Análisis del uso de desinfectantes y técnicas de fuzzing para la detección de vulnerabilidades en software

Autor: José Luis París Reyes
Tutor: Gustavo Romero López

Índice

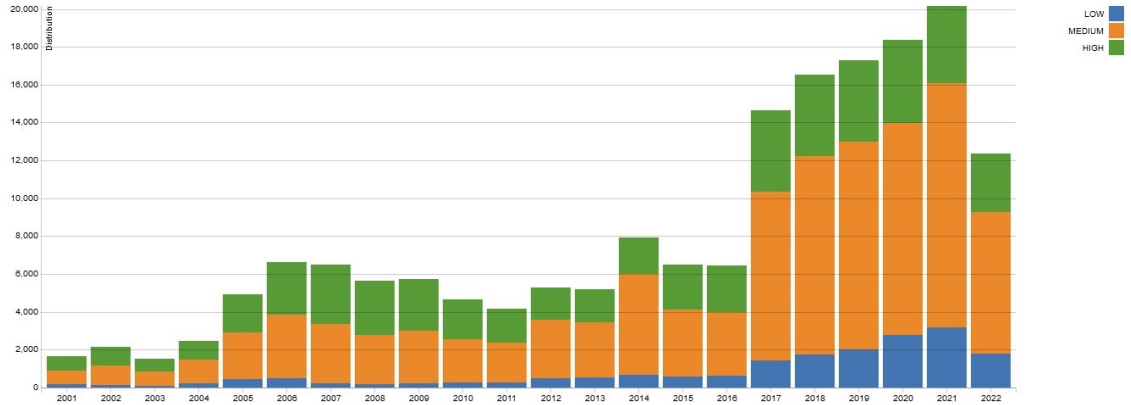
1.

Introducción

Motivación, contexto y metodología del proyecto

Motivación y contexto

- Constante evolución de la tecnología
  - Hardware
  - Software
- Detección de vulnerabilidades
  - Compilación → Análisis estático
  - Ejecución → Análisis dinámico y fuzzing



Metodología

1. Estudio de binarios
    a. Paquetería de Linux
    b. Software de código libre
2. Configuración de compilación → GCC y CLANG
    a. Análisis estático → Informes generados por el analizador
    b. Análisis dinámico → Errores en tiempo de ejecución
3. Fuzzing → AFL++
    a. Informes generados por el software de fuzzing
    b. Depuración con gdb
    c. Compilación con desinfectantes
    d. Reconocimiento del error o vulnerabilidad

2.
Requisitos

Planificación temporal, recursos utilizados y presupuesto estimado

Planificación temporal

Sección	Inicio	Fin	Duración (días)
S1	05/02/2022	02/06/2022	117
S1.1	05/02/2022	03/04/2022	57
S1.2	05/03/2022	10/05/2022	66
S1.3	20/04/2022	02/06/2022	43
S2	05/04/2022	25/04/2022	20
S3	25/04/2022	20/05/2022	25
S4	25/04/2022	25/06/2022	61
S5	12/02/2022	05/07/2022	143
Total			150

Recursos y presupuesto

- Recursos humanos
  - José Luis París Reyes → Autor
  - Gustavo Romero López → Tutor
- Recursos hardware
  - Ordenador de sobremesa y portátil
  - Conexión a internet
- Recursos software
  - Sistema operativo Linux
  - Compiladores con integración de desinfectantes
  - Software de fuzzing
  - Lenguajes de programación

Recurso	Coste	Tiempo	Coste total
Trabajo realizado por el autor	10€	375h	3.750€
Trabajo realizado por el tutor	30€	12h	360€
Alquiler de ordenador de sobremesa personal	5€	375h	1.875€
Alquiler de ordenador portátil personal	3€	50h	150€
Conexión de alta velocidad a internet	30€	6m	180€
Sistema Operativo Linux	0€	-	0€
Compiladores con integración de desinfectantes	0€	-	0€
Software de fuzzing	0€	-	0€
Lenguaje de programación	0€	-	0€
Total			6.315€

