Report for beleaf at 2024-06-16

Summary

Enumeration

Binary Information

File Name	Path	Format	Bit
beleaf	app/testFile/beleaf	ELF	64-bit

Security of the Binary

Basic Security Features			
Linked	**Stripped**	**RELRO**	**Canary**
dynamically linked	yes	full	yes

Advanced Security Mechanisms		
NX	**PIE**	**RPath**
yes	yes	no

Security Meta-Information		
RunPath	**Symbols**	**Fortify Source**
no	no	no

Strings

- Enter the flag
- .note.gnu.build-id

Vulnerable Functions

- scanf
- printf
- strlen

Libraries

- linux-vdso.so.1
- libc.so.6
- /lib64/ld-linux-x86-64.so.2

Assembly Code

```
xor ebp, ebp
mov r9, rdx
pop rsi
mov rdx, rsp
and rsp, 0xfffffffffffffff0
push rax
push rsp
lea r8, [rip + 0x34a]
lea rcx, [rip + 0x2d3]
lea rdi, [rip + 0x18d]
call qword ptr [rip + 0x2008c6]
hlt
nop dword ptr [rax + rax]
lea rdi, [rip + 0x200ec1]
push rbp
lea rax, [rip + 0x200eb9]
cmp rax, rdi
mov rbp, rsp
je 0x750
mov rax, qword ptr [rip + 0x20089a]
test rax, rax
je 0x750
pop rbp
jmp rax
nop word ptr cs:[rax + rax]
pop rbp
ret
nop dword ptr [rax]
nop word ptr cs:[rax + rax]
```

```
lea rdi, [rip + 0x200e81]
lea rsi, [rip + 0x200e7a]
push rbp
sub rsi, rdi
mov rbp, rsp
sar rsi, 3
mov rax, rsi
shr rax, 0x3f
add rsi, rax
sar rsi, 1
je 0x7a0
mov rax, qword ptr [rip + 0x200861]
test rax, rax
je 0x7a0
pop rbp
jmp rax
nop word ptr [rax + rax]
pop rbp
ret
nop dword ptr [rax]
nop word ptr cs:[rax + rax]
cmp byte ptr [rip + 0x200e31], 0
jne 0x7e8
cmp qword ptr [rip + 0x200837], 0
push rbp
mov rbp, rsp
je 0x7d3
mov rdi, qword ptr [rip + 0x20083a]
call 0x6e0
call 0x720
mov byte ptr [rip + 0x200e09], 1
pop rbp
ret
nop dword ptr [rax]
repz ret
nop word ptr [rax + rax]
push rbp
```

```
mov rbp, rsp
pop rbp
jmp 0x760
push rbp
mov rbp, rsp
mov eax, edi
mov byte ptr [rbp - 0x14], al
mov qword ptr [rbp - 8], 0
jmp 0x890
movsx edx, byte ptr [rbp - 0x14]
mov rax, qword ptr [rbp - 8]
lea rcx, [rax*4]
lea rax, [rip + 0x2007f9]
mov eax, dword ptr [rcx + rax]
cmp edx, eax
jne 0x834
mov rax, qword ptr [rbp - 8]
jmp 0x89f
movsx edx, byte ptr [rbp - 0x14]
mov rax, qword ptr [rbp - 8]
lea rcx, [rax*4]
lea rax, [rip + 0x2007d5]
mov eax, dword ptr [rcx + rax]
cmp edx, eax
jge 0x863
mov rax, qword ptr [rbp - 8]
add rax, rax
add rax, 1
mov qword ptr [rbp - 8], rax
jmp 0x890
movsx edx, byte ptr [rbp - 0x14]
mov rax, qword ptr [rbp - 8]
lea rcx, [rax*4]
lea rax, [rip + 0x2007a6]
mov eax, dword ptr [rcx + rax]
cmp edx, eax
jle 0x890
```

```
mov rax, qword ptr [rbp - 8]
add rax, 1
add rax, rax
mov qword ptr [rbp - 8], rax
cmp qword ptr [rbp - 8], -1
jne 0x810
mov rax, qword ptr [rbp - 8]
pop rbp
ret
push rbp
mov rbp, rsp
sub rsp, 0xc0
mov dword ptr [rbp - 0xb4], edi
mov qword ptr [rbp - 0xc0], rsi
mov rax, qword ptr fs:[0x28]
mov qword ptr [rbp - 8], rax
xor eax, eax
lea rdi, [rip + 0x195]
mov eax, 0
call 0x6b0
lea rax, [rbp - 0x90]
mov rsi, rax
lea rdi, [rip + 0x18e]
mov eax, 0
call 0x6c0
lea rax, [rbp - 0x90]
mov rdi, rax
call 0x690
mov qword ptr [rbp - 0xa0], rax
cmp qword ptr [rbp - 0xa0], 0x20
ja 0x92a
lea rdi, [rip + 0x160]
call 0x680
mov edi, 1
call 0x6d0
mov qword ptr [rbp - 0xa8], 0
jmp 0x99d
```

```
lea rdx, [rbp - 0x90]
mov rax, qword ptr [rbp - 0xa8]
add rax, rdx
movzx eax, byte ptr [rax]
movsx eax, al
mov edi, eax
call 0x7fa
mov qword ptr [rbp - 0x98], rax
mov rax, qword ptr [rbp - 0xa8]
lea rdx, [rax*8]
lea rax, [rip + 0x200b6e]
mov rax, qword ptr [rdx + rax]
cmp qword ptr [rbp - 0x98], rax
je 0x995
lea rdi, [rip + 0xf5]
call 0x680
mov edi, 1
call 0x6d0
add qword ptr [rbp - 0xa8], 1
mov rax, qword ptr [rbp - 0xa8]
cmp rax, qword ptr [rbp - 0xa0]
jb 0x937
lea rdi, [rip + 0xd2]
call 0x680
mov eax, 0
mov rcx, qword ptr [rbp - 8]
xor rcx, qword ptr fs:[0x28]
je 0x9d2
call 0x6a0
leave
ret
nop word ptr cs:[rax + rax]
nop
push r15
push r14
mov r15, rdx
push r13
```

```
push r12
lea r12, [rip + 0x20039e]
push rbp
lea rbp, [rip + 0x20039e]
push rbx
mov r13d, edi
mov r14, rsi
sub rbp, r12
sub rsp, 8
sar rbp, 3
call 0x650
test rbp, rbp
je 0xa36
xor ebx, ebx
nop dword ptr [rax + rax]
mov rdx, r15
mov rsi, r14
mov edi, r13d
call qword ptr [r12 + rbx*8]
add rbx, 1
cmp rbp, rbx
jne 0xa20
add rsp, 8
pop rbx
pop rbp
pop r12
pop r13
pop r14
pop r15
ret
nop
nop word ptr cs:[rax + rax]
repz ret
```

