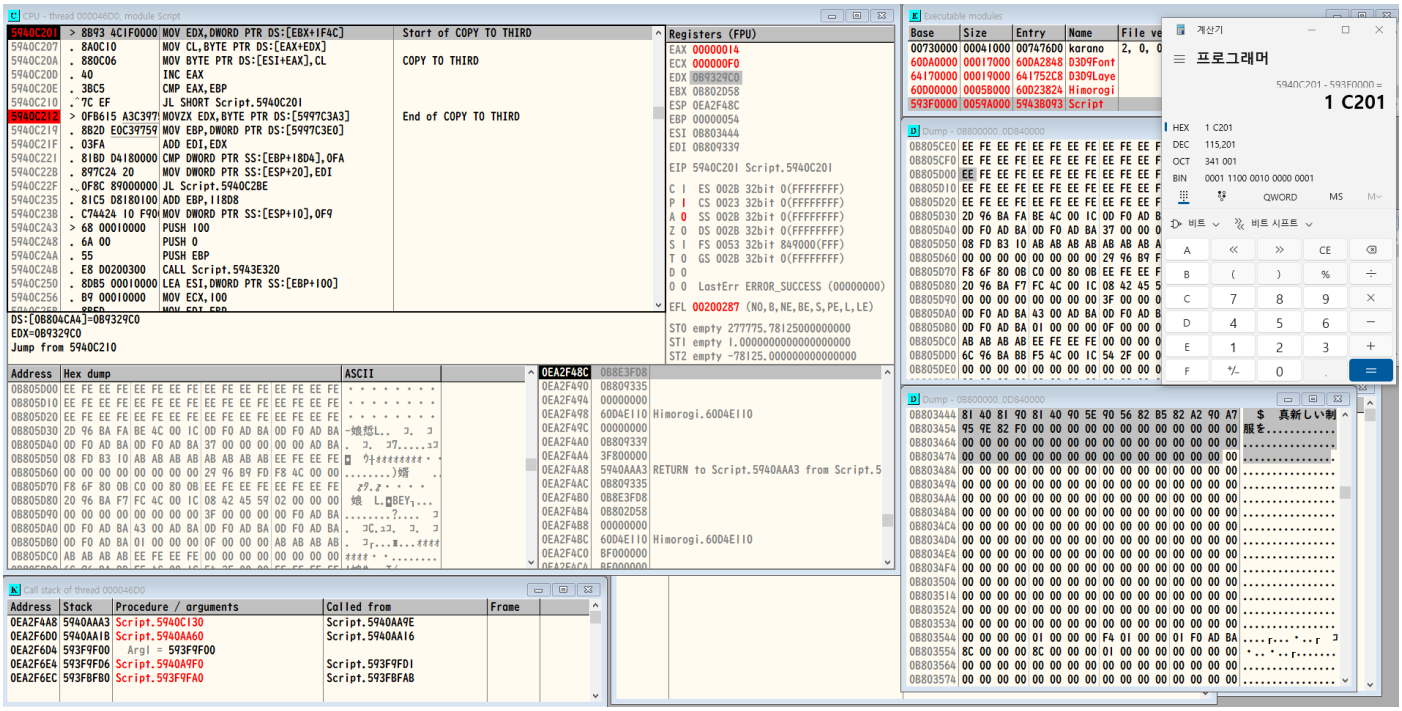
코드파인딩기록9

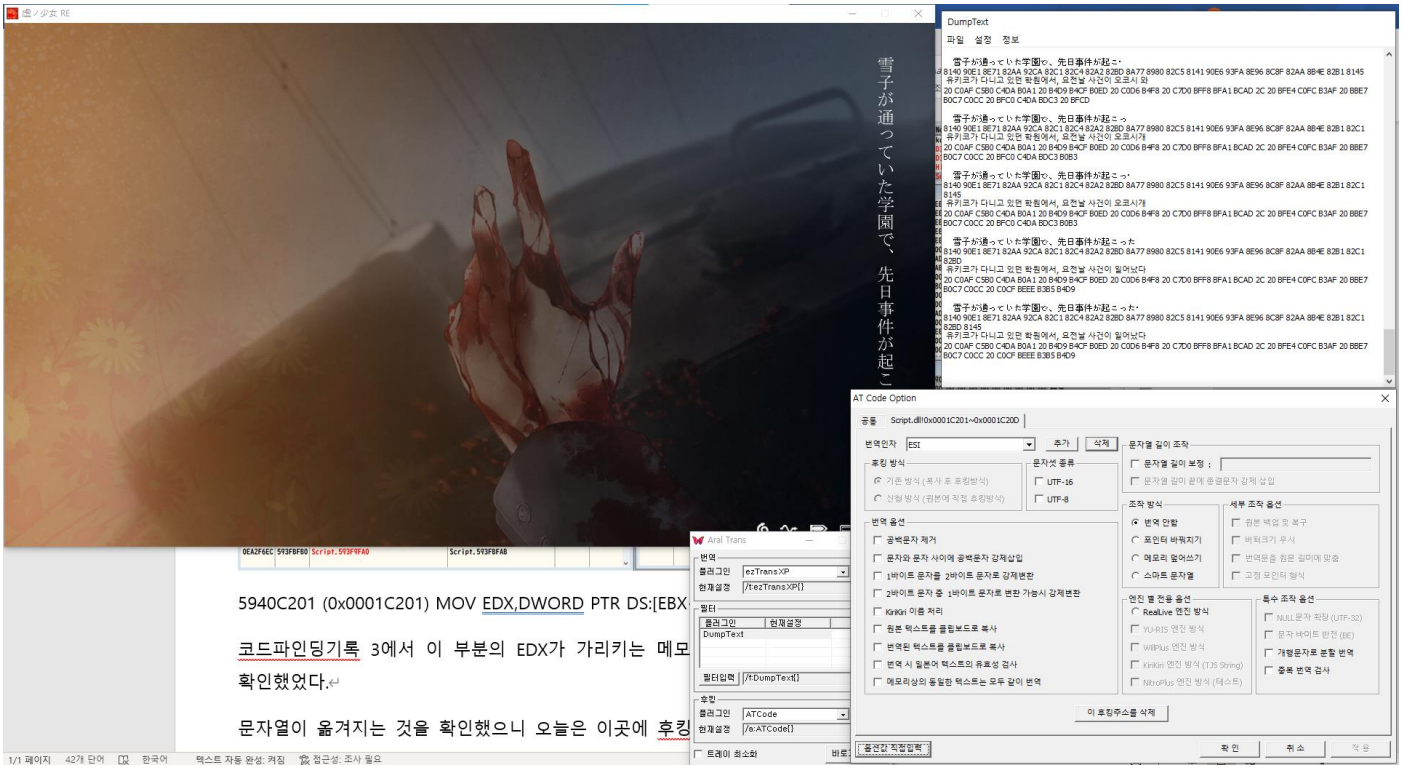


5940C201 (0x0001C201) MOV EDX,DWORD PTR DS:[EBX+1F4C]

코드파인딩기록 3에서 이 부분의 EDX가 가리키는 메모리의 값을 바꾸면, 화면에 출력되는 대사도 같이 바뀜을 확인했었다.

문자열이 옮겨지는 것을 확인했으니 오늘은 이곳 근처에 후킹을 걸어 번역이 제대로 이루어지는지 확인확인할 것이다.

(어째서 기록3에서 이곳에 테스트를 하지 않았는지가 의문이다...)

