vault-door-4

Medium | Reverse Engineering | picoCTF 2019

AUTHOR: MARK E. HAASE

Description

This vault uses ASCII encoding for the password.

The source code for this vault is here: VaultDoor4.java

Hints

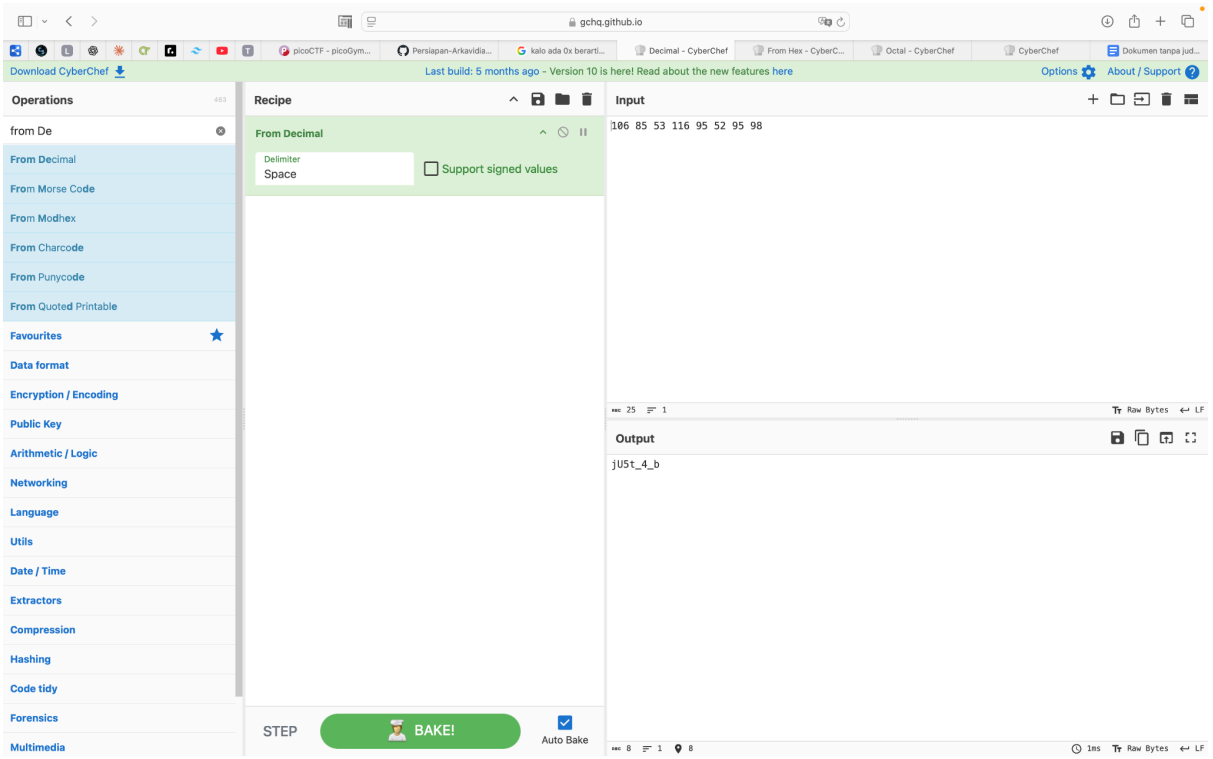
1 2

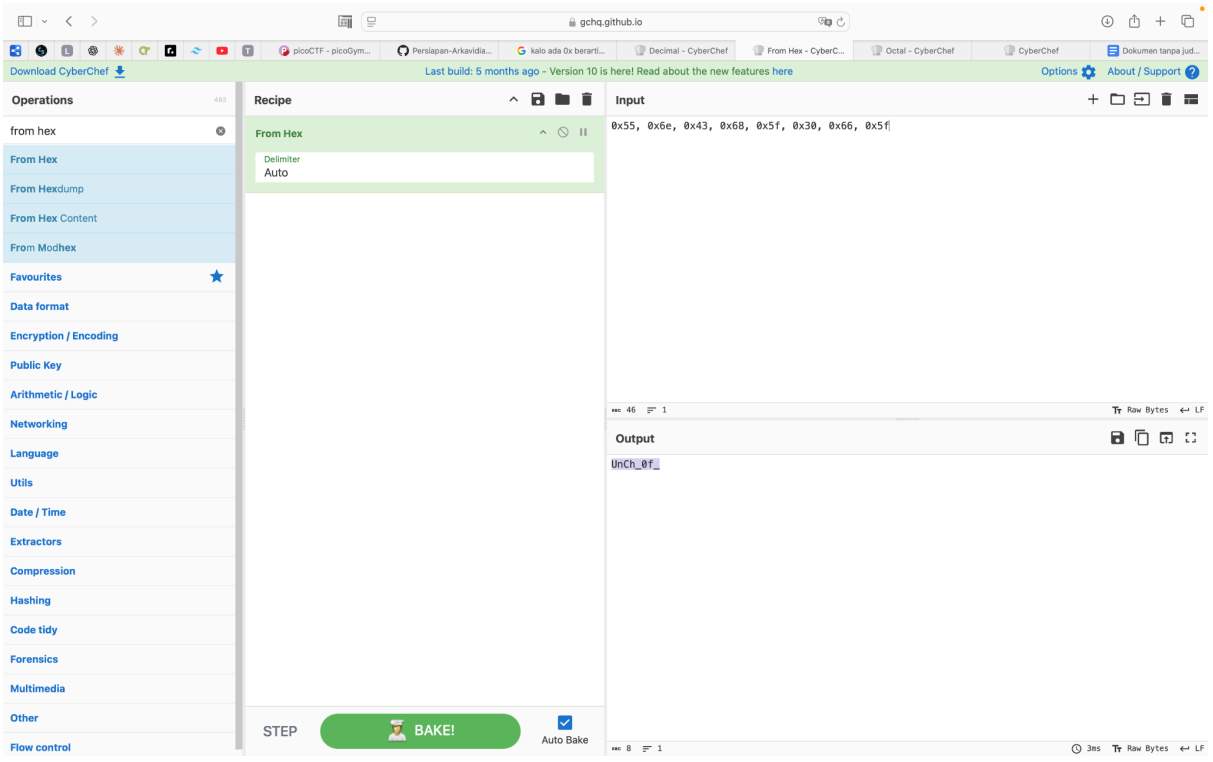
You will also need to know the difference between octal, decimal, and hexadecimal numbers.

