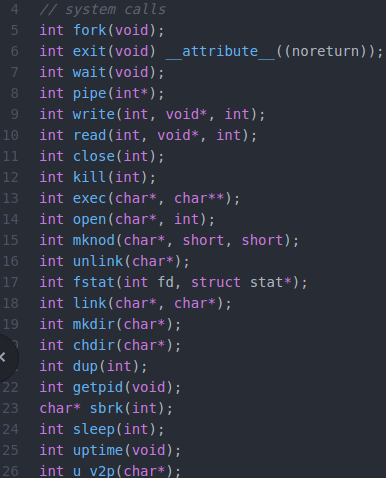


Figure 1: user.h u\_v2p def diff



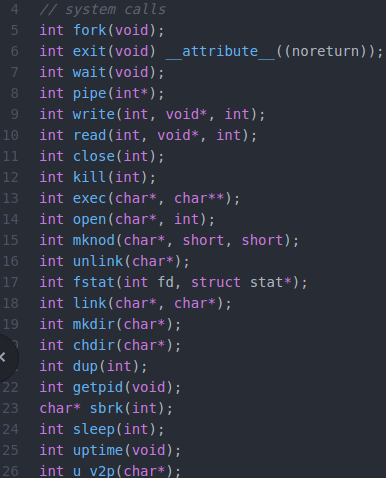


Figure 2: user.h u\_v2p def

Defining u\_v2p as a system call, callable by user.

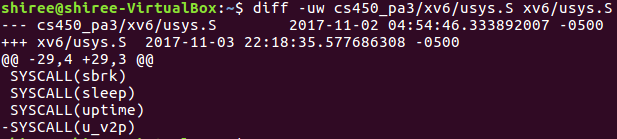
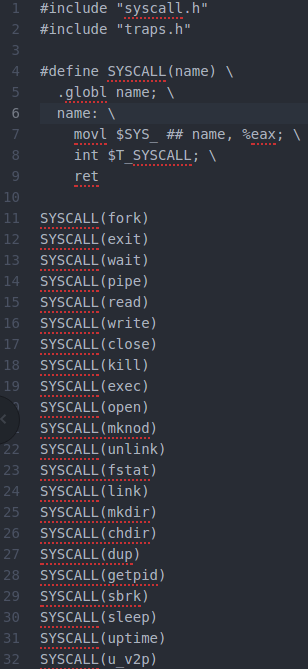


Figure 3: usys.S u\_v2p SYSCALL diff



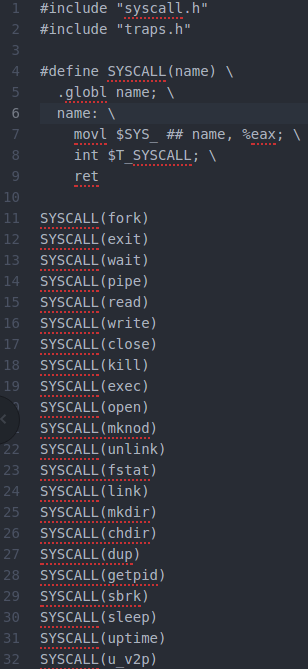


Figure 4: usys.S u\_v2p SYSCALL

Adding u\_v2p as a system call.

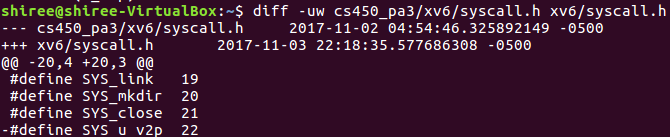
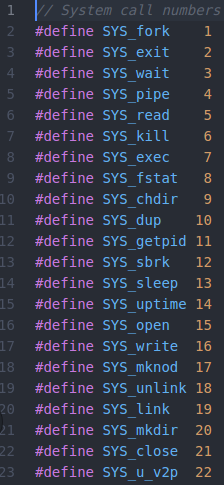


Figure 5: syscall.h SYS\_u\_v2p diff



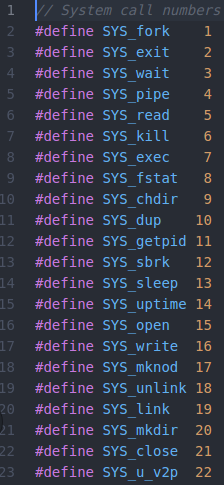


Figure 6: syscall.h SYS\_u\_v2p

Defining the system call number.

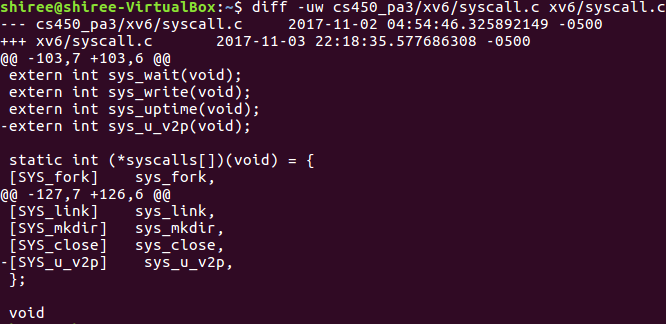
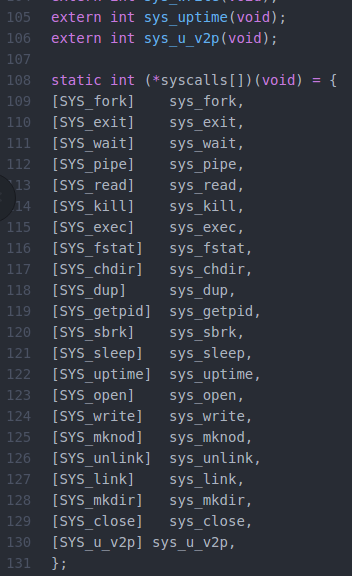


