Can you evolve it?

```
//gcc main.c -o main -lssl -lcrypto

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <openssl/aes.h>

void initialize() {
    setvbuf(stdin, NULL, _IONBF, 0);
    setvbuf(stdout, NULL, _IONBF, 0);
}

void string_to_hex(const unsigned char *input, char *output)
    while (*input) {
        sprintf(output, "%02x", *input);
        output += 2;
        input++;
    }
}

void aes_ecb_encrypt(const unsigned char *key, const unsigned
    AES_KEY encryptKey;
    AES_set_encrypt_key(key, 128, &encryptKey);

    int len = strlen((const char *)plaintext);
    for (int i = 0; i < len; i += AES_BLOCK_SIZE) {
        AES_ecb_encrypt(plaintext + i, ciphertext + i, &encry
    }
}

void aes_ecb_decrypt(const unsigned char *key, const unsigned
    AES_KEY decryptKey;
    AES_set_decrypt_key(key, 128, &decryptKey);
    int len = strlen((const char *)ciphertext);
    for (int i = 0; i < len; i += AES_BLOCK_SIZE) {
        AES_ecb_encrypt(ciphertext + i, plaintext + i, &decry
    }
}

void getFlag() {

    FILE* file = fopen("./flag.txt", "rb");

    unsigned char buffer[40] = {0};
    size_t bytesRead;

    if((bytesRead = fread(buffer, 1, sizeof(buffer), file)) >
        printf("FLAG : %s", buffer);
    } else {
    printf("File read error!!\n");
    }
    printf("\n");
}

int main() {

    const unsigned char* key = "4RG0S_Pass_K3Y!!";
    unsigned char encrypt_pwd[128] = {0};
    unsigned char* admin_pwd = "50c14d3ad61c89d391eb6f79e559e
    unsigned char decryptedtext[128] = {0};
    char* id_list[4] = {"argos", "guest", "chacha", "admin"};
    char id[20];
    unsigned char password[30] = {0};
    unsigned char hex_pwd[60] = {0};
    unsigned char hex_test[60] = {0};
    unsigned char test[128] = {0};

    int i, j;
    int available_id = -1;
    initialize();

    digivice();

    printf("Login Digivice!!\n");

    printf("[ >] ID : ");
    scanf("%s", id);

    printf("[ >] PW : ");
    scanf("%s", password);

    for(i=0; i < 4; i++) {

        if (strlen(id_list[i]) != strlen(id)) {
            continue;
        }

        if ( !strcmp(id_list[i], id) ) {
            available_id = i;
        }

    }
    if (available_id != -1) {
        if (available_id == 3) {
            aes_ecb_encrypt(key, password, encrypt_pwd);

            string_to_hex(encrypt_pwd, hex_pwd);
            string_to_hex(test, hex_test);
            for( i=0; i < strlen(admin_pwd); i++ ) {
                if (hex_pwd[i] != admin_pwd[i]) {

                    evolution();
                    fail();
                }
            }
            printf("Login Success!! Please wait.");
            fflush(stdout);
            usleep(1000000);
            printf("\rLogin Success!! Please wait..");
            fflush(stdout);
            usleep(1000000);
            printf("\rLogin Success!! Please wait...");
            fflush(stdout);
            usleep(1000000);
            evolution();
            flag();
            getFlag();
        }
        else{
            evolution();
            fail();
            printf("Hello, %s\n", id_list[available_id]);
        }
    } else {
        printf("ID not in list. Please try again..\n");
    }

    return 0;
}
```

소스 코드를 읽어보면 로그인에 성공하면 flag를 서버에서 읽어와 출력을 해준다.

```
void getFlag() {

    FILE* file = fopen("./flag.txt", "rb");

    unsigned char buffer[40] = {0};
    size_t bytesRead;

    if((bytesRead = fread(buffer, 1, sizeof(buffer), file)) >
        printf("FLAG : %s", buffer);
    } else {
    printf("File read error!!\n");
    }
    printf("\n");
}
```

로그인을 위해서는 id는 admin, password는 암호화된 hex값을 복호화하여 문자열을 찾아야 한다.