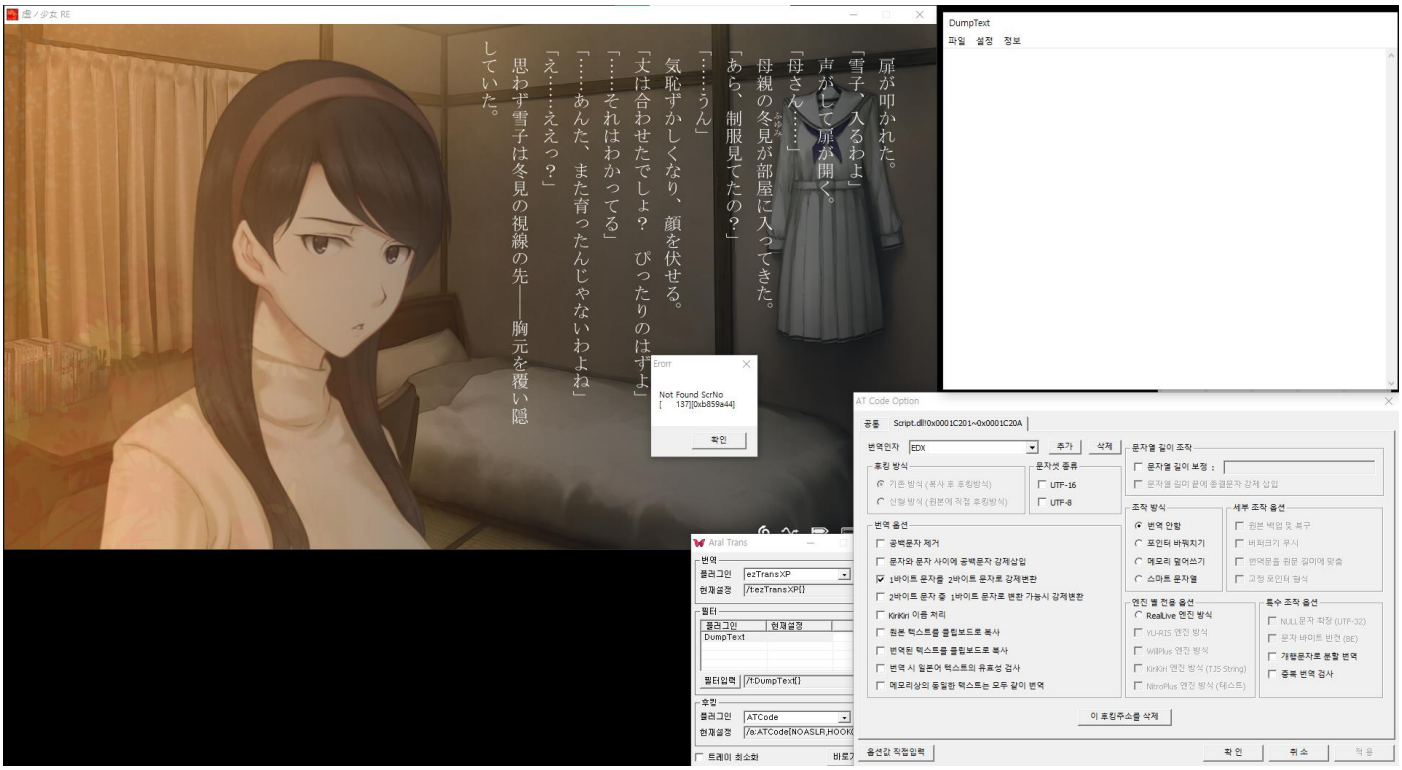


ㅋㅋ 반복문 안쪽에 후킹걸었더니 이런 참사가 일어났다.

반복한번 할때마다 번역을 해버리니 계단처럼 계속 나오네..

```
58F4C201  > 8B93 4C1F0000  MOV EDX,DWORD PTR DS:[EBX+1F4C]   Start of COPY TO THIRD
58F4C207  . 8A0C10         MOV CL,BYTE PTR DS:[EAX+EDX]
58F4C20A  . 880C06         MOV BYTE PTR DS:[ESI+EAX],CL      COPY TO THIRD
58F4C20D  . 40             INC EAX
58F4C20E  . 3BC5           CMP EAX,EBP
58F4C210  .^7C EF          JL SHORT Script.58F4C201
58F4C212  > 0FB615 A3C34B5 MOVZX EDX,BYTE PTR DS:[594BC3A3]  End of COPY TO THIRD
```

(수정하며 나중에 다시 캡처한 화면) 1C201~1C20A까지가 위 회색부분이다. 1C201에서 시작해서 1C20A에 반환하는 방식은 어떨까 테스트해본 것.

