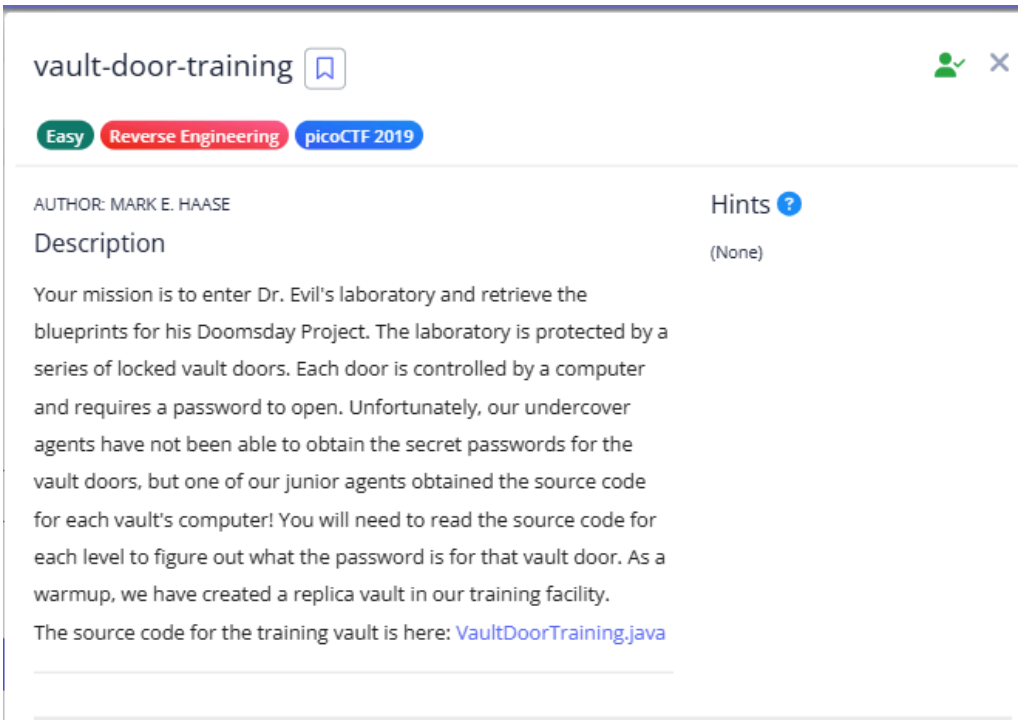
JASMINE C. OMANDAM

picoCTF - picoGym Challenges



vault-door-training

Easy Reverse Engineering picoCTF 2019

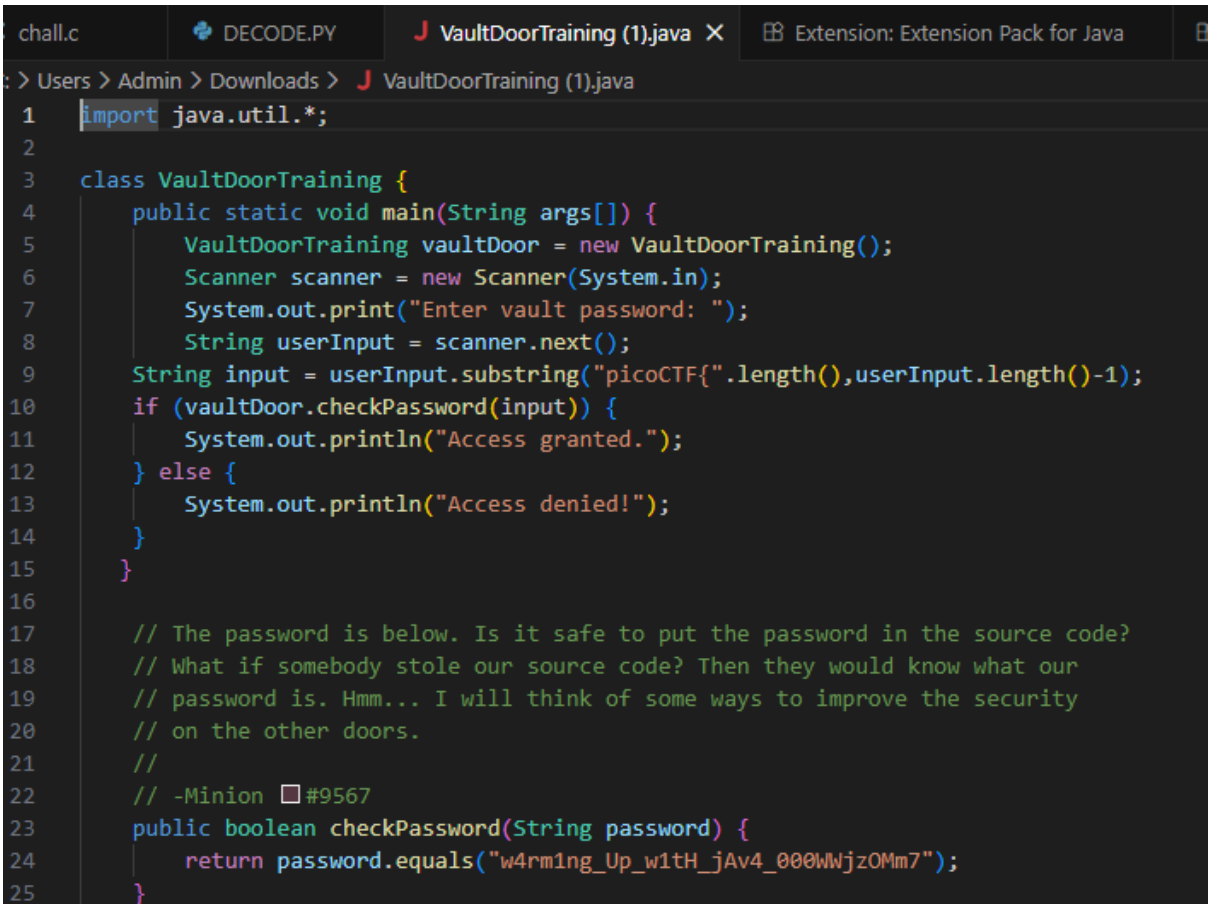
AUTHOR: MARK E. HAASE

Description

Your mission is to enter Dr. Evil's laboratory and retrieve the blueprints for his Doomsday Project. The laboratory is protected by a series of locked vault doors. Each door is controlled by a computer and requires a password to open. Unfortunately, our undercover agents have not been able to obtain the secret passwords for the vault doors, but one of our junior agents obtained the source code for each vault's computer! You will need to read the source code for each level to figure out what the password is for that vault door. As a warmup, we have created a replica vault in our training facility.
The source code for the training vault is here: VaultDoorTraining.java

Hints

(None)



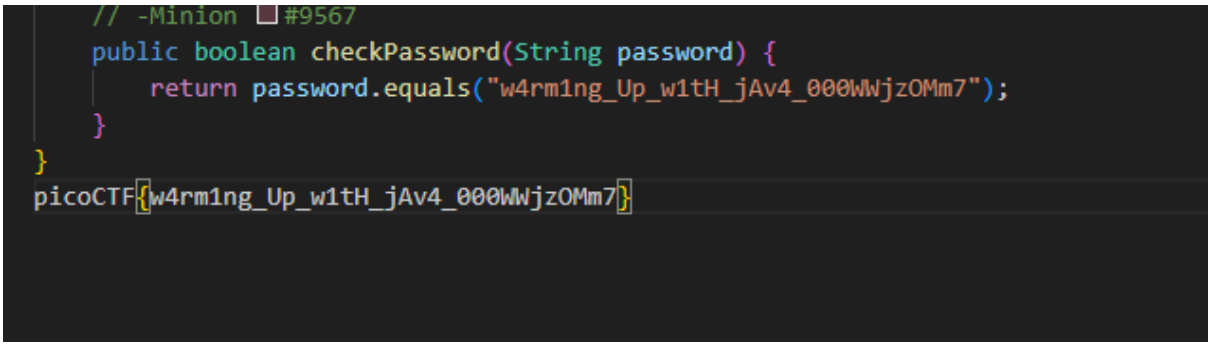
chall.c | DECODE.PY | VaultDoorTraining (1).java | Extension: Extension Pack for Java

: > Users > Admin > Downloads > VaultDoorTraining (1).java

```java
import java.util.*;

class VaultDoorTraining {
    public static void main(String args[]) {
        VaultDoorTraining vaultDoor = new VaultDoorTraining();
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter vault password: ");
        String userInput = scanner.next();
    String input = userInput.substring("picoCTF{".length(),userInput.length()-1);
    if (vaultDoor.checkPassword(input)) {
        System.out.println("Access granted.");
    } else {
        System.out.println("Access denied!");
    }
   }

