```
/*
 * Mark El-Khoury
 * April 2014
 * Available at http://mu.gl/key.c
 *
 *
 *
 * MAKEFILE (tab before gcc):
 *
 * INC=/usr/local/ssl/include/
 * LIB=/usr/local/ssl/lib/
 * all:
 *         gcc -I$(INC) -L$(LIB) -o enc key.c -lcrypto -ldl
 *
 *
 * Sample input:
 * Plaintext (total 21 characters): This is a top secret.
 * Ciphertext (in hex format): 8d20e5056a8d24d0462ce74e4904c1b513e10d1df4a2ef2ad4540fae1ca0aaf9
 *
 */

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

#define MAX_PLAINTEXT_LENGTH 21
#define MAX_WORD_LEN 16
#define MAX_BUF_LEN 128
#define IV_LEN 16

static unsigned char* hexdump(const unsigned char *s, int l) {
    l /= 4;
    unsigned char * strhex = (unsigned char *)malloc(2*l*sizeof(unsigned char));
    memset(strhex, '\0', 2*l);
    int n = 0;
    int printed = 0;
    for( ; n < l; ++n) {
        printed += sprintf(strhex + printed, "%02x", s[n]);
    }
    //sprintf(strhex + printed, "\n");
    //printf("%d ", printed);

    return (unsigned char *)strhex;
}

int main(int argc, char *arv[]) {

    /*unsigned char plaintext[MAX_PLAINTEXT_LENGTH];
    unsigned char ciphertext[MAX_BUF_LEN];*/
    unsigned char temp_cipher[MAX_BUF_LEN];
    unsigned char IV[] = {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};


    FILE *fp = fopen("words.txt", "r");
    if(fp == NULL) {
        perror("Failed to open the dictionnary \"words.txt\".");
        return EXIT_FAILURE;
    }
    /*

    memset(ciphertext, '\0', MAX_BUF_LEN);
    memset(plaintext, '\0', MAX_PLAINTEXT_LENGTH);

    printf("Plaintext (total %d characters): ", MAX_PLAINTEXT_LENGTH);
    fgets(plaintext, MAX_PLAINTEXT_LENGTH+3, stdin);
    printf("Ciphertext (in hex format): ");
    fgets(ciphertext, MAX_BUF_LEN, stdin);
    */
```

```
    //Hardcoded for debugging
    char ciphertext[] = "8d20e5056a8d24d0462ce74e4904c1b513e10d1df4a2ef2ad4540fae1ca0aaf9";
    char plaintext[] = "This is a top secret.";


    printf("\nIV: ");
    int i = 0;
    for( ; i < 16; i++)
        printf("%d", IV[i]);
    printf("\nCiphertext:\n%s, %d", ciphertext, strlen(ciphertext));
    printf("\nPlaintext: %s", plaintext);


    AES_KEY aeskey;
    unsigned char word[MAX_WORD_LEN];
    int wordcount = 0;
    int ret;
    unsigned char* temp_str;

    while(42) {

         memset(word, ' ', MAX_WORD_LEN);
         ret = fscanf(fp, "%16s", word);
         if(ret == EOF)
             break;
         wordcount++;

         //Encrypt the plaintext with the word as the key and compare the ciphertexts
         memset(temp_cipher, '\0', MAX_BUF_LEN);
         AES_set_encrypt_key (word, MAX_BUF_LEN, &aeskey);
         AES_cbc_encrypt (plaintext, temp_cipher, MAX_BUF_LEN, &aeskey, IV, AES_ENCRYPT);

        temp_str = hexdump(temp_cipher, MAX_BUF_LEN);
    /*

        if(wordcount%3000 == 0) {
        //Just for debugging.
            printf("temp:\n%s\n", temp_str);
        }*/

         if(strncmp(ciphertext, temp_str, MAX_BUF_LEN) == 0) {


             printf("\nKey found after trying %d words!\nKey without quotes:\n\"%s\"", wordcount, word);
             break;
         }
    }

    printf("\nReached end of word list, no key found after trying %d words. Aborting.\n", wordcount);


}
```