Rapport Exercices Sécurité des Applications : Série 2

Thomas Dagier

October, 10, 2020

1 EuroMillions (RND3)

Le but de cet exercice est de retrouver la date à laquelle le tirage des bons numéros pour l'EuroMillions a été fait. Pour l'exercice précédent, nous savions dans les grandes lignes comment les numéros étaient générés. Ici, la partie complexe est de comprendre par quels moyens les numéros sont générés pour reproduire la bonne séquence.

La première chose que j'ai fait est d'ouvrir le binaire avec ghidra pour tenter de comprendre le cheminement.

```
19      puts("Press Any Key to launch draw lots");
20      getchar();
21      local_28 = 0;
22      while (local_28 < 0x32) {
23        *(int *)((long)pvVar2 + (long)local_28 * 4) = local_28 + 1;
24        local_28 = local_28 + 1;
25      }
26      local_24 = 0;
27      while (local_24 < 0xc) {
28        *(int *)((long)pvVar3 + (long)local_24 * 4) = local_24 + 1;
29        local_24 = local_24 + 1;
30      }
31      secure_shuffle(pvVar2,0x32);
32      secure_shuffle(pvVar3,0xc);
33      printf("Draw : ");
34      local_20 = 0;
35      while (local_20 < 5) {
36        printf("%d ",(ulong)*(uint *)((long)pvVar2 + (long)local_20 * 4));
37        local_20 = local_20 + 1;
38      }
39      printf(" - ");
40      local_1c = 0;
41      while (local_1c < 2) {
42        printf("%d ",(ulong)*(uint *)((long)pvVar3 + (long)local_1c * 4));
43        local_1c = local_1c + 1;
44      }
45      putchar(10);
46    } while( true );
```

Figure 1: binaire avec ghidra

Le code est assez long mais la partie très importante est la fonction `secure_suffle()` (lignes 31 et 32). On remarque avec cette partie du code que `secure_suffle()` prend en paramètre un `void*` et une constante (soit 50 soit 12). J'en ai donc déduit, comme la fonction est appelée 2 fois sur 2 tableaux différents que `pvVar2` est le tableau contenant les numéros normaux de l'EuroMillions d'après Wikipédia et que `pvVar3` est le tableau des numéros "chance".

Ce qui m'a aussi mit sur la voie sont les boucles `while()` (lignes 22 et 27). Ce que l'on peut comprendre c'est que la première boucle va remplir le tableau `pvVar2` des valeurs entières de 1 à 50 et la seconde boucle va remplir le tableau `pvVar3` des valeurs entières de 1 à 12.

On a donc 2 tableaux qui sont modifiés tour à tour par la fonction `secure_shuffle()`. Ce que l'on remarque ensuite, c'est que les 2 boucles `while()` (lignes 35 et 41) font des `printf()` ce qui semble signifier que c'est ici que les valeurs des 2 tableaux sont lues. La manière de seléctionner les valeurs est toute simple : on prend les 5 premières valeurs du premier tableau et les 2 premières valeurs du tableau des numéros "chance".

Il faut alors comprendre comment fonctionne `secure_shuffle()` puis faire un code qui permet de reporduire les actions du binaire d'origine pour générer les bonnes combinaisons.

```
iVar2 = rand();
local_1c = 0;
while (local_1c < iVar2 % 5 + 2) {
  local_18 = 0;
  while (local_18 < iParm2) {
    iVar3 = rand();
    uVar1 = *(undefined4 *)(lParm1 + (long)local_18 * 4);
    *(undefined4 *)(lParm1 + (long)local_18 * 4) =
         *(undefined4 *)(lParm1 + (long)(iVar3 % iParm2) * 4);
    *(undefined4 *)((long)(iVar3 % iParm2) * 4 + lParm1) = uVar1;
    local_18 = local_18 + 1;
  }
  local_1c = local_1c + 1;
}
```

Figure 2: fonction `secure_shuffle()`

D'après ce que l'on voit sur l'image, on a une première boucle `while()` qui dépend d'un nombre random, modulo 5 auquel on ajoute 2. Une seconde boucle dépend, elle, de la taille du tableau. Dans cette boucle on a l'air de faire des swapp continuellement entre la valeur `local_18` (incrémente de 1 en partant de 1 jusqu'a la taille du tableau -1) et la valeur `ivar3` modulo la taille du tableau.

Ce qui se passe donc dans cette fonction, c'est qu'on fait des swapp entre la valeur à l'indice `local_18` et la valeur à un indice random. Toutes les valeurs du tableaux on donc changées de place et cette manipulation se fait un nombre aléatoire de fois.

