**발표를 시작하겠습니다.**

저는 사용자와 커널 간의 데이터 교환에 대해서 현재 프로젝트에서 사용하고 있는 Copy\_To\_User와 Copy\_From\_User를 통해서 설명드리겠습니다.

먼저 Copy\_From\_User () 및 Copy\_To\_User () 함수의 기능을 설명드리면 사용자 주소 공간과 커널 주소 공간 간에 데이터를 복사하는 것입니다. User memory에서 Kernel buffer로 데이터를 복사하는 것이 Copy\_From\_User 함수고 반대로 Kernel memory에서 User buffer로 데이터를 복사 하는 것이 Copy\_To\_User 함수입니다.

Userseg.c 파일 내에 있는 해당 함수들을 보게 되면 이런 구조를 가지고 있는 것을 확인할 수 있습니다.

먼저 Copy\_From\_User를 보게 되면 Copy\_From\_User 함수에서 파라미터로 받는 destInKernel은 커널 버퍼의 주소, srcInUser는 유저 버퍼의 주소를 뜻하고 bufsize는 복사할 바이트의 수를 뜻합니다. 하나하나 살펴보면 먼저 현재 process의 UserContext를 가져온 뒤 User buffer가 프로세스가 속한 메모리에 위치 하는지 검증하는 과정이 필요합니다.

이때 사용하는 private function이 있는데 바로 validate\_user\_memory 함수입니다. Private 하기 때문에 같은 파일 내에 위치하고 있고 이 함수를 살펴보게 되면 현재의 userContext, 현재 유저 버퍼의 주소, 복사할 바이트의 수와 for\_writing이라는 파라미터를 받고 있습니다. 전달 받은 파라미터를 바탕으로 검증작업을 거치게 되는데 먼저 userContext의 크기가 현재 유저 버퍼의 주소보다 크다면 false를 리턴해서 해당 검증이 유효하지 않음을 알려줍니다. 그리고 사C용할 수 있는 바이트의 수를 계산해 avail 변수에 할당하고 전달 받은 복사할 바이트의 수가 avail보다 크다면 그것 또한 false를 리턴해 준다. 그런 다음 만약 해당 검증 과정을 다 거친 뒤 문제가 없다면 최종적으로 true를 리턴해 주게 되고 memory에 복사가 가능하게 됩니다.

(다시 뒤로 넘어와서) 이제 검증을 완료했으니 memory를 복사해주어야 하는데 Copy\_from\_User 함수는 User 버퍼에서 Kernel로 데이터를 복사하는 것이기 때문에 memcpy를 사용할 때 복사받을 메모리를 가리키고 있는 포인터인 KernelBuffer의 주소 destInKernel을 첫 번째 파라미터로 넣어줍니다. 그리고 두 번째 파라미터로 복사할 데이터를 가리키는 포인터를 넣어줘야 하는데 여기서 User\_To\_Kernel 함수를 사용하고 있는 것을 보실 수 있습니다. 이거 또한 살펴보게 되면 이 함수는 단순히 User Buffer의 주소와 userContext의 memory를 더한 값. 다시 말해, 복사할 User Memory를 리턴해 주는 것을 확인할 수 있습니다. 이제 memcpy의 마지막 세 번 째 파라미터로 복사할 바이트의 길이 bufsize를 넣어주게 되면 메모리가 정상적으로 복사하게 되고 결과적으로 true를 리턴하면서 함수가 종료됩니다.

Copy\_To\_User 함수는 Copy\_From\_User와 반대로 데이터가 전달이 되는 것이기 때문에 거의 유사합니다. 간단하게 살펴보면 Copy\_From\_User 함수와 마찬가지로 세 가지의 파라미터로 받는데 destInUser는 유저 버퍼의 주소 srcInKernel은 커널 버퍼의 주소 bufSize는 복사할 바이트의 수를 뜻합니다. 이 함수는 Kernel Memory에서 User 버퍼로 메모리를 복사해주는 것이고 Copy\_From\_User와 같은 과정 검증을 거친 뒤 메모리 복사를 하면 true를 리턴하는 것으로 처리됩니다.

위 두 개의 함수를 사용하는 것을 간단하게 보여드리면 syscall.c 파일 안의 Copy\_User\_String 함수에서 확인할 수 있습니다. 함수 내에서 사용자가 입력한 문자를 위해 메모리를 할당하는 것을 볼 수 있습니다. 메모리가 0이면 Out of memory 에러를 발생시키고 그렇지 않다면 user space로부터 데이터를 복사해주게 되는데 여기 조건문을 보시게 되면 Copy\_From\_User가 False를 리턴할 때 Invalid argument 에러를 발생시키고 메모리를 해제해 주는 것을 확인하실 수 있습니다. Argument는 여기서 kernel buffer의 주소를 뜻하는 str 유저 버퍼의 주소를 뜻하는 uaddr 복사할 바이트의 수 다시 말해 사용자가 입력한 문자열의 길이를 뜻하는 len 이 됩니다. 이 Argument들 중에서 오류가 발생했기 때문에 Copy\_From\_User 함수가 False를 리턴하게 되면 결국 Invalid argument 에러가 발생하게 됩니다.