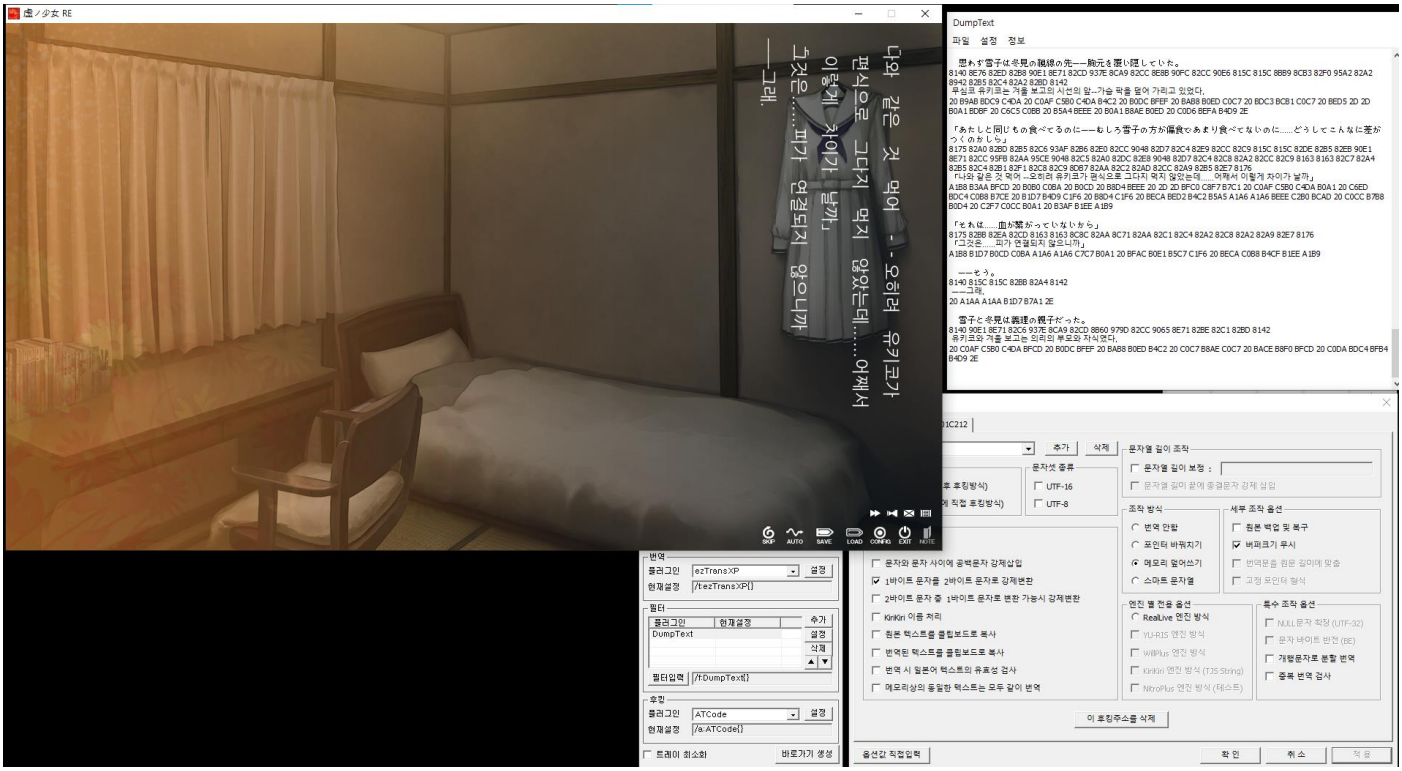


1C201 ~ 1C20A, EDX

반복문 안쪽에서 EDX를 잡아오는 시도 해보다가 어디선가 오류난 모습

일단 기록해두기.

<코드파인딩 초기에 발견한 1C212 지점 다시보기>



0x0001C212, EDX 후킹.

(위 사진은 크래시 나기 직전 놀라운 순발력으로 캡처한것이다..)

메모리 덮어쓰기로 바꾸는 것은 크래시를 유발하지 않으나, 세부조작옵션의 '버퍼크기 무시'를 켤 경우가 문제가 된다.

메모리덮어쓰기만을 이용하는경우 크래시 없이 작동한다. 하지만 이 경우는 번역문의 길이가 원문보다 길어지면 대사를 그냥 잘라버리고 뒷부분을 출력하지 않기 때문에 크래시가 나지 않는것이다. (잘라버린다기 보다는 원문의 길이를 저장해두는 값에 맞춰서 문자열을 복사,출력 하는것에 가까울듯.) 이렇게 대사가 잘리는 것은 플레이하기 좋지 않은 조건이므로 이 문제를 해결해야한다.

그래서 등장한 것이 버퍼크기 무시 옵션인데, 이것은 번역문을 그대로 옮겨오기는 하나, 위의 사진처럼 원문의 길이를 초과하는 번역문일 경우 그냥 크래시가 나기 때문에 사용할 수가 없다..

오늘의 기록 정리

굳이 결과를 적어보자면.. 1C201이 어디서 온건지 알아내서 그 직전을 조금 살펴봐야할것같다는정도..?

---- 기록 10을 끝내고 내용추가 ----

9번째 기록을 작성하면서 미처 생각치 못한 부분이 있어서, 이번 정리보다는 10번 기록을 보는 게 더 좋을거같아요~