Lazy binding(Feat. Now binding)

List

Lazy binding

Explanation

Lazy binding

- Lazy Binding은 lazy linking 또는 on-demand symbol resolution이라고도 합니다.
- Lazy Binding은 심볼이 실제로 사용될 때까지 심볼의 라이브러리 파일의 주소 값 확인이 수행되지 않습니다..

Lazy binding behavior flow

- 모든 동적라이브러리 함수는 PLT(Procedure Linkage Table) stub코드를 통해 호출됩니다.
  - PLT의 stub 코드는 상대 주소를 이용하여, 사용할 GOT(Global Offset Table)의 주소 값을 검색합니다.
  - PLT는 GOT의 위치를 알고 있으며, 해당 GOT에 저장된 대상 함수의 주소를 읽고 이동합니다.
- 이와 같은 동작이 실행되기 위해서 GOT에 적절한 주소가 채워 져야합니다.
  - Lazy Binding은 함수 호출이 처음 요청될 때 stub 코드를 이용해 GOT영역에 해당 함수의 심볼 주소를 생성합니다.
  - Stub 코드는 런타임 링커가 제공하는 바인딩 함수에 필요한 정보를 설정하는 역할을 하며, 설정이 완료되면 스텁 코드는 바인딩 함수로 이동합니다.
- 바인딩 함수는 "_dl_runtime_resolve" 함수에 대한 인수를 설정 후에 해당 함수를 호출합니다.
  - 그리고 "_dl_runtime_resolve" 함수에서 반환 된 주소로 이동합니다.
- 다음부터는 유저가 함수를 호출 할 때 PLT stub code는 호출한 함수로 바로 이동합니다.
  - GOT 영역에 해당 함수의 동적 라이브러리 주소 값이 저장되어 있기 때문입니다.
- 초기 GOT 영역에는 특수 Stub code의 주소가 저장되어 있고, 런타임 링커는 로드 기반에 대한 단순한 재배치만 수행합니다.
  - 런타임 링크는 부하 베이스에 대한 간단한 재배치만 수행합니다.

	Lazy binding	Now binding
Build option	-Wl,-zlazy	-Wl,-znow
Description	• 프로그램 실행 후 함수가 호출 될 때 심볼을 찾도록 설정합니다.	• 프로그램이 실행 될 때 dlopen()을 사용하여 공유 라이브러리가 링크 될 때 동적 링커에 모든 심볼을 찾게됩니다.

Example

Source code

RELRO.c

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

void main(){

        char address[16];
        size_t *pointer;
        int count = 1;

        while(count != 100)
        {
                printf("----- %d -----\n",count);
                memset(address,0,16);
                printf("Input Pointer address : ");
                fgets(address,16,stdin);

                pointer = strtol(address,0,16);
                printf("Pointer address : %p\n",pointer);

                printf("Input Pointer text : ");
                fgets(pointer,16,stdin);
                printf("Pointer text : %s\n",pointer);
                count++;
        }
        scanf("%s",address);
}
```

Lazy binding

- **다음과 같이 심볼 주소가 저장됩니다.**
  - main 함수는 printf 함수를 호출하기 위해 0x4005b0 주소를 호출합니다.
  - 0x4005b0 영역의 코드는 0x601020 영역에 저장된 주소로 이동합니다.
  - 0x601020 영역에는 0x4005b6 주소 값이 저장되어 있습니다.
    - printf 함수가 한번도 호출 되지 않았기 때문에 stub 코드 영역의 주소가 저장되어 있습니다.
  - 0x4005b6 영역에서는 개별 함수 번호를 스택에 저장하고, 0x400590 영역으로 이동합니다.
  - 0x400590 영역에서는 프로그램 파일의 정보를 스택에 저장하고, 0x601010 주소에 저장된 영역으로 이동합니다.
  - 0x601010 영역은 _dl_runtime_resolve() 함수의 코드 영역입니다.
  - _dl_runtime_resolve() 함수는 _dl_fixup() 호출을 통해 실제 함수 주소를 got 영역에 저장하고 리턴합니다.
- **해당 작업으로 인해 printf 함수만 got 영역에 실제 주소가 저장됩니다.**
  - 아직 호출되지 않은 함수들은 got 영역에 실제 주소가 저장되어 있지 않습니다.

Process of Lazy binding

