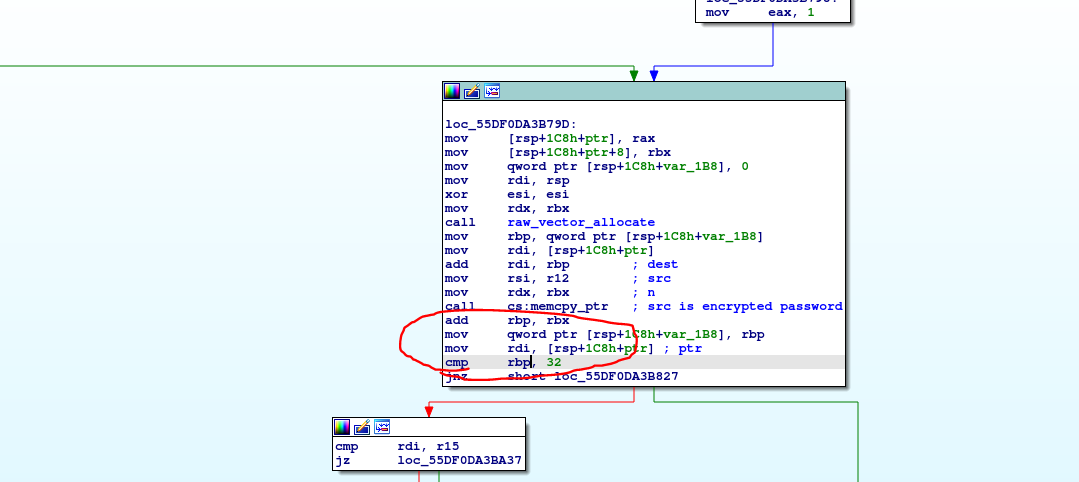


Written in rust so many mangled names, found the one that takes input

After debugging and reversing found the input is cloned into another string and encrypted using library

<https://docs.rs/salsa20/0.9.0/salsa20/>



After encryption it is compared whether it is 32 bytes long or not

After debugging found that encryption key and nounce is

ef39f4f20e76e33bd25f4db338e81b10

d4c270a3

Turns out for this particular algorithm the same key and nounce is encryption and decryption. And I discovered that the encrypted input is checked with

[ 0x5, 0x5, 0x5F ,0x0B1,0x0A3, 0x29 ,

    0x0A8,0x0D5, 0x58 ,0x0D9,0x0F5, 0x56 ,0x0A6,0x0CB, 0x31,

    0x0F3, 0x24, 0x43 , 0x2A , 0x31 ,0x0C9, 0x9D,0x0EC, 0x72 ,0x0E3,

    0x3E ,0x0B6, 0x6F , 0x62 ,0x0AD, 0x1B,0x0F9];

So I just encrypted it again to get the password

The code to get the password will be:

use salsa20::{Salsa20, Key, Nonce};

use salsa20::cipher::{NewCipher, StreamCipher, StreamCipherSeek};

fn main() {

*let* mut data =[ 0x5, 0x5, 0x5F ,0x0B1,0x0A3, 0x29 ,

    0x0A8,0x0D5, 0x58 ,0x0D9,0x0F5, 0x56 ,0x0A6,0x0CB, 0x31,

    0x0F3, 0x24, 0x43 , 0x2A , 0x31 ,0x0C9, 0x9D,0x0EC, 0x72 ,0x0E3,

    0x3E ,0x0B6, 0x6F , 0x62 ,0x0AD, 0x1B,0x0F9];

*let* key = Key::from\_slice(b"ef39f4f20e76e33bd25f4db338e81b10");

*let* nonce = Nonce::from\_slice(b"d4c270a3");

*let* mut password = String::new();

    // create cipher instance

*let* mut cipher = Salsa20::new(&key, &nonce);

    // apply keystream (encrypt)

    cipher.seek(0);

    cipher.apply\_keystream(&mut data);

    for i in data.iter(){

        password.push(\*i as *char*);

    }

    println!("Password is {}",password);

}

Be sure to include the salsa crate in you dependency as

***[package]***

***name = "ctf"***

***version = "0.1.0"***

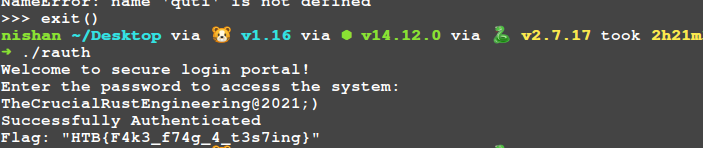
***authors = ["Nishan8583 <nishanmaharjan17@gmail.com>"]***

***edition = "2018"***

***# See more keys and their definitions at https://doc.rust-lang.org/cargo/reference/manifest.html***

***[dependencies]***

***salsa20 = "0.9.0"***



So it worked on our local files provided for the challenge, but the flag will not work cause it’s not the correct flag,

Connect to the instance using netcat, and got the flag after inserting the password:

