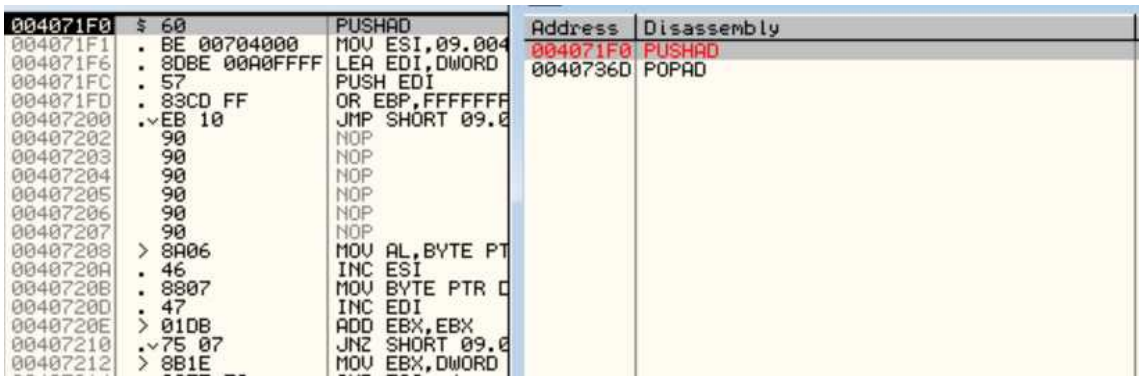
CodeEngn Basic RCE L09
: StolenByte를 구하시오 Ex) 75156A0068352040

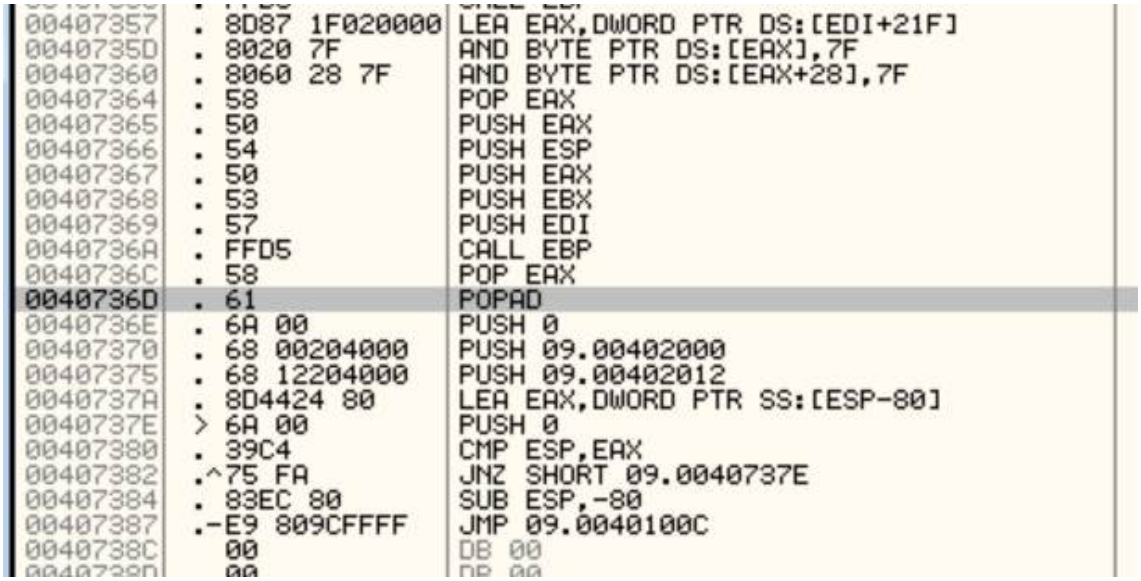
*STOLENBYTE란?
- 훔친 바이트란 의미로 프로그램의 한부분의 코드를 훔쳐내어 다른 부분으로 옮겨진 코드를 말합니다.
주로 옮겨지는 코드는 엔트리 포인트위의 몇개의 코드들이며 옮겨진 코드들은 OEP 주소로 점프하기 전에 위치에서 PUSH 됩니다.
이러한 StolenByte 는 주로 패커가 프로그램을 패킹할때 볼수있습니다. 이렇게 옮겨진 코드들은 할당된 메모리 공간에서 실행됩니다.
이때문에 패킹된 프로세스가 덤프될때 StolenByte를 복구하지 못하면 프로그램은 정상적으로 작동하지 못하게 됩니다.