```
lazenca0x0@ubuntu:~/Documents/Definition/protection/RELRO$ gdb -q ./RELRO-Relro
Reading symbols from ./RELRO-Relro...(no debugging symbols found)...done.

gdb-peda$ b *0x0000000000400748
Breakpoint 1 at 0x400748
gdb-peda$ r
Starting program: /home/lazenca0x0/Documents/Definition/protection/RELRO/RELRO-Relro
Breakpoint 1, 0x0000000000400748 in main ()
gdb-peda$ x/i 0x4005b0
   0x4005b0 <printf@plt>:        jmp    QWORD PTR [rip+0x200a6a]        # 0x601020
gdb-peda$ x/gx 0x601020
0x601020:        0x00000000004005b6
gdb-peda$ x/2i 0x00000000004005b6
   0x4005b6 <printf@plt+6>:        push   0x1
   0x4005bb <printf@plt+11>:        jmp    0x400590
gdb-peda$ x/4i 0x400590
   0x400590:        push   QWORD PTR [rip+0x200a72]        # 0x601008
   0x400596:        jmp    QWORD PTR [rip+0x200a74]        # 0x601010
   0x40059c:        nop    DWORD PTR [rax+0x0]
   0x4005a0 <__stack_chk_fail@plt>:        jmp    QWORD PTR [rip+0x200a72]        # 0x601018
gdb-peda$ x/gx 0x601010
0x601010:        0x00007ffff7dee870
```