    // The password is below. Is it safe to put the password in the source code?
    // What if somebody stole our source code? Then they would know what our
    // password is. Hmm... I will think of some ways to improve the security
    // on the other doors.
    //
    // -Minion #9567
    public boolean checkPassword(String password) {
        return password.equals("w4rm1ng_Up_w1tH_jAv4_000WWjzOMm7");
    }
```



Write-Up: VaultDoorTraining

When I opened the challenge file **VaultDoorTraining.java**, I immediately noticed that the program was asking for a vault password. Reading through the code, I saw how it handled user input:

String input = userInput.substring("picoCTF{".length(), userInput.length()-1);

This line revealed something important — the program expected the flag to start with **picoCTF{**, and it would strip that part off before checking the rest of the password. That meant the actual secret was hidden further down in the code.

Scrolling a bit, I found the `checkPassword` function:

return password.equal**s("w4rm1ng_Up_w1tH_jAv4_000WWjzOMm7");**

There it was the full password hardcoded inside the source. To reconstruct the flag, I simply added the prefix and suffix back:

picoCTF{w4rm1ng_Up_w1tH_jAv4_000WWjzOMm7}

Reflection

This was honestly the easiest challenge I've solved. The password was sitting right there in the source code, and all I had to do was piece it together with the proper format. It reminded me that sometimes in CTFs, the solution isn't about complex reversing or heavy tools — it's about carefully reading what's in front of you. It reinforced the idea that paying attention to details can be just as powerful as technical skill.