Figure 7: syscall.c sys\_u\_v2p diff



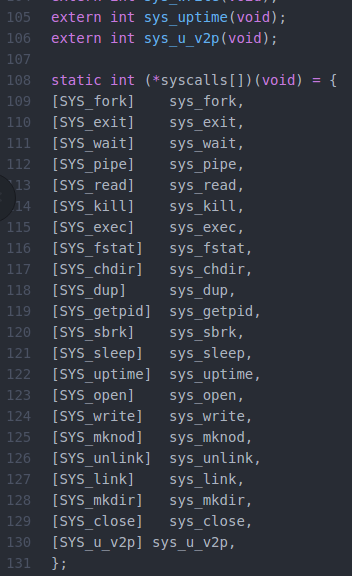


Figure 8: syscall.c sys\_u\_v2p

Adding the system call to the list and the associated function.

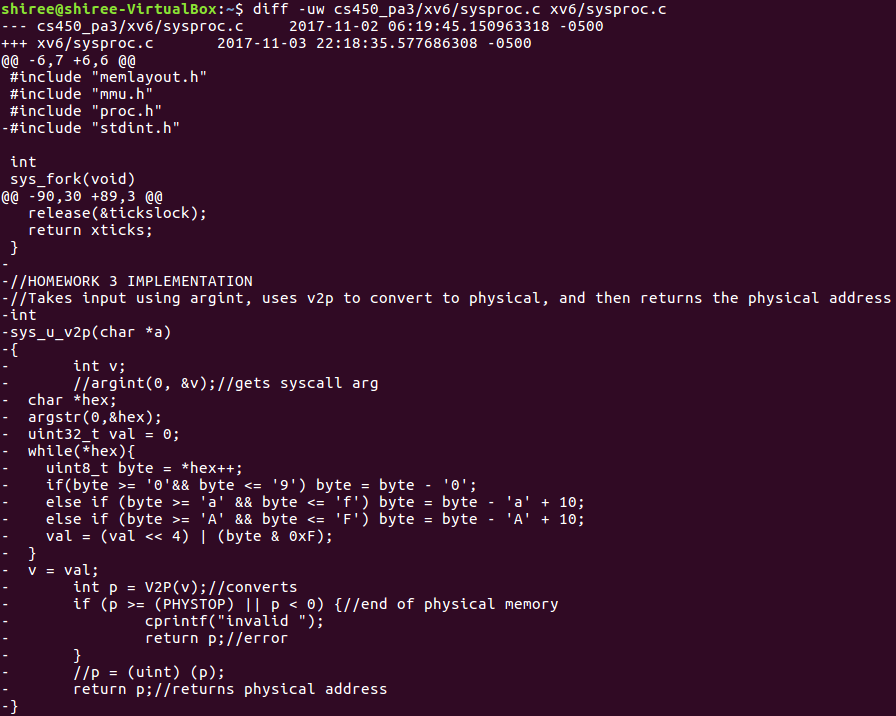


Figure 9: sysproc.c sys\_u\_v2p diff

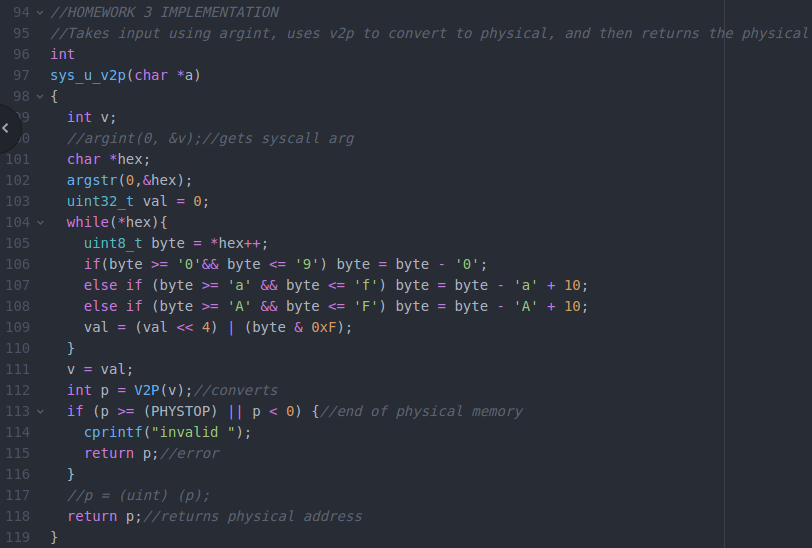


Figure 5: sysproc.c sys\_u\_v2p

Implementation of u\_v2p. Takes the syscall argument as a hex encoded string, converts into an integer and passes it as an argument to V2P. The returned value is checked between the bounds of 0 to PHYSTOP, in which if true states invalid. The returned value is the integer of the address.

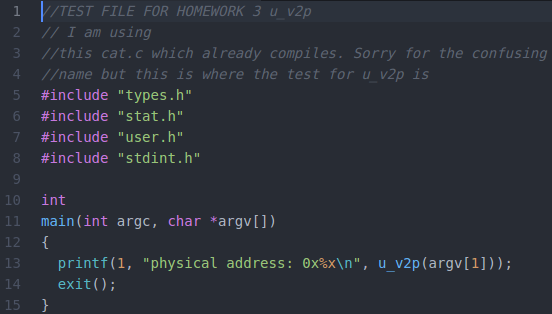


Figure 11: uv2p.c

Execution of u\_v2p, in which takes the argument and returns the hex translated address.