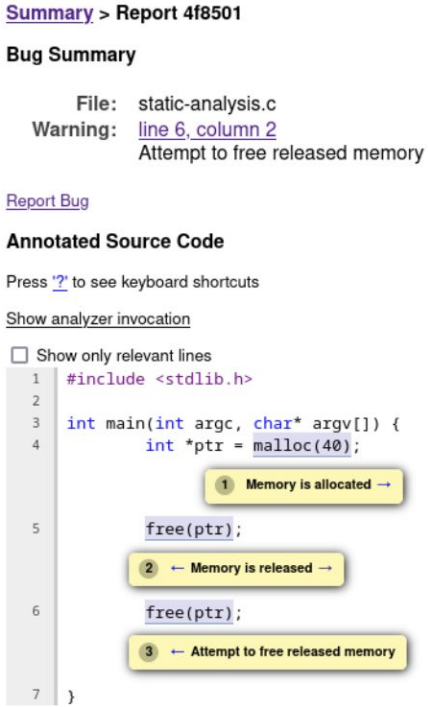
3.
Estado del arte

Análisis estático, análisis dinámico y técnicas de fuzzing

Análisis estático

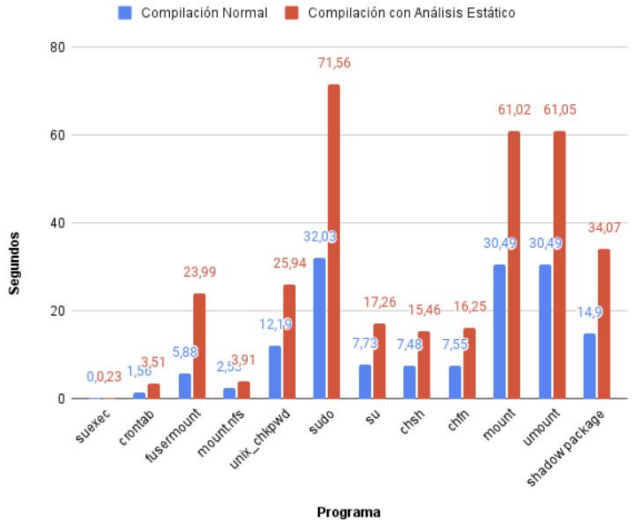
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main(int argc, const char *argv[]) {
    int *ptr = malloc(40);
    free(ptr);
    free(ptr);
    return 0;
}
```

```
→ gcc -fanalyzer static-analysis.c
static-analysis.c: In function 'main':
static-analysis.c:6:9: warning: double-'free' of 'ptr' [CWE-415] [-Wanalyzer-double-free]
    6 |         free(ptr);
      |         ^~~~~~~~~
  'main': events 1-3
    |
    |    4 |         int *ptr = malloc(40);
    |      |                    ^~~~~~~~~~
    |      |                    |
    |      |                    (1) allocated here
    |    5 |         free(ptr);
    |      |         ~~~~~~~~~
    |      |         |
    |      |         (2) first 'free' here
    |    6 |         free(ptr);
    |      |         ~~~~~~~~~
    |      |         |
    |      |         (3) second 'free' here; first 'free' was at (2)
    |
```



Análisis estático II

- Ventajas
  - Automatización de búsqueda de errores → Facilidad para detección y mayor agilidad
  - Estandarización de la salida → Informes generados
- Desventajas
  - Necesidad de acceso directo al código fuente
  - Falsos positivos
  - Mayor tiempo de compilación → Entre 2 y 5 veces mayor



Análisis dinámico

- Tiempo de ejecución
- Multitud de Sanitizers
  - Address Sanitizer
  - Thread Sanitizer
  - Memory Sanitizer
  - Leak Sanitizer
  - Undefined Behavior Sanitizer
- Detectan diferentes tipos de fallos
  - Buffer Overflow
  - Dangling pointer
  - Comportamientos inesperados del programa

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main(int argc, const char *argv[]) {
    char *s = malloc(100);
    strcpy(s, "Hello world!");
    printf("string is: %s\n", s);
    return 0;
}
```

```
→ ./leak
string is: Hello world!

=================================================================
==17859==ERROR: LeakSanitizer: detected memory leaks

Direct leak of 100 byte(s) in 1 object(s) allocated from:
    #0 0x7f8bb5d28dd9 in __interceptor_malloc /usr/src/debug/gcc/libsanitizer/asan/asan_malloc_linux.cpp:145
    #1 0x55b4aae351b1 in main (/home/xfear/Desktop/TFG/address-sanitizer/leak+0x11b1)
    #2 0x7f8bb5a9430f in __libc_start_call_main (/usr/lib/libc.so.6+0x2d30f)

SUMMARY: AddressSanitizer: 100 byte(s) leaked in 1 allocation(s).
```