```
gdb-peda$ x/51i 0x00007ffff7dee870
   0x7ffff7dee870 <_dl_runtime_resolve_avx>:        push   rbx
   0x7ffff7dee871 <_dl_runtime_resolve_avx+1>:        mov    rbx,rsp
   0x7ffff7dee874 <_dl_runtime_resolve_avx+4>:        and    rsp,0xfffffffffffffe0
   0x7ffff7dee878 <_dl_runtime_resolve_avx+8>:        sub    rsp,0x180
   0x7ffff7dee87f <_dl_runtime_resolve_avx+15>:        mov    QWORD PTR [rsp+0x140],rax
   0x7ffff7dee887 <_dl_runtime_resolve_avx+23>:        mov    QWORD PTR [rsp+0x148],rcx
   0x7ffff7dee88f <_dl_runtime_resolve_avx+31>:        mov    QWORD PTR [rsp+0x150],rdx
   0x7ffff7dee897 <_dl_runtime_resolve_avx+39>:        mov    QWORD PTR [rsp+0x158],rsi
   0x7ffff7dee89f <_dl_runtime_resolve_avx+47>:        mov    QWORD PTR [rsp+0x160],rdi
   0x7ffff7dee8a7 <_dl_runtime_resolve_avx+55>:        mov    QWORD PTR [rsp+0x168],r8
   0x7ffff7dee8af <_dl_runtime_resolve_avx+63>:        mov    QWORD PTR [rsp+0x170],r9
   0x7ffff7dee8b7 <_dl_runtime_resolve_avx+71>:        vmovdqa YMMWORD PTR [rsp],ymm0
   0x7ffff7dee8bc <_dl_runtime_resolve_avx+76>:        vmovdqa YMMWORD PTR [rsp+0x20],ymm1
   0x7ffff7dee8c2 <_dl_runtime_resolve_avx+82>:        vmovdqa YMMWORD PTR [rsp+0x40],ymm2
   0x7ffff7dee8c8 <_dl_runtime_resolve_avx+88>:        vmovdqa YMMWORD PTR [rsp+0x60],ymm3
   0x7ffff7dee8ce <_dl_runtime_resolve_avx+94>:        vmovdqa YMMWORD PTR [rsp+0x80],ymm4
   0x7ffff7dee8d7 <_dl_runtime_resolve_avx+103>:        vmovdqa YMMWORD PTR [rsp+0xa0],ymm5
   0x7ffff7dee8e0 <_dl_runtime_resolve_avx+112>:        vmovdqa YMMWORD PTR [rsp+0xc0],ymm6
   0x7ffff7dee8e9 <_dl_runtime_resolve_avx+121>:        vmovdqa YMMWORD PTR [rsp+0xe0],ymm7
   0x7ffff7dee8f2 <_dl_runtime_resolve_avx+130>:        bndmov [rsp+0x100],bnd0
   0x7ffff7dee8fb <_dl_runtime_resolve_avx+139>:        bndmov [rsp+0x110],bnd1
   0x7ffff7dee904 <_dl_runtime_resolve_avx+148>:        bndmov [rsp+0x120],bnd2
   0x7ffff7dee90d <_dl_runtime_resolve_avx+157>:        bndmov [rsp+0x130],bnd3
   0x7ffff7dee916 <_dl_runtime_resolve_avx+166>:        mov    rsi,QWORD PTR [rbx+0x10]
   0x7ffff7dee91a <_dl_runtime_resolve_avx+170>:        mov    rdi,QWORD PTR [rbx+0x8]
   0x7ffff7dee91e <_dl_runtime_resolve_avx+174>:        call   0x7ffff7de69f0 <_dl_fixup>
   0x7ffff7dee923 <_dl_runtime_resolve_avx+179>:        mov    r11,rax
   0x7ffff7dee926 <_dl_runtime_resolve_avx+182>:        bndmov bnd3,[rsp+0x130]
   0x7ffff7dee92f <_dl_runtime_resolve_avx+191>:        bndmov bnd2,[rsp+0x120]
   0x7ffff7dee938 <_dl_runtime_resolve_avx+200>:        bndmov bnd1,[rsp+0x110]
   0x7ffff7dee941 <_dl_runtime_resolve_avx+209>:        bndmov bnd0,[rsp+0x100]
   0x7ffff7dee94a <_dl_runtime_resolve_avx+218>:        mov    r9,QWORD PTR [rsp+0x170]
   0x7ffff7dee952 <_dl_runtime_resolve_avx+226>:        mov    r8,QWORD PTR [rsp+0x168]
   0x7ffff7dee95a <_dl_runtime_resolve_avx+234>:        mov    rdi,QWORD PTR [rsp+0x160]
   0x7ffff7dee962 <_dl_runtime_resolve_avx+242>:        mov    rsi,QWORD PTR [rsp+0x158]
   0x7ffff7dee96a <_dl_runtime_resolve_avx+250>:        mov    rdx,QWORD PTR [rsp+0x150]
   0x7ffff7dee972 <_dl_runtime_resolve_avx+258>:        mov    rcx,QWORD PTR [rsp+0x148]
   0x7ffff7dee97a <_dl_runtime_resolve_avx+266>:        mov    rax,QWORD PTR [rsp+0x140]
   0x7ffff7dee982 <_dl_runtime_resolve_avx+274>:        vmovdqa ymm0,YMMWORD PTR [rsp]
   0x7ffff7dee987 <_dl_runtime_resolve_avx+279>:        vmovdqa ymm1,YMMWORD PTR [rsp+0x20]
   0x7ffff7dee98d <_dl_runtime_resolve_avx+285>:        vmovdqa ymm2,YMMWORD PTR [rsp+0x40]
   0x7ffff7dee993 <_dl_runtime_resolve_avx+291>:        vmovdqa ymm3,YMMWORD PTR [rsp+0x60]
   0x7ffff7dee999 <_dl_runtime_resolve_avx+297>:        vmovdqa ymm4,YMMWORD PTR [rsp+0x80]
   0x7ffff7dee9a2 <_dl_runtime_resolve_avx+306>:        vmovdqa ymm5,YMMWORD PTR [rsp+0xa0]
   0x7ffff7dee9ab <_dl_runtime_resolve_avx+315>:        vmovdqa ymm6,YMMWORD PTR [rsp+0xc0]
   0x7ffff7dee9b4 <_dl_runtime_resolve_avx+324>:        vmovdqa ymm7,YMMWORD PTR [rsp+0xe0]
   0x7ffff7dee9bd <_dl_runtime_resolve_avx+333>:        mov    rsp,rbx
   0x7ffff7dee9c0 <_dl_runtime_resolve_avx+336>:        mov    rbx,QWORD PTR [rsp]
   0x7ffff7dee9c4 <_dl_runtime_resolve_avx+340>:        add    rsp,0x18
   0x7ffff7dee9c8 <_dl_runtime_resolve_avx+344>:        bnd jmp r11
   0x7ffff7dee9cc:        nop    DWORD PTR [rax+0x0]
gdb-peda$ b *0x7ffff7dee91e
Breakpoint 2 at 0x7ffff7dee91e: file ../sysdeps/x86_64/dl-trampoline.h, line 112.
gdb-peda$ c
Continuing.

