JUMPLOIT

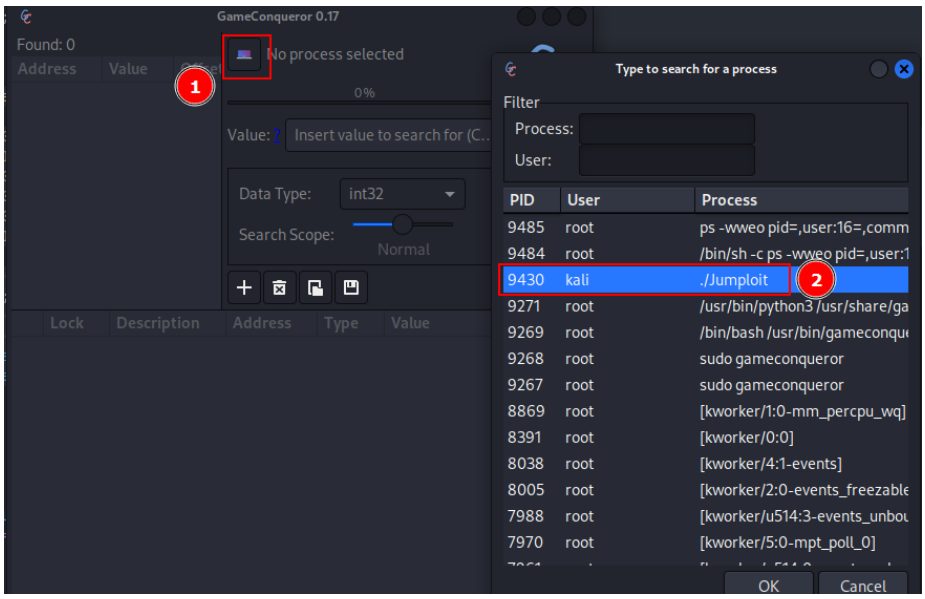
1. Decompilamos el binario con Ghidra.
2. Ejecutamos el binario:

```
./Jumploit
```

3. Ejecutamos en otra terminal:

```
sudo gameconqueror
```

4. Atacheamos el proceso del binario a gameconqueror:



5. En la funcion main del binario decompilado podemos ver el siguiente flujo de código:

```
if (((local_98[0] == 0x300) && (local_84 == 0x20)) && (local_11 != '\0')) {
  local_10 = -0x16;
  local_11 = '\0';
  local_18 = local_18 + 1;
  if (MAX_JUMP_COUNT <= local_18) {
    local_a = '\x01';
    local_9 = '\0';
  }
```

Este flujo nos indica que cuando se supera o se iguala el número de saltos maximo se llama a una función, ya que local_a se cambia a 1:

```
  SDL_DestroyRenderer(local_30);
  SDL_DestroyWindow(local_28);
  if (local_a != '\0') {
    FUN190320252011();
  }
  SDL_Quit();
  return 0;
}
```

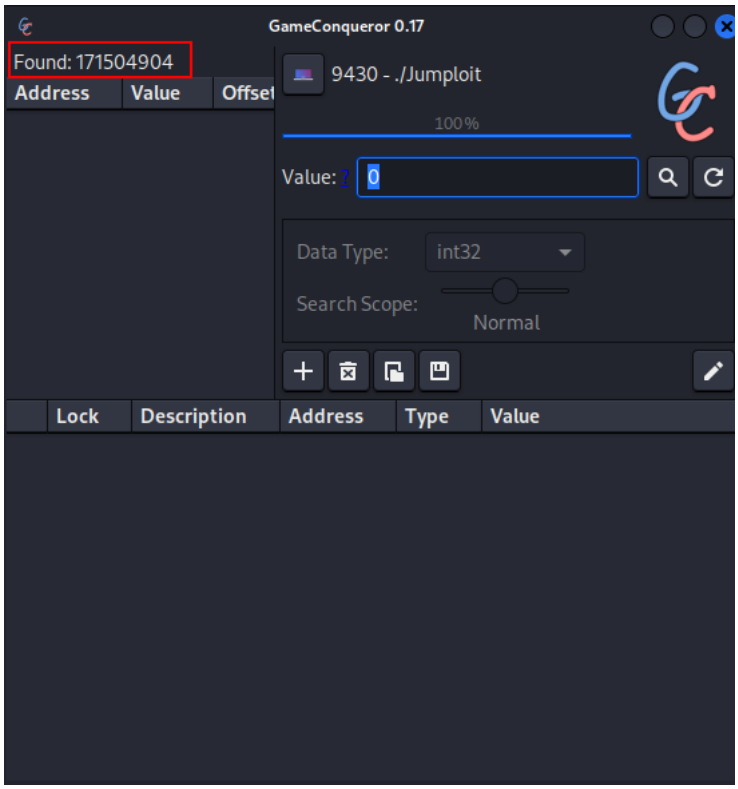
6. En la función llamada podemos ver que se renderiza otra pantalla diferente a la del videojuego, y se llama a otra función:

```
void FUN190320252011(void)

{
  int iVar1;
  int local_58 [14];
  undefined8 local_20;
  undefined8 local_18;
  char local_9;

  local_18 = SDL_CreateWindow(&DAT_00103c00,0x2fff0000,0x2fff0000,0x780,0x438,4);
  local_20 = SDL_CreateRenderer(local_18,0xffffffff,2);
  local_9 = '\x01';
  while (local_9 != '\0') {
    while( true ) {
      iVar1 = SDL_PollEvent(local_58);
      if (iVar1 == 0) break;
      if (local_58[0] == 0x100) {
        local_9 = '\0';
      }
    }
    SDL_SetRenderDrawColor(local_20,0,0,0,0xff);
    SDL_RenderClear(local_20);
    FUN171130062012(local_20,0x96,200,10);
    SDL_RenderPresent(local_20);
    SDL_Delay(0x10);
  }
  SDL_DestroyRenderer(local_20);
  SDL_DestroyWindow(local_18);
  return;
}
```