Análisis dinámico II

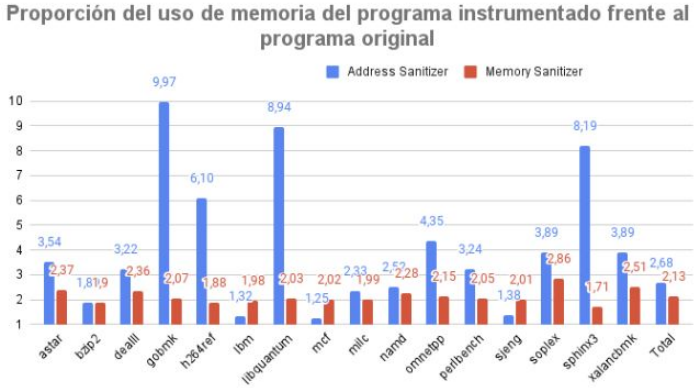
```c
#include <pthread.h>
#include <stdio.h>

int Global;

void *Thread1(void *x) {
  Global++;
  return NULL;
}

void *Thread2(void *x) {
  Global--;
  return NULL;
}

int main() {
  pthread_t t[2];
  pthread_create(&t[0], NULL, Thread1, NULL);
  pthread_create(&t[1], NULL, Thread2, NULL);
  pthread_join(t[0], NULL);
  pthread_join(t[1], NULL);
}
```

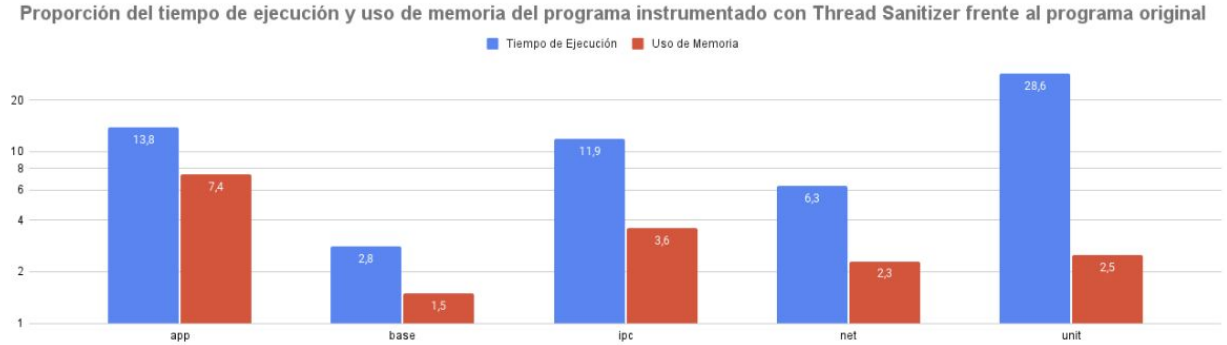
```
→ ./simple_race
==================
WARNING: ThreadSanitizer: data race (pid=24180)
  Write of size 4 at 0x559764906100 by thread T2:
    #0 Thread2 /home/xfear/Desktop/TFG/thread-sanitizer/simple_race.c:12:9 (simple_race+0xdeaf4)

  Previous write of size 4 at 0x559764906100 by thread T1:
    #0 Thread1 /home/xfear/Desktop/TFG/thread-sanitizer/simple_race.c:7:9 (simple_race+0xdeaa4)

  Location is global 'Global' of size 4 at 0x559764906100 (simple_race+0x0000033c2100)

  Thread T2 (tid=24183, running) created by main thread at:
    #0 pthread_create <null> (simple_race+0x9288e)
    #1 main /home/xfear/Desktop/TFG/thread-sanitizer/simple_race.c:19:3 (simple_race+0xdeb5f)

  Thread T1 (tid=24182, finished) created by main thread at:
    #0 pthread_create <null> (simple_race+0x9288e)
    #1 main /home/xfear/Desktop/TFG/thread-sanitizer/simple_race.c:18:3 (simple_race+0xdeb48)

SUMMARY: ThreadSanitizer: data race /home/xfear/Desktop/TFG/thread-sanitizer/simple_race.c:12:9 in
 Thread2
==================
ThreadSanitizer: reported 1 warnings
```

