**Walkthrough for ECC CTF Challenge**

**Introduction**

This challenge revolves around elliptic curve cryptography (ECC) and aims to confuse solvers using nonlinear transformations, entangled masks, and complex point operations. The goal is to locate the secret condition and derive the hidden flag.

**Step 1: Understanding the Code**

The program implements a basic elliptic curve over a finite field with the following main components:

- Point operations: `point\_add`, `point\_double`

- Scalar multiplication: `scalar\_mult`

- Nonlinear entanglement: XOR masks that evolve during scalar multiplication

The crucial part is the secret check in `scalar\_mult`:

if (mpz\_tstbit(N.x, 8) && mpz\_tstbit(N.y, 12)) {

printf("Secret found!\n");

}

The condition checks if the 8th bit of `N.x` and the 12th bit of `N.y` are both set.

**Step 2: Identifying the Base Point**

The program uses a well-known elliptic curve point `G` defined by:

G.x = 55066263022277343669578718895168534326250603453777594175500187360389116729240

G.y = 32670510020758816978083085130507043184471273380659243275938904335757337482424

These coordinates correspond to the base point of the secp256k1 curve.

**Step 3: Tracking the Secret Condition**

- The scalar multiplication uses a chaotic evolving mask:

mpz\_xor(N.x, N.x, mask);

mpz\_add\_ui(mask, mask, 1337);

This ensures that the final `N.x` and `N.y` values are entangled with the mask's evolution, complicating direct computation.

- The flag condition appears in:

if (mpz\_tstbit(N.x, 8) && mpz\_tstbit(N.y, 12))

Brute-forcing the scalar `k` or reverse-engineering the bit conditions is the only way forward.

**Step 4: Extracting the Flag**

If the condition is met, the program outputs:

Secret found!

However, there’s no direct printout of the flag. To reveal the flag, you can insert this line right after the condition check:

if (mpz\_tstbit(N.x, 8) && mpz\_tstbit(N.y, 12)) {

printf("%s\n", FLAG);

}

**The output will then display:**

Secret found!  
G8KEY{A\_R@NK\_HUNTER\_R3V3EL3D}

**Conclusion**

The challenge's difficulty stems from the nonlinear bitwise operations and dynamic masks, making traditional ECC attacks difficult. Skilled solvers must carefully analyze how point operations and masks interact to trigger the hidden condition.