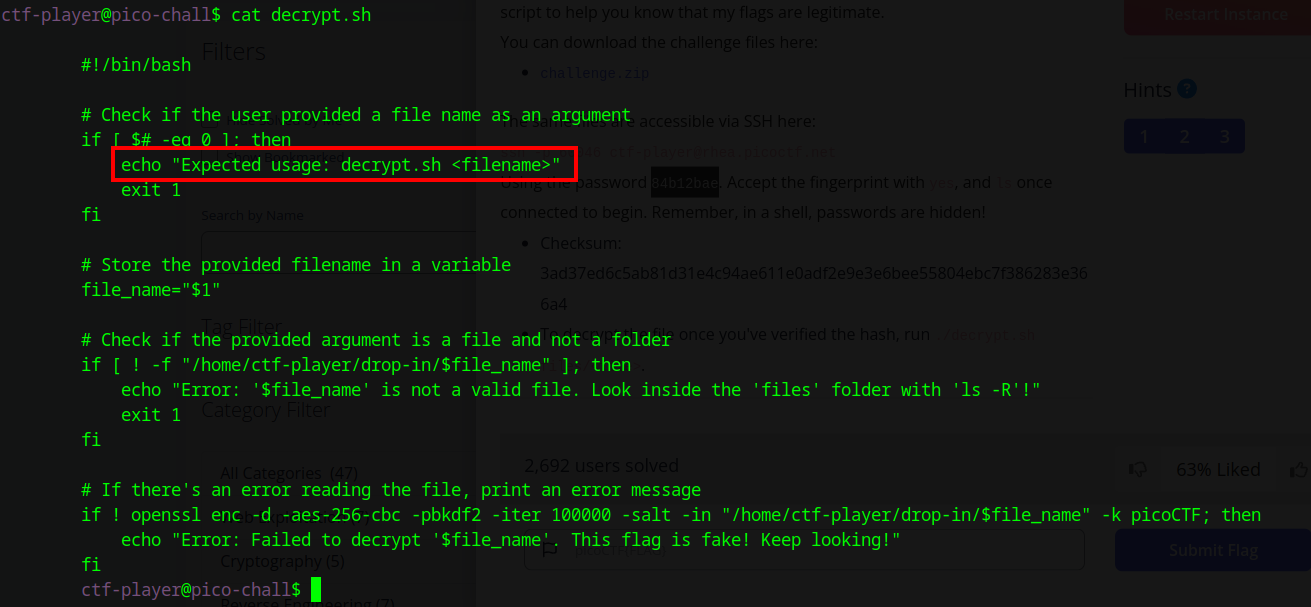
First start the instance and login via ssh to the given instance. We are given with two files and a folder.



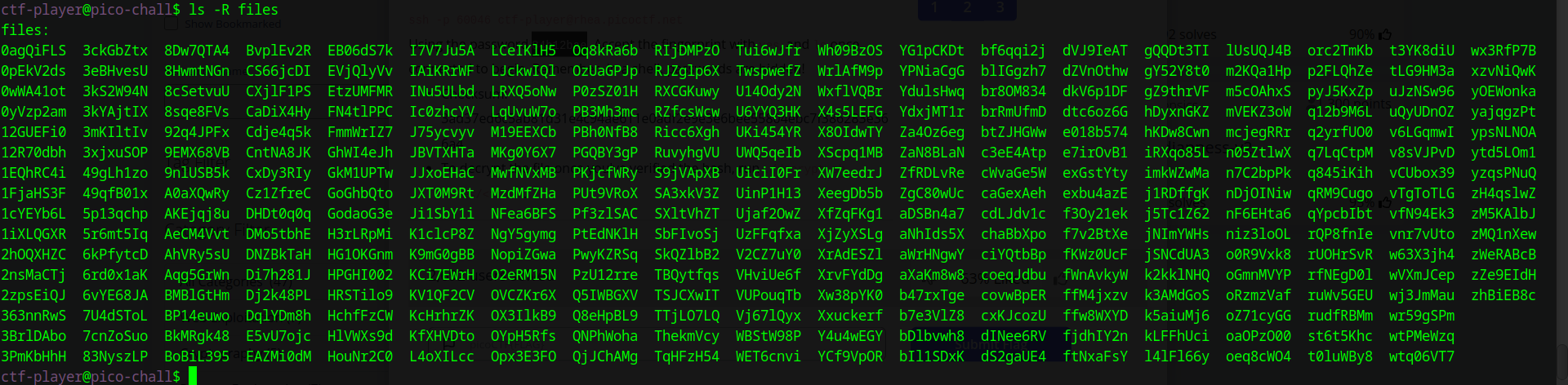
The checksum.txt file contains the hash sum of some file.



The decrypt.sh file is a bash script, which tries to decrypt a file.

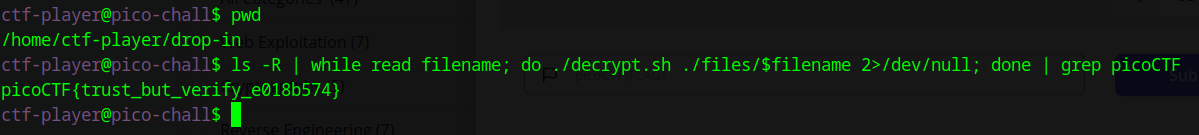


Next I checked the files directory. It has a lot of files. One of these files has the flag in encrypted form. We have to decrypt that file using the decrypt.sh file. We have to decrypt each file in this directory to find the flag.



I used a bash while loop to get the job done. Use the following command to get the flag.

ls -R | while read filename; do ./decrypt.sh ./files/$filename 2>/dev/null; done | grep picoCTF



Flag: picoCTF{trust\_but\_verify\_e018b574}

**Breakdown**

1. **ls -R**:
   * ls: This command lists the contents of directories.
   * -R: This option tells ls to list directories recursively, meaning it will list all files and subdirectories within the current directory and its subdirectories.
2. **| (Pipe)**:
   * The pipe (|) takes the output of the command on its left (ls -R) and passes it as input to the command on its right (while read filename; do ... done).
3. **while read filename; do ... done**:
   * This is a loop that reads each line of input from the previous command (ls -R) and executes the commands inside the loop for each line. Each line read is assigned to the variable filename.
4. **./decrypt.sh ./files/$filename**:
   * ./decrypt.sh: This runs the decrypt.sh script located in the current directory.
   * ./files/$filename: This provides the path to the file that will be decrypted by the script. $filename is the variable holding the current filename being processed in the loop.
5. **2>/dev/null**:
   * This part redirects any error messages (standard error stream) produced by decrypt.sh to /dev/null, effectively discarding them. This keeps the output clean by only showing relevant results.
6. **| (Pipe)**:
   * Another pipe takes the output from the loop (i.e., the result of running decrypt.sh on each file) and passes it to the next command (grep picoCTF).
7. **grep picoCTF**:
   * grep: This command searches for lines that match a specified pattern.
   * picoCTF: This is the pattern being searched for. In this context, it’s looking for any output from decrypt.sh that contains the string picoCTF, which is likely part of the flag format used in the CTF challenge.