A cartoon of a person in a jacket

AI-generated content may be incorrect.

Based one hint 1, the flag has three parts and based on number of point needed to get hint 2, we locate all code paths that XOR with 0x36.

sh -c "objdump -d reez\_revenge | grep -n '83 f0 36'"

We get

619: 1805: 83 f0 36 xor $0x36,%eax

1640: 257f: 83 f0 36 xor $0x36,%eax

2686: 3364: 83 f0 36 xor $0x36,%eax

Map each address to its function:  
gdb -q ./reez\_revenge -ex "info symbol 0x1805" -ex "info symbol 0x257f" -ex "info symbol 0x3364" -ex q

Output

aABbcc + 110 in section .text

AaBBcC + 122 in section .text

AAbbCC + 122 in section .text

Then we disassemble these chunk

gdb -q ./reez\_revenge -ex "disassemble aABbcc" -ex q

gdb -q ./reez\_revenge -ex "disassemble AaBBcC" -ex q

gdb -q ./reez\_revenge -ex "disassemble AAbbCC" -ex q

Extract the immediates from the movabs/movl/movb instructions and form the “chunk” bytes:

movabs loads a 64-bit immediate (displayed big-endian), but writes to memory in little-endian. Convert the immediate to little-endian bytes.

movl gives a 32-bit immediate (store as 4 bytes, little-endian).

movb gives a single byte. Running flip.py we got:

chunk hex: 50435869425e57586945595a405f5851695f424b

decoded: fun\_than\_solving\_it}

chunk hex: 405143554f465e53444d58575b5f58516900026950

decoded: vgucypher{naming\_64\_f

chunk hex: 435855425f595845695f456941574f695b59445369

decoded: unctions\_is\_way\_more\_  
Rearrange these fragment, we got the flag vgucypher{naming\_64\_functions\_is\_way\_more\_fun\_than\_solving\_it}