

[Active] leet code/ctf

notes/explanations

- [1-50](#)
 - [1. Two Sum](#)
- [PICOCTF WPA-ing Out \(Rockyou word list + aircrack-ng\)](#)
- [Python Wrangling](#)
- [New Page](#)
- [gobuster Quirks](#)

1-50

1-50

1. Two Sum

Given an array of integers `nums` and an integer `target`, return *indices of the two numbers such that they add up to* `target`.

You may assume that each input would have **exactly one solution**, and you may not use the *same* element twice.

You can return the answer in any order.

Example 1:

Input: `nums = [2,7,11,15]`, `target = 9`

Output: `[0,1]`

Explanation: Because `nums[0] + nums[1] == 9`, we return `[0, 1]`.

Example 2:

Input: `nums = [3,2,4]`, `target = 6`

Output: `[1,2]`

Example 3:

Input: `nums = [3,3]`, `target = 6`

Output: `[0,1]`

Constraints:

- `2 <= nums.length <= 104`
- `-109 <= nums[i] <= 109`
- `-109 <= target <= 109`
- **Only one valid answer exists.**

here was my English explanation of how i solved it

its for each number in the array we index it then we start with the first value (firstvalue_x) then subtract that from the target value(target_val), if the result (secondvalue_y) is in the index and isnt the index number(firstvalue_x) we subtracted from the target value(target_val) that is the answer(answer) otherwise continue with this loop by increasing the (firstvalue_x). we dont have to look at values we have tested already

```

public class Solution {
    public int[] TwoSum(int[] nums, int target) {
        Dictionary<int, int> dict = new Dictionary<int, int>();
        // We create an index (dictionary) to store each number and its position in the array.
        for (int n = 0; n < nums.Length; n++) {
            // We start a loop to go through each number in the array. The variable 'i' is the position of
            // the current number (firstvalue_x).
            int complement = target - nums[n];
            // We subtract the current number (firstvalue_x) from the target value (target_val), the
            // result is the complement (secondvalue_y).
            if (dict.ContainsKey(complement) && dict[complement] != n) {
                // We check if the complement (secondvalue_y) is in the index (dictionary) and its position is
                // not the same as the current position (firstvalue_x).
                return new int[] { dict[complement], n };
            }
            // If the above condition is true, we've found the answer (answer). We return the positions
            // of the two numbers that add up to the target.
            dict[nums[n]] = n;
            // If the complement is not in the index, we add the current number and its position to the
            // index (dictionary) and continue with the loop by increasing the current position
            // (firstvalue_x).
        }
        throw new Exception("There's a problem with your code or the input array is
        bad");
        // If we've gone through all the numbers and haven't found two numbers that add up to the
        // target, we throw an exception.
        // we need this to allow compiler to compile because the compiler requires that all code paths
        // in a function must return a value if the function is declared to return a value
        // we *have* to have this or it just fails at compiling
    }
}

```

cases for question

```

case 1 [2,7,11,15] target = 9
case 2 [3,2,4] target = 6
case 3 [3,3] target = 6

```

The code works for these test questions

Follow-up: Can you come up with an algorithm that is less than $O(n^2)$ time complexity?
i asked chat gpt coz idk if it is off the top of my head coz i cant think lol

"Yes, the provided solution fits the follow-up question. The time complexity of this solution is $O(n)$, which is less than $O(n^2)$.

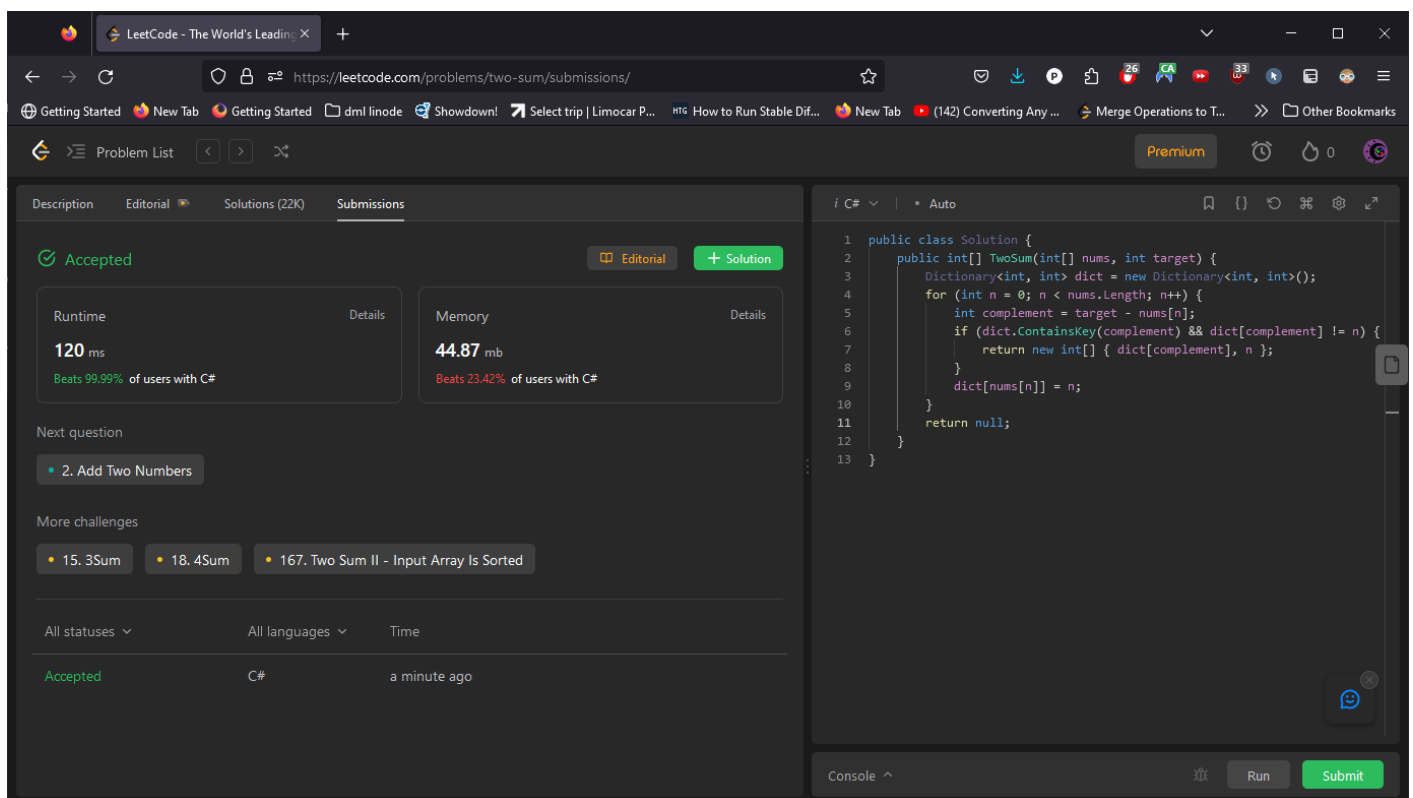
Here's why:

The solution iterates over the `nums` array only once. For each number, it performs operations that take constant time: calculating the complement, checking if the complement is in the dictionary, and adding a number to the dictionary. Therefore, the time complexity is proportional to the size of the `nums` array, which is $O(n)$."

what does leetcode say to my answer?

well it likes it :D i guess its about as fast as it can be, especially if you remove the comments

i did some minor optimizations
and heres what i got



The screenshot shows a LeetCode submission for the "Two Sum" problem. The submission is accepted, with a runtime of 120 ms and memory usage of 44.87 mb. The code is written in C# and uses a Dictionary to store the complement of each number. The code is as follows:

```
1 public class Solution {
2     public int[] TwoSum(int[] nums, int target) {
3         Dictionary<int, int> dict = new Dictionary<int, int>();
4         for (int n = 0; n < nums.Length; n++) {
5             int complement = target - nums[n];
6             if (dict.ContainsKey(complement) && dict[complement] != n) {
7                 return new int[] { dict[complement], n };
8             }
9             dict[nums[n]] = n;
10        }
11        return null;
12    }
13 }
```

Intuition

The problem asks for finding two numbers in an array that add up to a specific target. The "easy" but inefficient way would be to check all possible pairs of numbers, which would take $O(n^2)$ time. Instead, we can use the property of addition: for any number z , if $x + y$ equals z , then y must be equal to z minus x . So, for each number in the array, we can check its complement to the target $[z]$ (i.e., $[z] - x$) and then check if this complement is in the array. If it is, and it's not the same index we're currently looking at, we've found the two numbers we need. To make this check fast, we can use a hash table (or a Dictionary in C#) to store the numbers from the array and their indices.