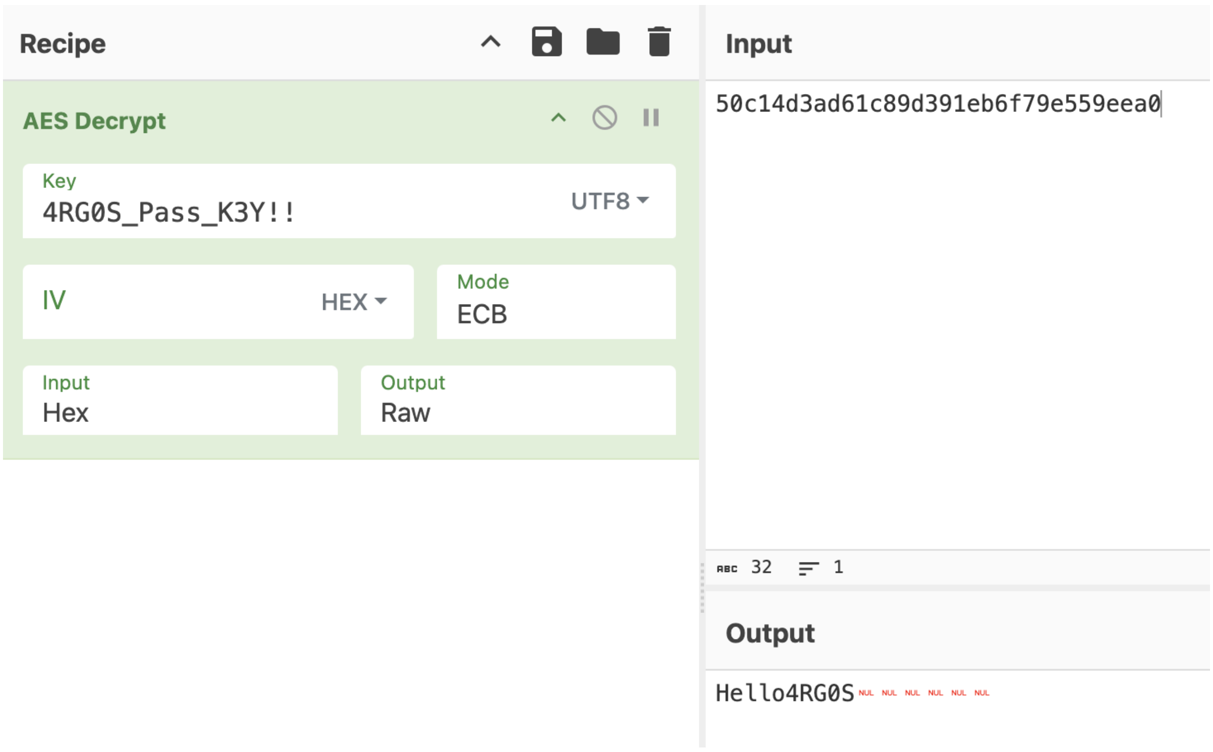
```
if (available_id != -1) {
        if (available_id == 3) {
            aes_ecb_encrypt(key, password, encrypt_pwd);

            string_to_hex(encrypt_pwd, hex_pwd);
            string_to_hex(test, hex_test);
            for( i=0; i < strlen(admin_pwd); i++ ) {
                if (hex_pwd[i] != admin_pwd[i]) {

                    evolution();
                    fail();
                }
            }
            printf("Login Success!! Please wait.");
            fflush(stdout);
            usleep(1000000);
            printf("\rLogin Success!! Please wait..");
            fflush(stdout);
            usleep(1000000);
            printf("\rLogin Success!! Please wait...");
            fflush(stdout);
            usleep(1000000);
            evolution();
            flag();
            getFlag();
        }
```

사용된 함수를 확인하면 aes-ecb를 사용하였고

```
const unsigned char* key = "4RG0S_Pass_K3Y!!";
```