Análisis dinámico III

- Ventajas
  - Automatización de búsqueda de errores → Facilidad para detección y mayor agilidad
  - Estandarización de la salida → Informes generados
- Desventajas
  - Necesidad de acceso directo al código fuente
  - Degradación en el rendimiento del programa
    - Memoria → Entre 5 y 10 veces más
    - Tiempo de ejecución → Entre 5 y 15 veces más

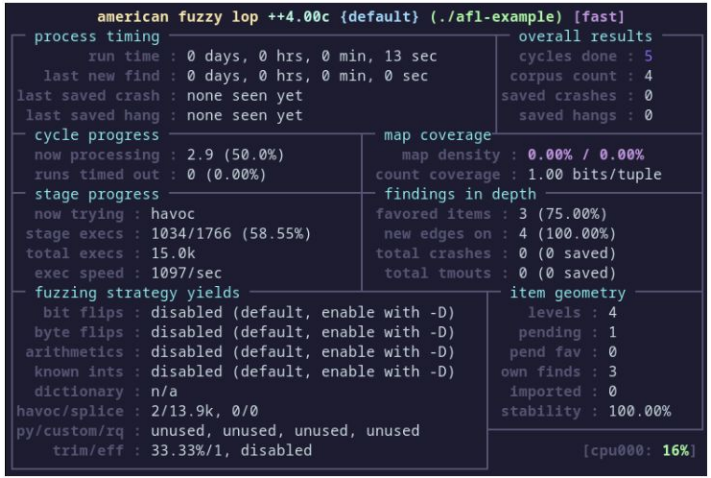
Análisis dinámico IV





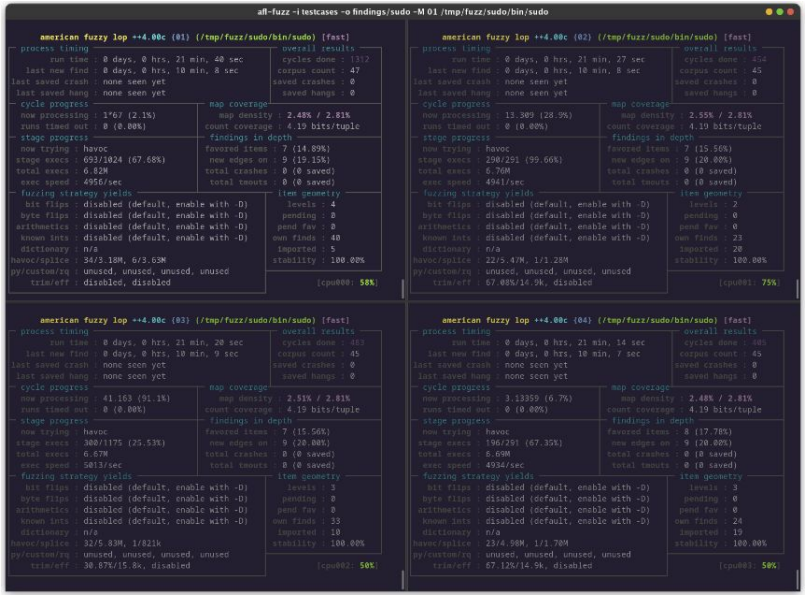
Fuzzing

- Tipos
  - Aleatorio
  - Basado en mutación
  - Basado en generación → Definición de reglas
  - Evolutivo → Algoritmos genéticos
- AFL++
  - Software de fuzzing evolutivo
  - Requiere compilación instrumentada para mejores resultados
  - Configuración sencilla



Fuzzing II

- Ventajas
  - Detección de fallos
  - Ejecución paralela → No requiere atención del usuario
  - Escalabilidad horizontal
- Desventajas
  - Tiempo de ejecución infinito
  - Uso de CPU intensivo
  - Binarios de línea de comandos requieren modificación de código fuente
    - Sudo
    - Cat
    - Cp



4.
Análisis

Estudio del problema

Análisis

1. Paquetería de la distribución Linux
   a. Encontrar a qué paquete pertenece un binario
   b. Descarga de archivos fuente del paquete
2. Detectar el software interesante de estudio → bit SUID activado
3. Configuración del entorno de compilación
4. Análisis de los informes generados
   a. Análisis estático
   b. Análisis dinámico
5. Uso de técnicas de fuzzing y comprobación de fallos