Approach

My English pseudo code/thought process is

"for each number in the array, we index it.

then we start with the first indexed value (firstvalue_x)

then subtract that from the target value (target_val_z),

if the result (secondvalue_y) is in the index and isn't the index number(firstvalue_x) we subtracted from the target value(target_val_z) that is the answer ($[x, y]$) otherwise continue with this loop by increasing the (firstvalue_x).

we do not have to look at values we have tested already because that would be redundant

The approach is to iterate through the array and for each number, calculate its complement to the target and check if this complement is already in the Dictionary. If it is, and it's not the same element (i.e., it has a different index), we've found two numbers that add up to the target and we return their indices. If the complement is not in the Dictionary, we add the current number and its index to the Dictionary. This way, we don't have to revisit numbers we've already processed, which would be redundant.

Complexity

- Time complexity:

The time complexity is $O(n)$, where n is the length of the array. We're iterating through the array once, and for each number, we're performing constant time operations (calculating the complement, checking if it's in the dictionary, and adding a number to the dictionary).

- Space complexity:

The space complexity is also $O(n)$, where n is the length of the array. In the worst case, we might end up adding every number in the array to the dictionary.

Extra

this is my first post.

It took me a bit of digging to learn what to use in c# for this specifically coz ive only ever used it for

unity @~@ and i usually use python so it was a bit of a challenge to get the code to not give an error XD

and i found out that it wont compile if we dont have an exception handler if theres an error in a hypothetical input....

sooo since we are using c# ive found that using

```
throw new Exception("exception");
```

is slower on avg

so

```
return null;
```

is used

also since im new to leetcode.com ive foundout that the compieler speeds can be subject to rng, also submissions have processing priority ig.

my codes runtime is 120 and worst is 145

its avg is in the low 130s

Code

```
public class Solution {  
    public int[] TwoSum(int[] nums, int target) {  
        Dictionary<int, int> dict = new Dictionary<int, int>();  
        for (int n = 0; n < nums.Length; n++) {  
            int complement = target - nums[n];  
            if (dict.ContainsKey(complement) && dict[complement] != n) {  
                return new int[] { dict[complement], n };  
            }  
            dict[nums[n]] = n;  
        }  
        return null;  
    }  
}
```

PICOCTF WPA-ing Out (Rockyou word list + aircrack-ng)

| 200 points

Tags: picoGym Exclusive Forensics

Author: MistressVampy

Description

I thought that my password was super-secret, but it turns out that passwords passed over the AIR can be CRACKED, especially if I used the same wireless network password as one in the rockyou.txt credential dump. Use this 'pcap file' and the rockyou wordlist. The flag should be entered in the picoCTF{XXXXXX} format.

uhh
bro this took time, not because it's hard but getting the rock you wordlist was not working on the kali linux wsl install i had. so i just manually downloaded it because BROO i cba if it doesn't wanna work

Pico CTF pcap for this challenge

https://artifacts.picoctf.net/c/41/wpa-ing_out.pcap

O.o step 1

if you Wireshark the whole thing, it's impossible to run through

```
sudo apt install aircrack-ng
```

this is what happens when you try to air crack without the wordlist being accessible

```
└─(naruzkurai@YunonZKurai)-[/mnt/a/ctf/WPA-ing Out]
└─$ aircrack-ng -w /usr/share/wordlists/rockyou.txt wpa-ing_out.pcap
ERROR: Opening dictionary /usr/share/wordlists/rockyou.txt failed (No such file or
directory)
```


ERROR: Opening dictionary /usr/share/wordlists/rockyou.txt failed (No such file or directory)

Reading packets, please wait...

Opening wpa-ing_out.pcap

Resetting EAPOL Handshake decoder state.

Resetting EAPOL Handshake decoder state.

Read 23523 packets.

#	BSSID	ESSID	Encryption
1	00:5F:67:4F:6A:1A	Gone_Surfing	WPA (1 handshake)

Choosing first network as target.

Reading packets, please wait...

Opening wpa-ing_out.pcap

Resetting EAPOL Handshake decoder state.

Resetting EAPOL Handshake decoder state.

Read 23523 packets.

1 potential targets

Please specify a dictionary (option -w).

uhhh ok lets try to install the word-list @~@ (I actually didn't try anything else)

Manual download of wordlist if kali is being stooooopid

```
wget https://github.com/danielmiessler/SecLists/raw/master/Passwords/Leaked-Databases/rockyou.txt.tar.gz
tar -xzf rockyou.txt.tar.gz
mv rockyou.txt /usr/share/wordlists/
```

Answer

```
└─(naruzkurai@YunonZKurai)-[/mnt/a/ctf/WPA-ing Out]
└─$ aircrack-ng -w /usr/share/wordlists/rockyou.txt wpa-ing_out.pcap
```

Reading packets, please wait...
Opening wpa-ing_out.pcap
Resetting EAPOL Handshake decoder state.
Resetting EAPOL Handshake decoder state.
Read 23523 packets.

