Attach 함수

어떤 드라이버에 전달되는 IRP들을 감시하고 필요한 작업을 추가하기 위하여 Filter Driver를 만든다면, Filter Driver는 IRP를 처리하는 기존의 드라이버 위에 attach 되어야 Filter Driver가 원하는 동작을 처리할 수 있다. 각각의 Device Object 구조체는 IRP를 처리하는 Device Object에 attach된 장치의 목록을 링크드 리스트(linked list)로 관리하기 위하여 AttachedDevice라는 필드를 가지고 있다. 어떤 Device Object에 attach 되는 Device Object는 하나만 존재하지 않기 때문에, 하나의 Device Object에 attach된 Device Object는 링크드 리스트로 관리된다.
기본적으로 Filter Driver를 기존의 드라이버 위에 attach 시키기 위하여 사용하는 함수는 IoAttachDeviceByPointer(), IoAttachDeviceToDeviceStack(), IoAttachDevice() 이다. 여기서는 새로 생성된 Device Object를 attach 시키기 위한 각 함수들의 차이점에 대해서 간단히 알아보자.

1. IoAttachDeviceByPointer()

IoAttachDeviceByPointer() 함수를 사용하여 새로 생성된 Device Object를 감시 대상이 되는 Device Object에 attach 시킨다면, I/O Manager는 IoAttachDeviceByPointer(), IoAttachDeviceToDeviceStack(), IoAttachDevice()의 순서대로 attach 동작을 구현한다.

1. I/O Manager는 IRP를 처리하는 Device Object에 attach된 다른 Device Object가 있는지 조사하여 가장 상위에 위치한 Device Object의 포인터를 얻는다. 이러한 동작을 수행하기 위하여 I/O Manager는 IoGetAttachedDevice() 함수를 제공하며, 기본적으로 동작하는 과정은 다음과 같다.

```
PDEVICE_OBJECT
IoGetAttachedDevice (PDEVICE_OBJECT TargetDeviceObject)
{
    PDEVICE_OBJECT   ReturnedDeviceObject = TargetDeviceObject;

    While( ReturnedDeviceObject->AttachedDevice )
    {
        ReturnedDeviceObject = TargetDeviceObject->AttachedDevice;
    }
}
```

스택에 기반을 둔 링크드 리스트에 attach된 Device Object의 링크드 리스트를 생각해 보면, 리스트에 가장 마지막으로 attach된 Object는 리스트의 처음에 놓인다. 따라서 attach 동작을 구현하기 위한 마지막 Device Object는 IRP를 마지막으로 처리하는 Device Object에 전달되는 IRP를 가로채기 위한 첫 번째 Object임을 알 수 있다.
이러한 방식(마지막으로 저장된 Object가 감시 대상이 되는 Device Object에 전달되는 IRP를 첫 번째로 받는 방식)을 유지하기 위해, I/O Manager는 attach 요청을 계속 수행하기 위하여 Device Object의 링크드 리스트에 있는 가장 상위의 Device Object에 대한 포인터를 얻는다. 그러나 새로 생성된 Device Object가 감시 대상이 되는 Device Object에 대한 attach를 첫 번째로 요구한다면(다른 종류의 Device Object가 감시 대상이 되는 Device Object에 attach 되지 않은 상태), I/O Manager는 다음 단계를 수행하기 위해 직접 감시 대상이 되는 Device Object의 포인터를 사용한다.

만약 Device Object가 제거 되었거나, 새로 생성된 Device Object가 attach 되고자 하는 드라이버가 IRP를 마지막으로 처리하는 Device Object에 대한 unload 작업을 수행 중이라면, I/O Manager는 Device Object의 attach 요청을 바로 거부한다. 이런 상황에서 반환되는 에러 코드는 STATUS_NO_SUCH_DEVICE이다. I/O Manager가 드라이버 스택의 가장 상위에 있는 Device Object에 대한 포인터를 얻었다면, 다음 단계를 진행한다.

2. I/O Manager는 구현하고자 하는 attach 동작을 완료한다. I/O Manager는 attach 동작을 완료하기 위하여 다음과 같은 단계를 수행한다.
   1) ReturnedDeviceObject->AttachedDevice 필드는 attach 하고자 하는 Device Object의 포인터로 설정된다.
   2) Attach 하고자 하는 Device Object 구조체 내에 있는 StackSize 필드는 (ReturnedDeviceObject->StackSize + 1)로 설정된다.
   일단 attach가 성공적으로 마무리 되면, I/O Manager는 IRP를 마지막으로 처리하는 드라이버에 전달되는 모든 IRP를 새로 attach 된 드라이버에게 전달한다. I/O Manager는 새로 attach 된 드라이버에서 IRP를 가지고 무엇을 하는지 모른다. 그러나 최악의 상황을 고려하면, 새로 attach 된 드라이버에서 어떤 작업을 수행하고(선처리 작업) Completion Routine을 등록한 후, IRP를 계층화된 드라이버 리스트의 다음 드라이버에게 수정된 IRP를 전달하는 상황을 생각할 수 있다. 새로 추가된 Device Object는 하나 이상의 계층화된 드라이버를 통하여 전달된 IRP를 요구하기 때문에, I/O Manager는 IRP를 마지막으로 처리하는 Device Object에 전달하기 위하여 계속해서 발생하는 모든 IRP를 위해 할당된 Stack Location의 수를 IRP를 처리하는 모든 드라이버를 통과하여 마지막 드라이버에서 처리할 수 있도록 보장해야 한다. I/O Manager는 새로 추가된 드라이버에서 생성된 Device Object에 있는 AlignmentRequirement와 Secorsize 필드를 감시 대상이 되는

Device Object에 있는 필드 값과 동일하게 설정해야 한다.