Bug Summary

Bug Type	Quantity	Display?
All Bugs	**177**	☑
API		
Argument with 'nonnull' attribute passed null	1	☑
Logic error		
Assigned value is garbage or undefined	48	☑
Dereference of undefined pointer value	22	☑
Result of operation is garbage or undefined	3	☑
Uninitialized argument value	98	☑
Unused code		
Dead assignment	5	☑

5.
Diseño e implementación

Diseño de la solución e implementación

Diseño e implementación

- Descarga de código fuente de los paquetes
- Generar el archivo Makefile y compilar los binarios
- Calcular el tiempo de ejecución y compilación de los binarios
- Configuración del software de fuzzing para inicio y finalización de las pruebas

Índice	Nombre	Paquete	Índice	Nombre	Paquete
1	su	shadow	15	expiry	shadow
2	fusermount	fuse2	16	sudo	sudo
3	fusermount3	fuse3	17	ksu	krb5
4	chsh	shadow	18	sg	shadow
5	cdrecord	cdrtools	19	mount	util-linux
6	mount.nfs	nfs-utils	20	crontab	cronie
7	chfn	shadow	21	readcd	cdrtools
8	rscsi	cdrtools	22	gpasswd	shadow
9	nvidia-modprobe	nvidia-utils	23	chage	shadow
10	passwd	shadow	24	unix_chkpwd	pam
11	suexec	apache	25	umount	util-linux
12	pkexec	polkit	26	newgrp	util-linux
13	mount.cifs	cifs-utils	27	mount.ecryptfs_private	ecryptfs-utils
14	cdda2wav	cdrtools			

```
#!/bin/bash

file=$1
while read line, do
    package = $(pacman -Qo $(which $line) | cut -d " " -f 5 | sort
        | uniq)
    asp export $package
done < $file
```

6.
Pruebas

Resultados de las pruebas realizadas de análisis estático, análisis dinámico y fuzzing

Análisis estático

Error	crontab	sudo	su	chsh	chfn	mount
Allocator sizeof operand mismatch	1	-	-	-	-	-
Argument with "nonnull" attribute passed null	1	1	-	-	-	-
Dead assignment	1	5	2	2	2	2
Dead increment	1	-	-	-	-	-
Dead nested assignment	1	-	8	8	8	8
Use of zero allocated	1	-	-	-	-	-
Assigned value is garbage or undefined	-	48	-	-	-	-
Dereference of undefined pointer value	-	22	-	-	-	-
Result of operation is garbage or undefined	-	3	2	2	2	2
Unitialized argument value	-	98	-	-	-	-
Memory leak	-	-	-	10	-	-

Error	umount	shadow	unix_chkpwd	fusemount	mount.nfs
Argument with "nonnull" attribute passed null	1	1	-	-	-
Dead assignment	1	5	2	2	2
Dead increment	1	-	-	-	-
Dead nested assignment	1	-	8	8	8
Use of zero allocated	1	-	-	-	-
Assigned value is garbage or undefined	-	48	-	-	-
Result of operation is garbage or undefined	-	3	2	2	2
Unitialized argument value	-	98	-	-	-
Memory leak	-	-	-	10	-

Análisis estático II

```
if (pw->pw_shell) {
        (void)memcpy(cp, pw->pw_shell, ssize);
        newpw->pw_shell = cp;
        cp += ssize;
```

Value stored to 'cp' is never read

```
}
```

Dead Increment - Cronie

```
if (ctl.interactive)
```

9 ← Taking true branch →

```
        ask_info(&ctl);

add_missing(&ctl);
```

10 ← Calling 'add_missing' →

23 ← Returned allocated memory →

```
if (!ctl.changed) {
```

24 ← Assuming field 'changed' is 0 →

25 ← Taking true branch →

```
        printf(_("Finger information not changed.\n"));
```

26 ← Potential leak of memory pointed to by 'ctl.newf.office_phone'

```
        return EXIT_SUCCESS;
}

return save_new_data(&ctl) == 0 ? EXIT_SUCCESS : EXIT_FAILURE;
```