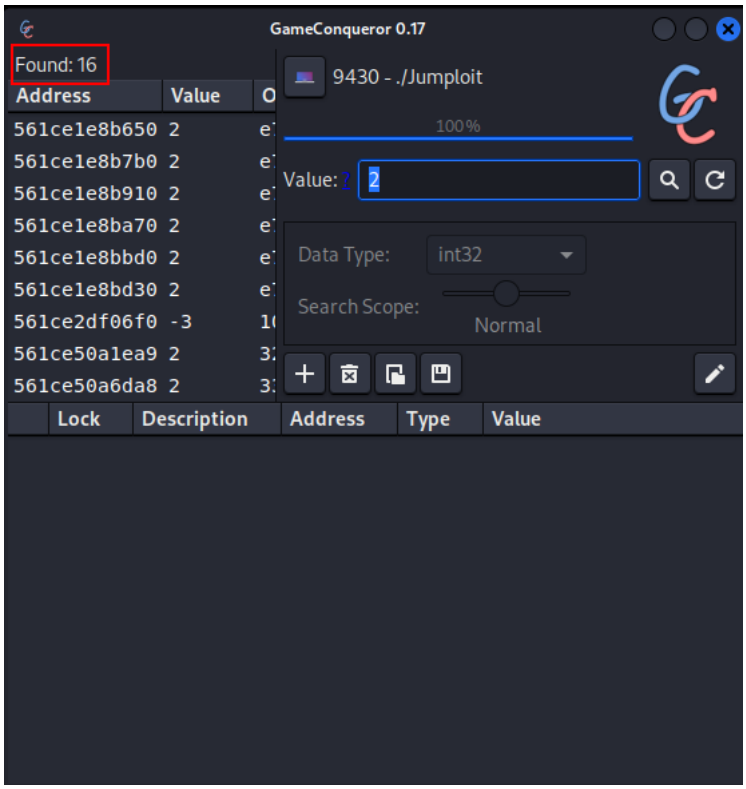
7. En esta función vemos como se define una matrix de 6 x 0x7c (124 en decimal). También podemos ver como se comprueba si cada datos de la matriz es 1, se dibuja una figura blanca en esa posición, para despues renderizarlo:

```
for (local_c = 0; local_c < 6; local_c = local_c + 1) {
  for (local_10 = 0; local_10 < 0x7c; local_10 = local_10 + 1) {
    if (*(int *)((long)local_bb8 + ((long)local_c * 0x7c + (long)local_10) * 4) == 1) {
      local_bc8 = param_2 + local_10 * param_4;
      local_bc4 = param_3 + local_c * param_4;
      local_bc0 = param_4;
      local_bbc = param_4;
      SDL_SetRenderDrawColor(param_1,0xff,0xff,0xff,0xff);
      SDL_RenderFillRect(param_1,&local_bc8);
    }
  }
}
return;
```