출처: https://stih.tistory.com/67 [STIH]

엔트리 포인트 위 몇개의 코드들 & OEP 주소로 점프하기전 위치에서 PUSH 인 걸로 보아,
POPAD로 OEP코드를 찾고 그 위에 있는 코드들을 살펴보면 되겠다는 생각이 듭니다 !!



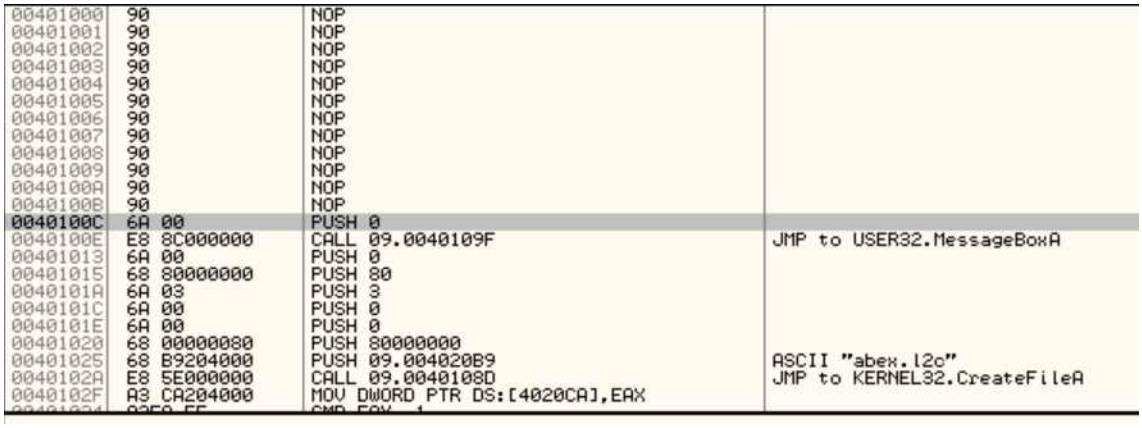
올리디버거로 틀면 PUSHAD가 나오고, 패킹되어 있으니깐 POPAD를 찾으면
하나의 POPAD가 존재하는 걸 볼 수 있습니다.



POPAD로 이동한 화면

0040737E ~ 00407382의 어셈블리어를 보면 JNZ 명령으로 계속 LOOP를 돌고 있는 거를 볼 수 있습니다.
그렇다면 OEP로 이동하는 주소는 루프를 빠져나와 점프하게 되는 주소인 0040100C인 것을 알 수 있어요 !

STOLENBYTE는 그 위에 PUSH되는 것들이라 했으니 PUSH와 OEP이동코드에 BP를 걸어 주었고, F9로 실행 시 ASCII코드가 나옵니다.



OEP 주소로 이동해서 자동으로 UNPACK을 해주었고, ASCII코드를 보면 EXE파일이 대충 어떤 문자열을 출력하는 지 볼 수 있어요!

OEP 시작 부분 바로 밑에 MessageBox가 있는데 이 함수는 파라미터가 4개인데, 위 쪽을 보면 PUSH 0 이외에 다 NOP으로 채워져 있습니다.

따라서 총 3개의 파라미터가 부족하고, 이전에 OEP 주소 위쪽에 있던 3개가 STOLEN BYTE 인것을 알 수 있답니다 ㅎㅎ

STOLEN BYTE는 6A006800204000068122040000 이네요 :)