Memory Leak - Chfn

Análisis dinámico

```
Uninitialized bytes in __interceptor_fopen at offset 0 inside [0x701000000140, 14)
==226143==WARNING: MemorySanitizer: use-of-uninitialized-value
    #0 0x7f61e0dd4825  (/usr/lib/libpam.so.0+0x5825)
    #1 0x7f61e0dd5c7f  (/usr/lib/libpam.so.0+0x6c7f)
    #2 0x7f61e0dd7642  (/usr/lib/libpam.so.0+0x8642)
    #3 0x55b8f2d21006 in supam_authenticate su-common.c
    #4 0x55b8f2d1b6f3 in su_main (/home/xfear/Desktop/TFG/binaries_source_code/util-linux/src/util-lin
ux-2.38/su+0xa96f3)
    #5 0x55b8f2d187cb in main (/home/xfear/Desktop/TFG/binaries_source_code/util-linux/src/util-linux-
2.38/su+0xa67cb)
    #6 0x7f61e0adf28f  (/usr/lib/libc.so.6+0x2928f)
    #7 0x7f61e0adf349 in __libc_start_main (/usr/lib/libc.so.6+0x29349)
    #8 0x55b8f2c94594 in _start /build/glibc/src/glibc/csu/../sysdeps/x86_64/start.S:115

SUMMARY: MemorySanitizer: use-of-uninitialized-value (/usr/lib/libpam.so.0+0x5825)
Exiting
```

MemorySanitizer - su

LeakSanitizer - gpasswd

```
==560425==ERROR: LeakSanitizer: detected memory leaks

Direct leak of 48 byte(s) in 1 object(s) allocated from:
    #0 0x7f7896b0cf68 in __interceptor_calloc /usr/src/debug/gcc/libsanitizer/lsan/lsan_interceptors.cpp:90
    #1 0x55fa25a7d7b5 in __pw_dup /home/xfear/Desktop/TFG/binaries_source_code/shadow/src/shadow-4.11.1/lib/pwmem.c:24

Indirect leak of 35 byte(s) in 5 object(s) allocated from:
    #0 0x7f7896b0d453 in __interceptor_malloc /usr/src/debug/gcc/libsanitizer/lsan/lsan_interceptors.cpp:75
    #1 0x7f789692fd5e in __strdup (/usr/lib/libc.so.6+0x9ed5e)

SUMMARY: LeakSanitizer: 83 byte(s) leaked in 6 allocation(s).
```

Análisis dinámico II

```
==585329==ERROR: LeakSanitizer: detected memory leaks

Direct leak of 48 byte(s) in 1 object(s) allocated from:
    #0 0x7fd94b357411 in __interceptor_calloc /usr/src/debug/gcc/libsanitizer/asan/asan_malloc_linux.cpp:
77
    #1 0x5564df6f6115 in __pw_dup /home/xfear/Desktop/TFG/binaries_source_code/shadow/src/shadow-4.11.1/l
ib/pwmem.c:24

Indirect leak of 12 byte(s) in 1 object(s) allocated from:
    #0 0x7fd94b30afaa in __interceptor_strdup /usr/src/debug/gcc/libsanitizer/asan/asan_interceptors.
cpp:439
    #1 0x5564df6f62a4 in __pw_dup /home/xfear/Desktop/TFG/binaries_source_code/shadow/src/shadow-4.11.1/l
ib/pwmem.c:54

Indirect leak of 9 byte(s) in 1 object(s) allocated from:
    #0 0x7fd94b30afaa in __interceptor_strdup /usr/src/debug/gcc/libsanitizer/asan/asan_interceptors.cpp:
439
    #1 0x5564df6f62e2 in __pw_dup /home/xfear/Desktop/TFG/binaries_source_code/shadow/src/shadow-4.11.1/l
ib/pwmem.c:61

Indirect leak of 6 byte(s) in 1 object(s) allocated from:
    #0 0x7fd94b30afaa in __interceptor_strdup /usr/src/debug/gcc/libsanitizer/asan/asan_interceptors.cpp:
439
    #1 0x5564df6f625e in __pw_dup /home/xfear/Desktop/TFG/binaries_source_code/shadow/src/shadow-4.11.1/l
ib/pwmem.c:47

Indirect leak of 2 byte(s) in 1 object(s) allocated from:
    #0 0x7fd94b30afaa in __interceptor_strdup /usr/src/debug/gcc/libsanitizer/asan/asan_interceptors.cpp:
439
    #1 0x5564df6f6218 in __pw_dup /home/xfear/Desktop/TFG/binaries_source_code/shadow/src/shadow-4.11.1/l
ib/pwmem.c:40

SUMMARY: AddressSanitizer: 77 byte(s) leaked in 5 allocation(s).
```

