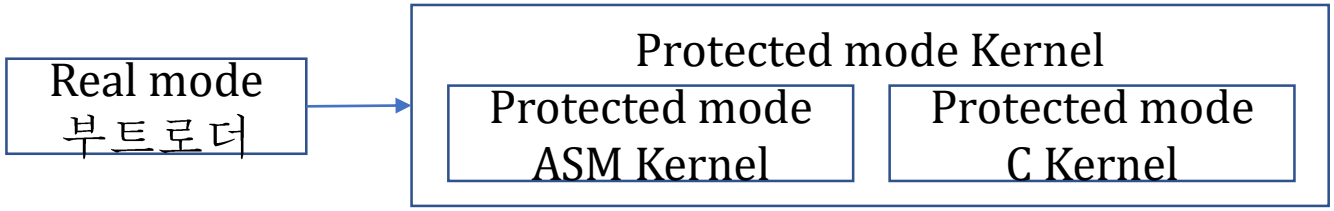
Homework #2

- Secure Boot이란 부팅 시 사용되는 부트로더 및 커널의 코드가 Known good state에 있는지 확인하고, 그럴 경우에만 부팅을 허용하는 것이다.
- 본래 Secure Boot을 구현하려면, 각각의 부트로더 및 커널 이미지에 디지털 서명 기능을 추가하고, 부팅 과정에서 부트로더 및 커널이 변조되었는지를 확인하는 해야 한다.
- 본 과제에서는 이를 단순화 하여 구현하고자 한다.

Homework #2

- Secure Boot 메커니즘은 커널 이미지에만 적용한다.



- ImageMaker를 수정하여 커널 이미지의 "최상단"에 hash 값이 저장되도록 한다.
  - 4 byte hash값을 가정한다
  - hash 값은 전체 커널 이미지를 4 byte로 쪼갠 뒤 모두 xor을 하여 계산한다.

    - hash value = (1^{st} 4B) xor (1^{nd} 4B) xor (3^{rd} 4B) xor (4^{th} 4B) xor …
  - 4 byte hash값은 Kernel32.bin 파일의 0번지에 저장된다.

Homework #2

- BootLoader는 커널 이미지를 로딩 한 후에, 로딩된 커널 이미지에서 hash값을 앞서와 동일한 방법으로 계산한다.
  - 단! 이 때, 커널 이미지의 0번지에는 4byte의 hash값이 있기 때문에, hash값 계산은 4번지부터 시작하도록 한다.
- hash값이 계산되면, 커널 이미지의 0번지에 저장된 hash값과 일치하는지 확인한다.

Homework #2

- 최종적으로 기대하는 출력화면은 다음과 같다.

  MINT64 OS Boot Loader Start~!!
  Current Time: 23:12:44
  OS Image Loading… Complete~!!
  OS Image Checking… Okay~!!
  Switch To Protected Mode Success~!!
  C Language Kernel Started~!!

- Kernel32.bin 파일을 hex editer로 임의로 수정한 뒤에는 다음과 같은 출력화면이 나와야 한다.

  MINT64 OS Boot Loader Start~!!
  Current Time: 23:12:44
  OS Image Loading… Complete~!!
  OS Image Checking… Fail~!!
  Loaded Hash Value: 1234ABCD
  Calcuated Hash Value: 3322AABB

Hint #1

- OS의 hash값 계산 및 무결성 검사는 리얼모드에서 Kernel32.bin을 로딩할 때 해야 한다
- 리얼드에서는 16-bit로 동작한다. 따라서 4 B로 xor 연산을 수행하는데 유념한다.
- hash 값이 커널 이미지에 추가되었다. 이 hash값은 secure boot용도로만 사용된다. 따라서, 커널 이미지를 로딩할 때, 이 hash값은 버리도록 하자.