위의 key를 사용하여 암/복호화를 진행한다

Cyberchef를 사용하면



Key: 4RG0S_Pass_K3Y!! (UTF8)
IV: (HEX)
Mode: ECB
Input: Hex
Output: Raw

Input:

```
50c14d3ad61c89d391eb6f79e559eea0
```

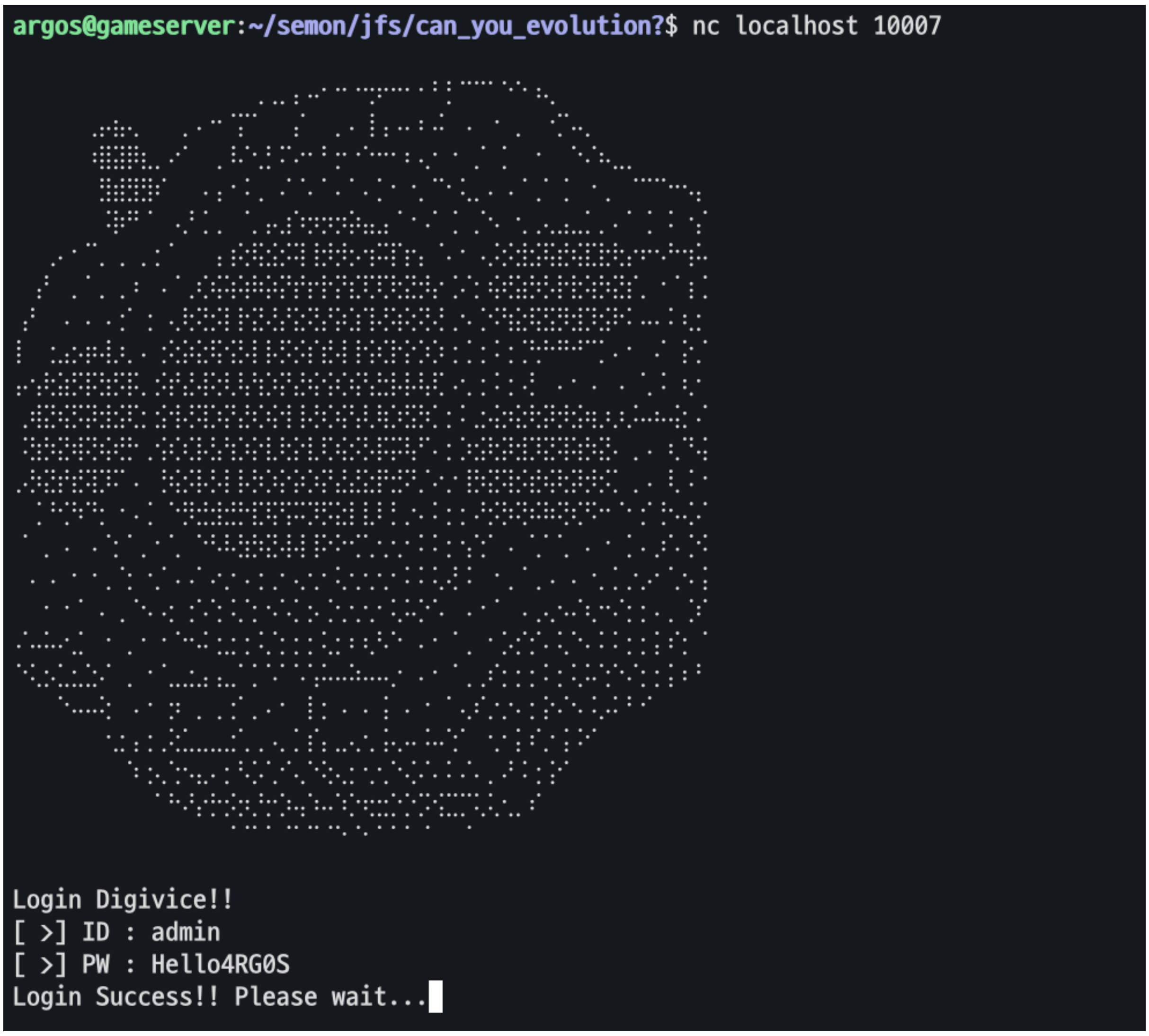
ABC 32 ☰ 1

Output:

```
Hello4RG0S NUL NUL NUL NUL NUL NUL
```

위와 같이 Hello4RG0S(password)를 구할 수 있습니다.