AddressSanitizer - passwd

Fuzzing

```
root:~# xxd /tmp/out/f01/crashes/id:000006*
00000000: 73df 3131 7375 6480 7375 6480 5e5e 3d5f  s.11sud.sud.^^=_
00000010: 5e5e 555e 5e5e 7375 646f 0192 8564 6f65  ^^U^^^sudo...doe
00000020: 6469 7400 2d41 002d 7300 8065 002d 6700  dit.-A.-s..e.-g.
00000030: 2d5c 002d 6800 2d42 5e5e 555e 5e8c 6d64  -\.-h.-B^^U^^.md
00000040: 6f01 9275 646f 6564 6974 002d 4100 2d62  o..udoedit.-A.-b
00000050: 002d 6500 2d00 2d31 7375 6480 7375 6480  .-e.-.-1sud.sud.
00000060: 5e46 5e5f 5e5e 555e 5e5e 7375 646f 0192  ^F^_^^U^^^sudo..
00000070: 8564 6f65 6469 7400 2d41 002d 7300 8065  .doedit.-A.-s..e
00000080: 002d 6700 2d48 002d 6800 2d42 5e5e 555e  .-g.-H.-h.-B^^U^
00000090: 5e5e 8c75 646f 0192 7564 6f65 6469 73f5  ^^.udo..udoedis.
000000a0: 2d41 002d 6200 2d65 005e 555e 5e5e 8c75  -A.-b.-e.^U^^^.u
000000b0: 646f 0192 7564 6f65 6469 7400 2d41 002d  do..udoedit.-A.-
000000c0: 6200 2d65 ee2d 002d 3173 7564 8073 7564  b.-e.-.-1sud.sud
000000d0: 805e 465e 5f5e 5e55 5e5e 5e00 8065 002d  .^F^_^^U^^^..e.-
000000e0: 6700 2d48 002d 6800 2d42 5e5e 555e 5e00  g.-H.-h.-B^^U^^.
000000f0: 8000 006f 0192 7564 6f65 6469 7400 2d41  ...o..udoedit.-A
00000100: 002d 6200 2d65 002d 002d 4300 2d45 2d00  .-b.-e.-.-C.-E-.
00000110: 2d43 002d 4500 2d65 002d 6700 2d48 002d  -C.-E.-e.-g.-H.-
00000120: 2d69 ef2d 4b00 4300 2d45 002d 0100 2d67  -i.-K.C.-E.-..-g
00000130: 052d 4800 2d2d 6900 2d4b 002d            .-H.--i.-K.-
```

```
root:~# xxd minimized.testcase
00000000: 3065 6469 7400 2d73 0030 3030 3030 5c00  0edit.-s.00000\.
00000010: 3030 3030 3030 3030 3030 3030 3030 3030  0000000000000000
00000020: 3030 3030 3030 3030 3030 3030 3030 3030  0000000000000000
00000030: 3030 3030 3030 3030 3030 3030 3030 3030  0000000000000000
00000040: 30                                       0
```

Fuzzing II

```
(gdb) up
#1  0x00007ffff7bf6859 in __GI_abort () at abort.c:79
79      abort.c: No such file or directory.
(gdb) up
#2  0x00007ffff7c6126e in __libc_message (action=action@entry=do_abort,
    fmt=fmt@entry=0x7ffff7d8b298 "%s\n") at ../sysdeps/posix/libc_fatal.c:155
155     ../sysdeps/posix/libc_fatal.c: No such file or directory.
(gdb) up
#3  0x00007ffff7c692fc in malloc_printerr (
    str=str@entry=0x7ffff7d8da50 "malloc(): invalid size (unsorted)") at malloc.c:5347
5347    malloc.c: No such file or directory.
(gdb) up
#4  0x00007ffff7c6c0b4 in _int_malloc (av=av@entry=0x7ffff7dc0b80 <main_arena>,
    bytes=bytes@entry=262148) at malloc.c:3736
3736    in malloc.c
(gdb) up
#5  0x00007ffff7c6e154 in __GI___libc_malloc (bytes=262148) at malloc.c:3058
3058    in malloc.c
(gdb) up
#6  0x00005555555a08c7 in sudo_getgrouplist2_v1 (name=0x5555560eb9e8 "root", basegid=0,
    groupsp=0x7fffffffcd38, ngroupsp=0x7fffffffcd34) at ./getgrouplist.c:101
101             groups = reallocarray(NULL, grpsize, sizeof(*groups));
```