À la fin, on a donc 2 tableaux avec des valeurs qui ont changées de place et on prend les 5 premières et 2 premières de chaque tableau. Il ne reste alors qu'a trouver la graine permettant d'obtenir ce tableau en réutilisant le code de l'exercice RND2 :

```c
int main(){
    struct tm start = {0,0,0,1,6,119};
    struct tm stop = {0,0,0,1,0,120};
    time_t debut = mktime(&start);
    time_t end = mktime(&stop);
    bool flag = true;
    int count = 0;
    for (time_t crt = debut; crt < end; crt++) {
        if(flag){
            srand(crt);
        }
        int numbers[50];
        int luckyNumbers[12];
        for(int i = 1; i <= 50; i++){
            numbers[i-1] = i;
            if(i < 13){
                luckyNumbers[i-1] = i;
            }
        }
        secureShuffle(50, numbers);
        secureShuffle(12, luckyNumbers);
        if(numbers[0] == 17 && numbers[1] == 38 && numbers[2] == 33 &&
           numbers[3] == 22 && numbers[4] == 26 &&
           luckyNumbers[0] == 5 && luckyNumbers[1] == 9){
            printf("%s", ctime(&crt));
            flag = false;
        }
        if(flag == false){
            printf("%d %d %d %d %d - %d %d\n",numbers[0], numbers[1],
            numbers[2], numbers[3], numbers[4], luckyNumbers[0], luckyNumbers[1]);
            count++;
            if(count == 5){
                break;
            }
        }
    }
    return 0;
}
```

Figure 3: code principal(`main()`)

```c
int secureShuffle(int size, int array[size]){
    int rnd = rand();
    for(int i = 0; i < rnd % 5 + 2; i++){
        for(int j = 0; j < size; j++){
            int value = rand() % size;
            int tmp = array[j];
            array[j] = array[value];
            array[value] = tmp;
        }
    }
}
```

Figure 4: fonction `secure_shuffle()`

Le programme va alors générer 2 tableaux qui seront modifiés aléatoirement avec cette fonction pour chaque `srand()` entre le 1er Juillet 2019 et le 1er Janvier 2020. Pour chacune des ces valeurs, on va aussi comparer les tableaux aux valeurs que l'on sait être les bonnes. Si une comparaison s'avère être correcte, on conserve le timestamp et on génère les codes qui feront de nous le grand gagnant de l'EuroMillion :

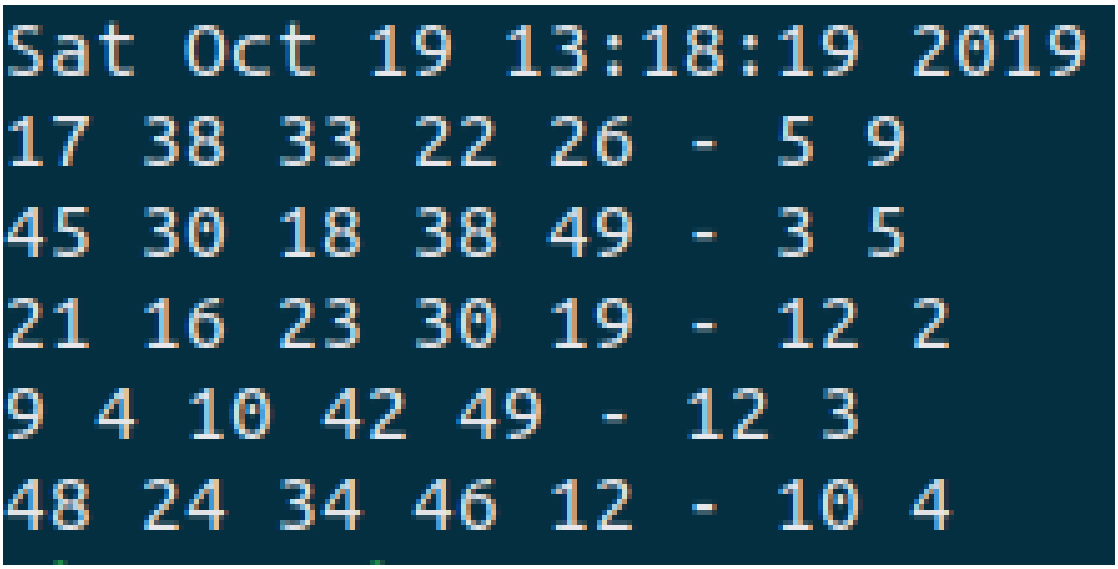

Figure 5: resultat