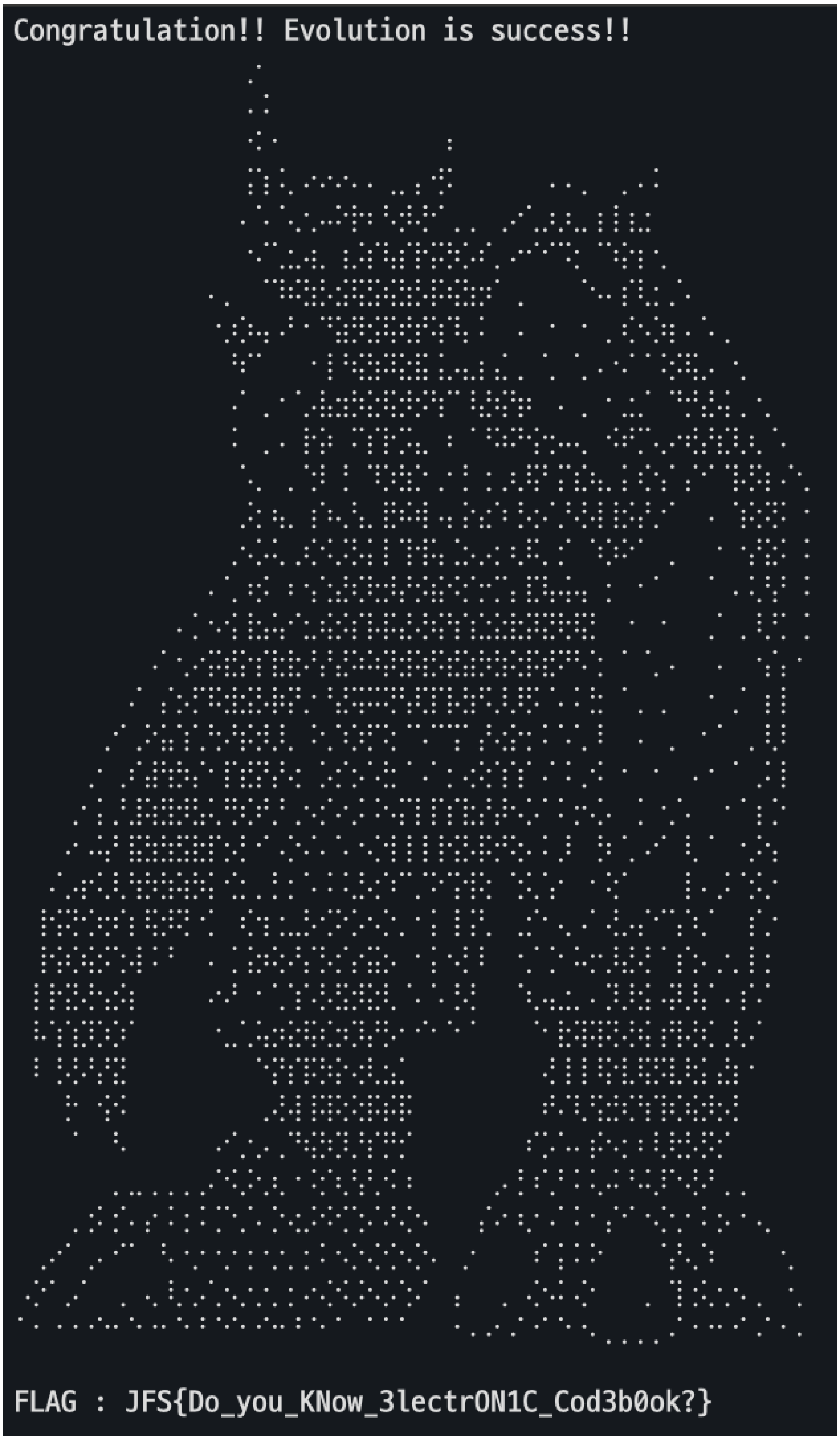
이를 이용하여 로그인을 시도하면



```
argos@gameserver:~/semon/jfs/can_you_evolution?$ nc localhost 10007
```

```
Login Digivice!!
[ >] ID : admin
[ >] PW : Hello4RG0S
Login Success!! Please wait...
```

아래와 같이 성공 문자열과 함께 진화에 성공하고 flag를 얻을 수 있다.



```
Congratulation!! Evolution is success!!
```

```
FLAG : JFS{Do_you_KNow_3lectrON1C_Cod3b0ok?}
```

`FLAG : JFS{Do_you_KNow_3lectrON1C_Cod3b0ok?}`