Fuzzing III

```
==109209==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x606000000959 at pc 0x0000006569ee bp 0x7fff2e178750
 sp 0x7fff2e178748
WRITE of size 1 at 0x606000000959 thread T0
    #0 0x6569ed in set_cmnd /pwd/sudo-ASAN/plugins/sudoers/./sudoers.c:868:10
    #1 0x64a844 in sudoers_policy_main /pwd/sudo-ASAN/plugins/sudoers/./sudoers.c:306:19
    #2 0x61d0e3 in sudoers_policy_check /pwd/sudo-ASAN/plugins/sudoers/./policy.c:872:11
    #3 0x5506b9 in policy_check /pwd/sudo-ASAN/src/./sudo.c:1140:11
    #4 0x5433dc in main /pwd/sudo-ASAN/src/./sudo.c:255:11
    #5 0x7f6ce6afd082 in __libc_start_main /build/glibc-SzIz7B/glibc-2.31/csu/../csu/libc-start.c:308:16
    #6 0x41da2d in _start (/pwd/sudo-ASAN/src/sudo+0x41da2d)

0x606000000959 is located 0 bytes to the right of 57-byte region [0x606000000920,0x606000000959)
allocated by thread T0 here:
    #0 0x49616d in malloc (/pwd/sudo-ASAN/src/sudo+0x49616d)
    #1 0x655f41 in set_cmnd /pwd/sudo-ASAN/plugins/sudoers/./sudoers.c:854:36
    #2 0x64a844 in sudoers_policy_main /pwd/sudo-ASAN/plugins/sudoers/./sudoers.c:306:19
    #3 0x61d0e3 in sudoers_policy_check /pwd/sudo-ASAN/plugins/sudoers/./policy.c:872:11
    #4 0x5506b9 in policy_check /pwd/sudo-ASAN/src/./sudo.c:1140:11
    #5 0x5433dc in main /pwd/sudo-ASAN/src/./sudo.c:255:11
    #6 0x7f6ce6afd082 in __libc_start_main /build/glibc-SzIz7B/glibc-2.31/csu/../csu/libc-start.c:308:16

SUMMARY: AddressSanitizer: heap-buffer-overflow /pwd/sudo-ASAN/plugins/sudoers/./sudoers.c:868:10 in set_cmnd
Shadow bytes around the buggy address:
  0x0c0c7fff80d0: 00 00 00 00 00 00 06 fa fa fa fa fa 00 00 00 00
  0x0c0c7fff80e0: 00 00 04 fa fa fa fa fa 00 00 00 00 00 00 00 fa
  0x0c0c7fff80f0: fa fa fa fa fd fd fd fd fd fd fd fa fa fa fa fa
  0x0c0c7fff8100: fd fd fd fd fd fd fd fa fa fa fa fa 00 00 00 00
  0x0c0c7fff8110: 00 00 00 00 fa fa fa fa 00 00 00 00 00 00 00 00
=>0x0c0c7fff8120: fa fa fa fa 00 00 00 00 00 00 00[01]fa fa fa fa
  0x0c0c7fff8130: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c0c7fff8140: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c0c7fff8150: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c0c7fff8160: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c0c7fff8170: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
```

```
Shadow byte legend (one shadow byte represents 8 application bytes):
  Addressable:           00
  Partially addressable: 01 02 03 04 05 06 07
  Heap left redzone:       fa
  Freed heap region:       fd
  Stack left redzone:      f1
  Stack mid redzone:       f2
  Stack right redzone:     f3
  Stack after return:      f5
  Stack use after scope:   f8
  Global redzone:          f9
  Global init order:       f6
  Poisoned by user:        f7
  Container overflow:      fc
  Array cookie:            ac
  Intra object redzone:    bb
  ASan internal:           fe
  Left alloca redzone:     ca
  Right alloca redzone:    cb
  Shadow gap:              cc
==109209==ABORTING
```

7.
Conclusiones

Trabajo realizado y trabajo futuro

Conclusiones

- Trabajo realizado
  - Análisis estático y dinámico → Degradación del rendimiento y mayor tiempo de compilación durante el **desarrollo**
  - Fuzzing → Puede no arrojar resultados válidos tras horas de uso
  - Binarios → Casi todos actualizados a 2022
- Trabajo futuro
  - Análisis estático → Analizar todos los informes generados
  - Realizar análisis a todos los paquetes de los repositorios
  - Uso de diferente software de fuzzing

¡Gracias!

¿Alguna pregunta?