#	BSSID	ESSID	Encryption
1	00:5F:67:4F:6A:1A	Gone_Surfing	WPA (1 handshake)

Choosing first network as target.

Reading packets, please wait...
Opening wpa-ing_out.pcap
Resetting EAPOL Handshake decoder state.
Resetting EAPOL Handshake decoder state.
Read 23523 packets.

1 potential targets

Aircrack-ng 1.7

[00:00:00] 1173/10303727 keys tested (20737.79 k/s)

Time left: 8 minutes, 16 seconds 0.01%

KEY FOUND! [mickeymouse]

Master Key : 61 64 B9 5E FC 6F 41 70 70 81 F6 40 80 9F AF B1
4A 9E C5 C4 E1 67 B8 AB 58 E3 E8 8E E6 66 EB 11

Transient Key : 26 85 7B AC DD 2C 44 E6 06 18 03 B0 0F F2 75 A2
32 63 F7 35 74 2D 18 10 1C 25 F9 14 BC 41 DA 58
52 48 86 B0 D6 14 89 F6 77 00 67 E0 AD 10 1B 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

EAPOL HMAC : 65 2F 6C 0E 75 F0 49 27 6A AA 6A 06 A7 24 B9 A9

answer (if you put things together)

picoCTF{mickeymouse}

Python Wrangling

files you will need

ende.py

```
import sys
import base64
from cryptography.fernet import Fernet

usage_msg = "Usage: "+ sys.argv[0] +" (-e/-d) [file]"
help_msg = usage_msg + "\n" +\
    "Examples:\n" +\
    "  To decrypt a file named 'pole.txt', do: " +\
    "'$ python "+ sys.argv[0] +" -d pole.txt'\n"

if len(sys.argv) < 2 or len(sys.argv) > 4:
    print(usage_msg)
    sys.exit(1)

if sys.argv[1] == "-e":
    if len(sys.argv) < 4:
        sim_sala_bim = input("Please enter the password:")
    else:
        sim_sala_bim = sys.argv[3]

    ssb_b64 = base64.b64encode(sim_sala_bim.encode())
    c = Fernet(ssb_b64)

    with open(sys.argv[2], "rb") as f:
```

```
        data = f.read()
        data_c = c.encrypt(data)
        sys.stdout.write(data_c.decode())

elif sys.argv[1] == "-d":
    if len(sys.argv) < 4:
        sim_sala_bim = input("Please enter the password:")
    else:
        sim_sala_bim = sys.argv[3]

    ssb_b64 = base64.b64encode(sim_sala_bim.encode())
    c = Fernet(ssb_b64)

    with open(sys.argv[2], "r") as f:
        data = f.read()
        data_c = c.decrypt(data.encode())
        sys.stdout.buffer.write(data_c)

elif sys.argv[1] == "-h" or sys.argv[1] == "--help":
    print(help_msg)
    sys.exit(1)

else:
    print("Unrecognized first argument: "+ sys.argv[1])
    print("Please use '-e', '-d', or '-h'.")
```

pw.txt

68f88f9368f88f9368f88f9368f88f93

flag.txt.en

gAAAAABgUAIv8D5MJdzgLLTkkMlbU84ARVwfX4brMt2rJQCjKpLItytfWVZe1L2dp4K8VrKgRU3axStKJEAqcM0iDaxiYE54Boh8UfAAo1RNifKn\DrFz0gLaznVSFVj2xAUa4V35180

solution

```
PS A:\ctf\Python Wrangling> python .\ende.py -d .\flag.txt.en
Please enter the password:68f88f9368f88f9368f88f9368f88f9368f88f93
```

answer (this is the output)

picoCTF{4p0110_1n_7h3_h0us3_68f88f93}

New Page

script to solve it

```
encoded_str = "\x00\x00\x00\x00\x00\x00\x00\x00"
decoded_str = ""

for char in encoded_str:
    ascii_val = ord(char)
    first_char = chr(ascii_val >> 8)
    second_char = chr(ascii_val % 256)
    decoded_str += first_char + second_char

print(decoded_str)
```

output with answer

```
PS A:\ctf\Transformation> & C:/Python312/python.exe a:/ctf/Transformation/decode.py
picoCTF{16_bits_inst34d_of_8_e703b486}
```

gobuster Quirks

- add this encase it just times out when u use vhost

```
-append-domain
```

- needs to be installed from git repo
- <https://github.com/danielmiessler/SecLists> is usefull