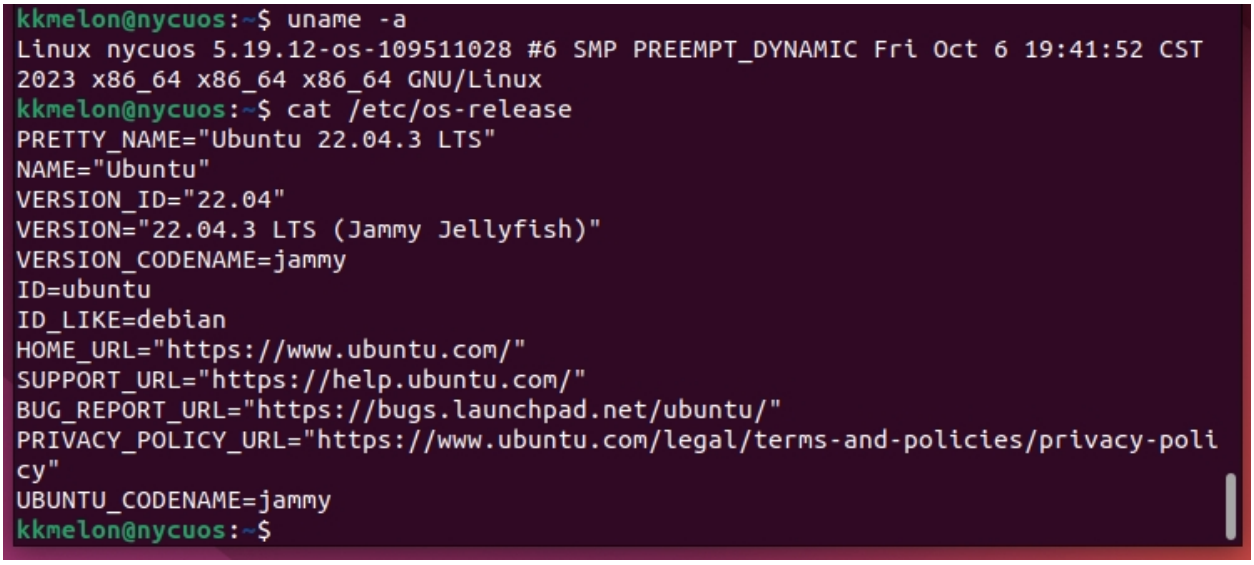
OS HW1

- Notion 排版有點麻煩，還請助教體諒QAQ

Kernel Compilation



System Call

Common

- `linux-5.19.12/include/linux/syscalls.h`
  - This file contain the declarations for system calls.
  - asmlinkage tells the compiler that the essential data are stored in CPU stacks now

```
asmlinkage long sys_hello(void);
asmlinkage long sys_revstr(int len, char __user *str);
```

- `linux-5.19.12/include/uapi/asm-generic/unistd.h`
  - Define the id of newly-created System Calls.

- __NR_syscalls is the dummy syscall that is used to represent the number of syscalls, so it is always the last syscall and the number should be #(syscalls)

```
...
#define __NR_hello 451
__SYSCALL(__NR_hello, sys_hello)
#define __NR_revstr 452
__SYSCALL(__NR_revstr, sys_revstr)
...
#undef __NR_syscalls
#define __NR_syscalls 453
...
```

- `arch/x86/entry/syscalls/syscall_64.tbl`
  - Generate the system call table for compiling kernel code.

```
...
451 common hello sys_hello
452 common revstr sys_revstr
...
```

sys_hello

```c
/*
 * sys_hello - Say "Hello World" and <student-id> to User.
 */
int ksys_hello(void)
{
    printk(KERN_INFO "Hello World\n");
    printk(KERN_INFO "109511028\n");
    return 0;
}
SYSCALL_DEFINE0(hello)
{
    return ksys_hello();
}
```

sys_revstr

```c
int ksys_revstr(int size, char __user *msg)
{
    char buf[256];
    long copied = strncpy_from_user(buf, msg, size);
    if(copied < 0 || copied == size)
        return -EFAULT;
    char rev[256];
    for(int i = 0; i < size; i++){
        rev[size - 1 - i] = buf[i];
    }
    rev[size] = '\0';
    printk(KERN_INFO "The origin string: %s\n", buf);
    printk(KERN_INFO "The reversed string: %s\n", rev);
    return 0;
}
SYSCALL_DEFINE2(revstr, int, size, char __user *, msg)
{
    return ksys_revstr(size, msg);
}
```

`dmesg` Result

```
comm="snap-confine" capability=38  capname="perfmon"
[  554.555653] The origin string: hello
[  554.555658] The reversed string: olleh
[  554.555660] The origin string: 5Y573M C411
[  554.555660] The reversed string: 114C M375Y5
[  566.930447] Hello World
[  566.930451] 109511028
kkmelon@nycuos:~$
```

Reference

Linux Kernel: What does asmlinkage mean in the definition of system calls?

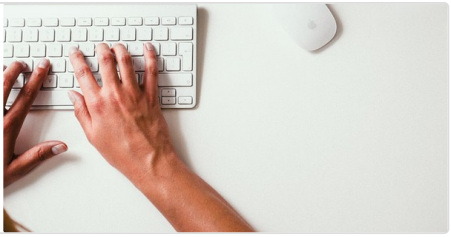
Answer (1 of 3): The short answer to your question is that asmlinkage tells your compiler to look on the CPU stack for the function parameters, instead of registers. In fact, it uses GCC's regparam attribute (http://gcc.gnu.org/onlinedocs/gcc/Function-

https://www.quora.com/Linux-Kernel-What-does-asmlinkage-mean-in-the-definition-of-system-calls

What are registers, and what is stack?

Answer: Every CPU has some number of internal registers. Registers are used by CPU instructions and could be considered as fastest memory CPU has. Some CPUs have internal registers as internal storage (eg ARM, x86)

https://www.quora.com/What-are-registers-and-what-is-stack



什麼是

jollen 發表於 October 26, 2006 4:15 PM

https://www.jollen.org/blog/2006/10/_asmlinkage.html

https://member.adl.tw/ernieshu/syscall_3_14_4.html