Breakpoint 2, _dl_runtime_resolve_avx () at ../sysdeps/x86_64/dl-trampoline.h:112
112        ../sysdeps/x86_64/dl-trampoline.h: No such file or directory.
gdb-peda$ x/gx 0x601020
0x601020:        0x00000000004005b6
gdb-peda$ ni
gdb-peda$ x/gx 0x601020
0x601020:        0x00007ffff7a62800
gdb-peda$ i r rax
rax            0x7ffff7a62800        0x7ffff7a62800
gdb-peda$ x/i 0x7ffff7a62800
   0x7ffff7a62800 <__printf>:        sub    rsp,0xd8
gdb-peda$
```

Now binding

- **다음과 같이 심볼 주소가 저장됩니다.**
  - main함수는 printf 함수를 호출하기 위해 0x4005c8 주소를 호출합니다.
  - 0x4005c8 영역의 코드는 0x600fc8 영역에 저장된 주소로 이동합니다.
  - lazy binding과 달리 0x600fc8 영역에는 0x0 값이 저장되어 있습니다.
  - Now binding의 경우 프로그램이 실행될 때 해당 프로그램에서 사용되는 함수들의 주소를 읽어와 got영역에 저장합니다.
  - 해당 과정을 확인하지 위해 0x600fc8 영역을 watchpoint로 설정하고 프로그램을 실행하면 해당 영역의 값이 실제주소값으로 변경되는 것을 확인할 수 있습니다.

Process of Now binding

```
lazenca0x0@ubuntu:~/Documents/Definition/protection/RELRO$ gdb -q RELRO-FullRelro
Reading symbols from RELRO-FullRelro...(no debugging symbols found)...done.
gdb-peda$ x/i 0x4005c8
   0x4005c8:        jmp    QWORD PTR [rip+0x2009fa]        # 0x600fc8
gdb-peda$ x/gx 0x600fc8
0x600fc8:        0x0000000000000000
gdb-peda$ watch *0x600fc8
Hardware watchpoint 1: *0x600fc8
gdb-peda$ r
Starting program: /home/lazenca0x0/Documents/Definition/protection/RELRO/RELRO-FullRelro
Hardware watchpoint 1: *0x600fc8

Old value = 0x0
New value = 0xf7a62800
0x00007ffff7de388f in elf_machine_rela (skip_ifunc=<optimized out>, reloc_addr_arg=<optimized out>,
version=<optimized out>, sym=<optimized out>, reloc=<optimized out>, map=<optimized out>)
    at ../sysdeps/x86_64/dl-machine.h:435
435        ../sysdeps/x86_64/dl-machine.h: No such file or directory.
gdb-peda$
```

Related information

- N/a