

# Access Breaker – Reverse Engineering Writeup

Challenge Overview: An ELF binary called rev\_easy prompts for a flag and validates it using obfuscated arithmetic and XOR logic.

Step 1 – Recon: Using strings revealed prompts, success messages, and obfuscated byte sequences.

Step 2 – Disassembly: Using objdump, main logic was located near fgets and strlen calls.

Step 3 – Length Check: The program checks that input length is exactly 18 bytes.

Step 4 – Transformation Logic: Each byte is transformed as:  $\text{out} = ((\text{input} \wedge \text{edx}) + \text{esi}) \wedge \text{edi}$ . Registers update each iteration.

Step 5 – Extract Secret: Hidden byte array was extracted from the .rodata section.

Step 6 – Reverse Formula:  $\text{input} = ((\text{out} \wedge \text{edi}) - \text{esi}) \wedge \text{edx}$ .

Step 7 – Python Solver: A script reversed the operations to recover the original input.

Final Flag: SECE{rev4fun\_2025}