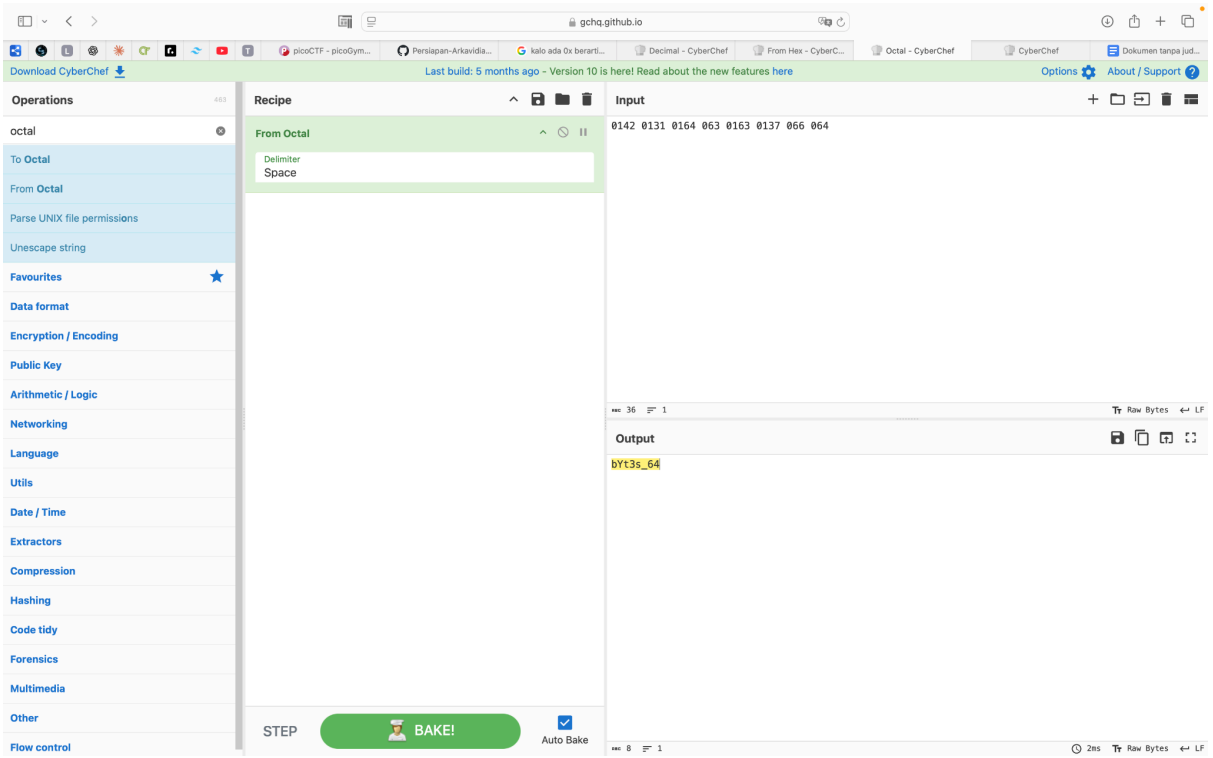
Jika file VaultDoor4.java di cat, hasilnya seperti ini. Kita bisa lihat terdapat logic validasi password yang harus sama seperti pada value myBytes dalam fungsi checkPassword().

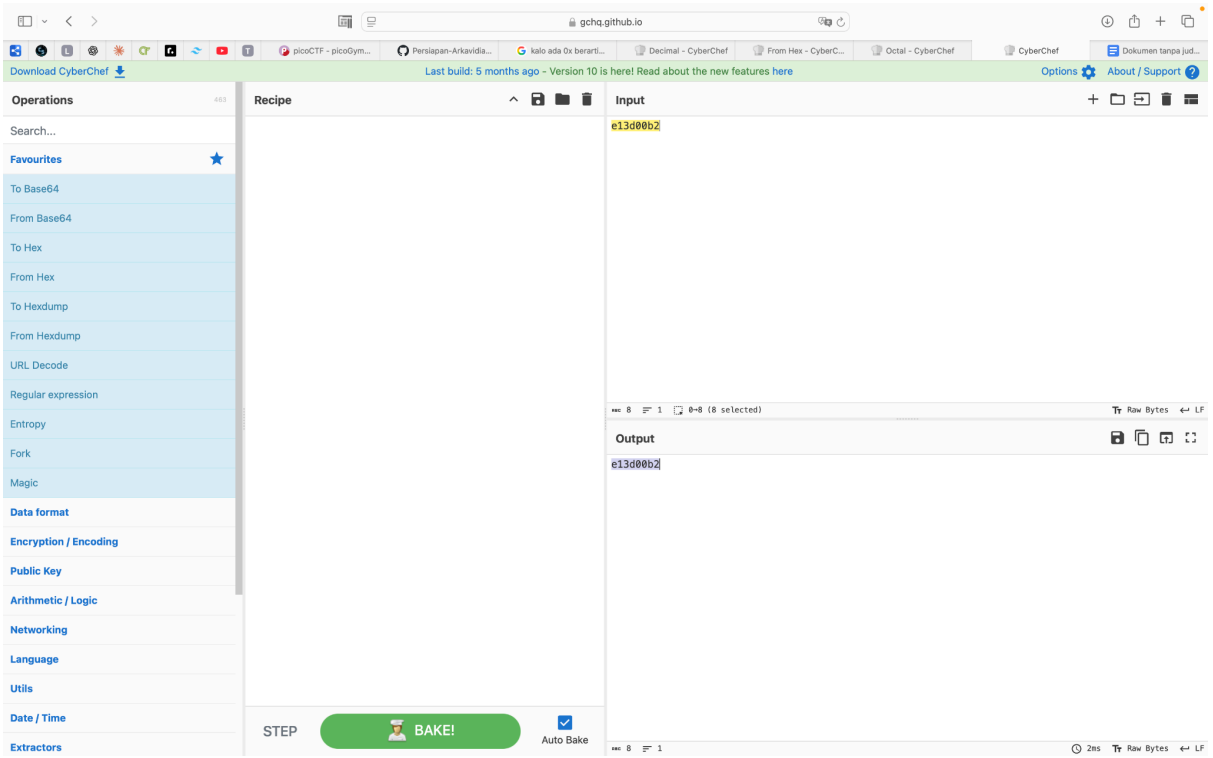
```
class VaultDoor4 {
    public static void main(String args[]) {
        VaultDoor4 vaultDoor = new VaultDoor4();
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter vault password: ");
        String userInput = scanner.next();
        String input = userInput.substring("picoCTF{".length(),userInput.length()-1);
        if (vaultDoor.checkPassword(input)) {
            System.out.println("Access granted.");
        } else {
            System.out.println("Access denied!");
        }
    }

    // I made myself dizzy converting all of these numbers into different bases,
    // so I just *know* that this vault will be impenetrable. This will make Dr.
    // Evil like me better than all of the other minions--especially Minion
    // #5620--I just know it!
    //
    //  .:::.   .:::.
    // :::::::.:::::::
    // :::::::::::::::
    // '::::::::::::::'
    //   '::::::::::'
    //     '::::::'
    //       ':'
    // -Minion #7781
    public boolean checkPassword(String password) {
        byte[] passBytes = password.getBytes();
        byte[] myBytes = {
            106 , 85  , 53  , 116 , 95  , 52  , 95  , 98  ,
            0x55, 0x6e, 0x43, 0x68, 0x5f, 0x30, 0x66, 0x5f,
            0142, 0131, 0164, 063 , 0163, 0137, 066 , 064 ,
            'e' , '1' , '3' , 'd' , '0' , '0' , 'b' , '2' ,
        };
        for (int i=0; i<32; i++) {
            if (passBytes[i] != myBytes[i]) {
                return false;
            }
        }
        return true;
    }
}
```

Decode isi dari tiap baris mybytes seperti berikut.


gchq.github.io
Download CyberChef
Last build: 5 months ago - Version 10 is here! Read about the new features here
Options
About / Support
Operations
from De
From Decimal
From Morse Code
From Modhex
From Charcode
From Punycode
From Quoted Printable
Favourites
Data format
Encryption / Encoding
Public Key
Arithmetic / Logic
Networking
Language
Utils
Date / Time
Extractors
Compression
Hashing
Code tidy
Forensics
Multimedia
Recipe
From Decimal
Delimiter
Space
Support signed values
STEP
BAKE!
Auto Bake
Input
106 85 53 116 95 52 95 98
Raw Bytes
LF
Output
jU5t_4_b
Raw Bytes
LF


gchq.github.io
Download CyberChef
Last build: 5 months ago - Version 10 is here! Read about the new features here
Options
About / Support
Operations
from hex
From Hex
From Hexdump
From Hex Content
From Modhex
Favourites
Data format
Encryption / Encoding
Public Key
Arithmetic / Logic
Networking
Language
Utils
Date / Time
Extractors
Compression
Hashing
Code tidy
Forensics
Multimedia
Other
Flow control
Recipe
From Hex
Delimiter
Auto
STEP
BAKE!
Auto Bake
Input
0x55, 0x6e, 0x43, 0x68, 0x5f, 0x30, 0x66, 0x5f
Raw Bytes
LF
Output
UnCh_0f_
Raw Bytes
LF





Flag tinggal disambung saja.