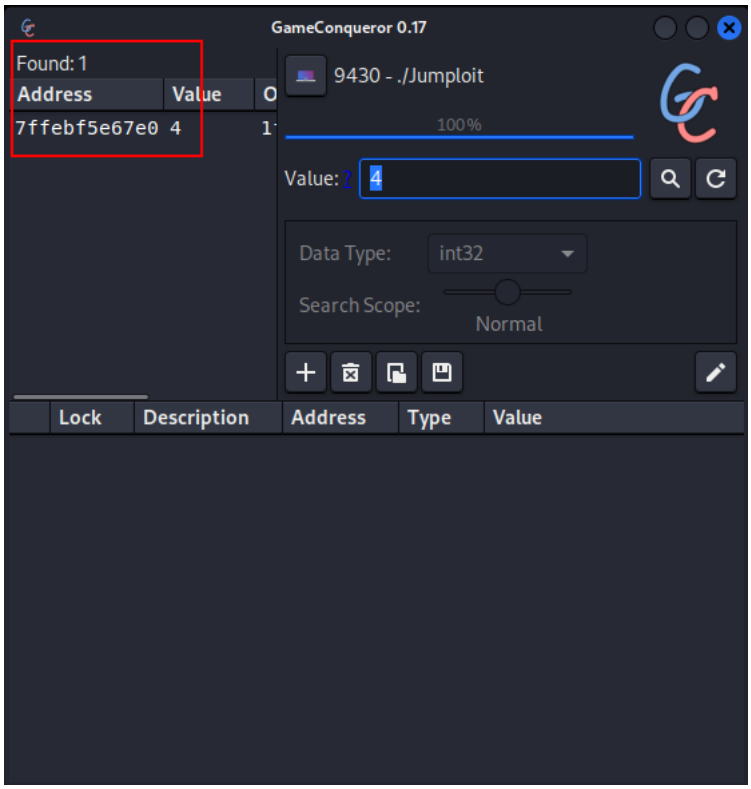
8. Ahora que tenemos la lógica, tenemos que superar los saltos máximos. Volvemos a gameconqueror y buscamos los números enteros con valor igual a 0. Obteniendo muchos valores para saber cual es el referido a los saltos:



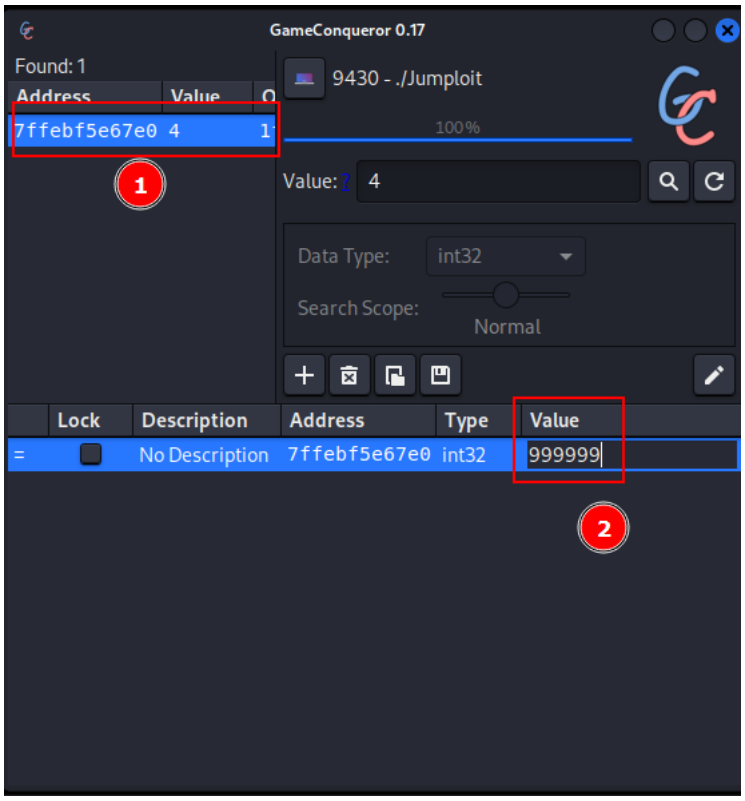
9. Para acotar ese número de resultado, vamos a nuestro juego y saltamos 2 veces. Seguidamente volvemos a realizar la busqueda pero con un valor de 2:



10. Ahora tenemos 16 valores encontrados, como siguen siendo muchos, repetimos el proceso, saltamos 2 veces más y buscamos los valores iguales a 4:



11. Ya tenemos solo 1 valor, el cual será nuestro número de saltos. Ahora hacemos docle click sobre la variable encontrada. Y en el panel inferior cambiamos su valor a 999999:



12. Ahora con la variable a 999999, volvemos al juego y saltamos de nuevo, desbloqueando así la pantalla con la flag:

FLAG: hfctf{h4ck5_t0_jUMP_th3_f14G}