Implement system call of iOS

Implementing a system call on iOS is a complex task that typically requires advanced knowledge of kernel programming and is generally discouraged for standard application development.

Implementing a system call in iOS involves several steps and requires a deep understanding of the iOS kernel and the way system calls are structured in Unix-like operating systems. However, it's important to note that modifying the iOS kernel or adding new system calls typically requires jailbreaking the device, which voids warranties and can lead to security vulnerabilities.

Kernel space implementation:

```
#include <sys/syscall.h>

#include <sys/kern_syscall.h>

#define SYS_my_custom_syscall 300 // Example syscall number

// Function implementation

int my_custom_syscall(int arg1, const char *arg2) {

   // Perform some operations

   return 0; // Return success

}

// Add syscall to syscall table (this is pseudo-code)

void initialize_syscalls() {

   syscall_table[SYS_my_custom_syscall] = my_custom_syscall;

}
```

Use space implementation:

```
#include <unistd.h>

#include <sys/syscall.h>

#define SYS_my_custom_syscall 300 // Match with kernel definition

int my_custom_syscall(int arg1, const char *arg2) {

   return syscall(SYS_my_custom_syscall, arg1, arg2);
```

```
}

// Usage example

int main() {

    int result = my_custom_syscall(42, "Hello");

    return 0;

}
```