

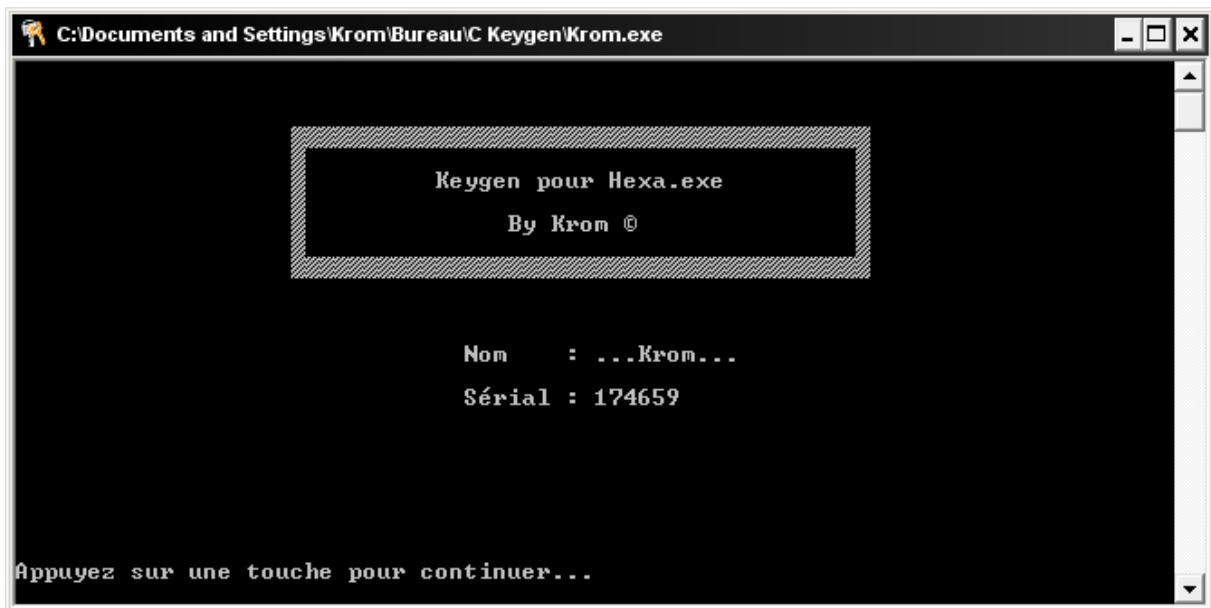
Cours N° 5

Ce cours vous montrera comment créer un Keygen en DOS pour Hexa.exe.

Télécharger le Keygen :

- <http://www.KromCrack.com/prog/Keygen.exe>

Voici a quoi ressemble le Keygen :



L'avantage de créer un Keygen sous DOS est qu'on ne doit pas se soucier de la mise en page et de l'affichage dans une fenêtre et que donc on peut se concentrer sur l'algorithme (DOS utilise l'application " cmd.exe ", est c'est cette application qui gère tout l'affichage) Pour créer un Keygen, il vous faudra de nombreuses notions en programmation. Un excellent cours pour débutant et confirmé à été fait par m@teo21 de l'équipe du [Site du Zer0](#) , merci à lui :)