Code Analysis

Pseudo C Code

_ITM_deregisterTMCloneTable.c

```
/* WARNING: Control flow encountered bad instruction data
*/

void _ITM_deregisterTMCloneTable(void)

{
                    /* WARNING: Bad instruction -
Truncating control flow here */
  halt_baddata();
}
```

FUN_001007fa.c

```
long FUN_001007fa(char param_1)

{
  long local_10;

  local_10 = 0;
  while ((local_10 != -1 && ((int)param_1 != *(int *)
(&DAT_00301020 + local_10 * 4)))) {
    if ((int)param_1 < *(int *)(&DAT_00301020 + local_10 *
4)) {
      local_10 = local_10 * 2 + 1;
    }
```

```
    else if (*(int *)(&DAT_00301020 + local_10 * 4) <
(int)param_1) {
      local_10 = (local_10 + 1) * 2;
    }
  }
  return local_10;
}
```

entry.c

```
void processEntry entry(undefined8 param_1,undefined8
param_2)

{
  undefined auStack_8 [8];

  (*(code *)PTR___libc_start_main_00300fe0)

(FUN_001008a1,param_2,&stack0x00000008,FUN_001009e0,FUN_00100a50,param_
  do {
                    /* WARNING: Do nothing block with
infinite loop */
  } while( true );
}
```

FUN_001008a1.c

```
undefined8 FUN_001008a1(void)
```

```
{
  size_t sVar1;
  long lVar2;
  long in_FS_OFFSET;
  ulong local_b0;
  char local_98 [136];
  long local_10;

  local_10 = *(long *)(in_FS_OFFSET + 0x28);
  printf("Enter the flag\n>>> ");
  __isoc99_scanf(&DAT_00100a78,local_98);
  sVar1 = strlen(local_98);
  if (sVar1 < 0x21) {
    puts("Incorrect!");
                    /* WARNING: Subroutine does not return
*/
    exit(1);
  }
  for (local_b0 = 0; local_b0 < sVar1; local_b0 = local_b0
+ 1) {
    lVar2 = FUN_001007fa((int)local_98[local_b0]);
    if (lVar2 != *(long *)(&DAT_003014e0 + local_b0 * 8))
{
      puts("Incorrect!");
                    /* WARNING: Subroutine does not return
*/
      exit(1);
    }
  }
  puts("Correct!");
  if (local_10 != *(long *)(in_FS_OFFSET + 0x28)) {
                    /* WARNING: Subroutine does not return
*/
    __stack_chk_fail();
  }
  return 0;
}
```

_ITM_registerTMCloneTable.c

```
/* WARNING: Control flow encountered bad instruction data
*/

void _ITM_registerTMCloneTable(void)

{
                    /* WARNING: Bad instruction -
Truncating control flow here */
  halt_baddata();
}
```

ChatGPT Analysis

Exploit

Fuzzing

Exploit success with these input :

- Enter the flag
- .note.gnu.build-id

Buffer Overflow

Format String

Credits

This report was generated using automated tools and the expert analysis of security researchers.