Avant de continuer, je conseille d'avoir des bases de programmations, savoir ce qu'est une variable, une boucle, un code source, connaître au moins quelques instructions fondamentales (printf(""); / if() - else / while() - for() / %d, %c) et savoir avec quels outils peut on modifier et compiler un projet en C.

```
/*
 *      Keygen.c
 *
 *      - Keygen pour Hexa.exe
 *
 *      Auteur           : - Krom -
 *      Date de Création  : - 07 - 09 - 2007 -
 *      Dernière modification : - 16 - 11 - 2007 -
 *
 */
*****/

// Importations des bibliothèques

#include <stdio.h>
#include <stdlib.h>

// Méthode main : début de l'exécution du programme

int main()
{
    // Déclarations des variables

    char car = 0;
    long serial = 0;
    long result = 0;
    long count = 0;
    long count2 = 0;
    char caractere[24];
    long algo1[33] = {0x00, 0xD5, 0x00, 0xF7, 0x24, 0x5D, 0x5C, 0x0A, 0x8C,
0x10, 0xD6, 0x9E, 0xB5, 0x9C, 0x66, 0x00, 0x24, 0x6F, 0x20, 0x44, 0x4D,
0xD4, 0x63};
    long ascii = 0;
    for(count; count < 33; count++)
    {
        caractere[count] = 0;
    }
    count = 0;

    // Début du Code

    printf("\n\n");
    printf("                22222222222222222222222222222222222222222222222\n");
    printf("                2                                2\n");
    printf("                Keygen pour Hexa.exe                2\n");
    printf("                2                                2\n");
    printf("                By Krom %c                            2\n", 184);
    printf("                2                                2\n");
    printf("                22222222222222222222222222222222222222222222222\n");
    printf("                Nom : ");

    while(caractere[count] != 13 && count < 23 && car != 95 && caractere[count] != 32)
    {
        count++;
        caractere[count] = getch();
        printf("%c", caractere[count]);
    }
    caractere[count] = 0;
    if(count > 10)
    {
        for(count2 = 1; count2 < count; count2++)
        {
            result = algo1[count2] ^ caractere[count2];
            count2--;
            serial = serial + (result * pow(2, count2));
            count2++;
        }
        printf("\n                S%crial : %d \n\n", 130, serial);
    }
    else
    {
        printf("\n                Le nom doit contenir au minimum 10 caract%cs\n", 138);
    }
    printf("\n\n\n\n\n\n\n");
    system("PAUSE");
    return 0;
} // Fin de la méthode main
```

Il vous faudra des bases de programmations pour comprendre ce code mais je peut quand même vous expliquer où se retrouve notre fameux algorithme qui a été trouvé de le Cours 4.

Alors, la partie du haut du programme sert à importer les bibliothèques qui nous seront utiles (<stdio.h> et <stdlib.h>) et à déclarer les variables qui nous seront utiles pour la suite du programme. Voici le détail de l'algorithme de génération du Sériat :

```

printf("                                Nom      : "); (1)

while(caracter[count] != 13 && count < 23 && car != 95 && caracter[count] != 32) (2)
{
    count++;
    caracter[count] = getch();
    printf("%c", caracter[count]);
}
caracter[count] = 0;
if(count > 10) (3)
{
    for(count2 = 1; count2 < count; count2++)
    {
        result = algo1[count2] ^ caracter[count2]; (4)
        count2--;
        serial = serial + (result * pow(2, count2)); (5)
        count2++;
    }
    printf("\n\n                                S%crial : %d \n\n", 130, serial); (6)
}
else
{
    printf("\n\n                                Le nom doit contenir au minimum 10 caractères ... \n\n", 138); (7)
}

```

(1) Affichage du texte qui nous demande un nom d'enregistrement.

(2) Boucle d'enregistrement de frappe (boucle while = tant que X est vrai), à chaque caractère entré, le chiffre ou la lettre est stocké dans ce qu'on appelle un tableau. Un tableau n'est qu'en fait qu'une variable à taille indéterminée, l'intérêt dans ce cas d'avoir un tableau est qu'on ne sait pas combien l'utilisateur va entrer de caractère. Donc, cette boucle dit que "tant que" (Boucle while) l'on ne presse pas "Enter", la boucle enregistre chaque caractères.

(3) Cette instruction permet de contrôler si le nom entré a bien 10 caractères ou plus, si il en contient moins, inutile de continuer plus loin.

(4) Ici, souvenez vous de l'algorithme, compare des nombres en Hexadécimal avec des lettres entrés avec un XOR. Eh bien ici " algo1[count2] " contient ces fameux nombres et " caracter[count2] " contient chaque lettres du nom entré. Le " ^ " est en XOR en C, donc la, on a déjà fait le XOR entre les deux valeurs et le résultat est mis dans la variable " result ".

(5) Maintenant, pour reproduire correctement l'algorithme, on doit multiplier par puissance de 2 . donc la on voit notre variable " result " qui contient le résultat de notre XOR multiplié par puissance de 2 grâce à l'instruction " powl " et met le résultat de la multiplication dans la variable " serial "

(6) Affiche le résultat de toute ces opérations qui sont contenus dans la variable " serial ".

(7) Si le nom contient moins de 10 caractères, le Keygen affiche directement le message d'erreur sans passer par le calcul de Sérial.

J'espère que ce cours a été clair ;)

Si vous avez rencontré une erreur ou que quelque chose ne marche pas, vous pouvez m'envoyer un mail à **Admin@KromCrack.com** ou en parler sur le forum :

- <http://www.KromCrack.com/forum/>