2. IoAttachDeviceToDeviceStack()

이 함수는 Windows NT 4.0에서 처음 사용되었으며, 기능적으로 IoAttachDeviceByPointer() 함수의 처리 과정과 유사하고 동일한 방법으로 호출된다(특히 개발자가 이 함수를 사용하기 위해서는 attach 하고자 하는 Device Object와 attach 대상이 되는 Device Object의 값을 제공해야 한다). 그러나 이 함수는 하나의 추가적인 작업을 구현한다. 만약 attach 동작이 성공적으로 마무리되면 IoAttachDeviceToDeviceStack() 함수는 새로 생성된 Device Object가 attach 된 드라이버 계층 구조상 가장 상위에 있는 Device Object의 포인터를 반환한다. 반환된 Device Object 포인터는 Filter Driver가 수정된 IRP를 드라이버의 스택 구조상 다음에 위치한 드라이버에게 전달한다면 유효하다고 말할 수 있다.
만약 새로 생성된 드라이버가 attach 대상이 되는 Device Object에 대해서 처음 attach 동작을 구현한다면, 반환된 Device Object 포인터는 새로 생성된 드라이버가 IoAttachDeviceToDeviceStack() 함수를 호출할 때 전달한 attach 대상이 되는 Device Object 포인터와 같다.

3. IoAttachDevice()

이 함수는 다음과 같이 정의되어 있다.

```
NTSTATUS
IoAttachDevice (
    IN  PDEVICE_OBJECT      SourceDevice,
    IN  PUNICODE_STRING     TargetDevice,
    OUT PDEVICE_OBJECT      *AttachedDevice
);
```

IoAttachDevice() 함수는 두 Device Object의 연결 동작을 구현한다. 그러나 이 함수는 감시 대상이 되는 Device Object 포인터 대신에 감시 대상이 되는 장치의 이름을 받아들인다. 이것은 attach 하고자 하는 드라이버를 대신하여 attach 대상이 되는 Device Object를 열고 실질적인 attach 동작을 구현하기 위하여 감시 대상이 되는 Device Object 포인터를 사용한다.
이 함수에서 실행되는 절차는 다음과 같다.

1. IoAttachDevice() 함수는 attach 대상이 되는 Device Object 포인터를 얻기 위하여

내부적으로 IoGetDeviceObjectPointer() 함수를 호출한다. 이때 DesiredAccess 인자 값은 FILE_READ_ATTRIBUTES로 설정된다.

2. IoAttachDevice() 함수는 attach 시키고자 하는 Device Object와 attach 대상이 되는 Device Object 사이의 연결을 구현하기 위해 앞에서 설명했던 것과 같은 동일한 단계를 차례로 실행한다.
   IoAttachDeviceByPointer() 함수의 경우처럼 I/O Manager는 attach 시키고자 하는 Device Object 구조체에 있는 StackSize, AlignmentRequirement, SectorSize 필드의 값을 초기화 한다.
3. I/O Manager는 내부적으로 IoGetDeviceObjectPointer()를 호출함으로써 반환된 File Object 포인터 값을 감소시킨다.
4. Attach 되기 전 가장 상위에 있는 Device Object의 포인터는 AttachedDevice 인자에 저장되어 호출자에게 반환된다.

한가지 명심해야 할 것은 새로 생성된 드라이버는 IoAttachDevice() 함수를 호출하기 전 attach 시키고자 하는 Device Object를 생성해야만 한다.

개발자는 아마 새로 개발한 드라이버에서 IoAttachDeviceByPointer() 함수나 IoAttachDeviceToDeviceStack() 함수를 호출한 후에 IoGetDeviceObjectPointer() 함수를 호출하는 대신, 직접 IoAttachDevice() 함수를 호출하는 것이 더 바람직한지에 대해 걱정할 것이다.

Attach 동작을 구현하는데 있어서 두 방법 사이에 하나의 미묘한 차이가 있다. 이러한 차이점은 새로 개발한 드라이버가 File System Driver의 Logical Volume Device Object(Lower-level Device Driver인 Disk Device Object의 위에 놓여지는 것과는 반대의 개념으로) 위에 추가되기를 바란다면 이 차이점은 중요하다.

IoAttachDevice() 함수는 항상 FILE_READ_ATTRIBUTES로 설정된 DesiredAccess 값을 가지고 감시 대상이 되는 장치를 연다. 이런 파일 I/O는 마운트 동작이 아직 발생하지 않았다면, I/O Manager는 목표로 하는 물리 장치나 가상 장치 또는 놀리 장치(Physical/Virtual/Logical Device) 위에서 초기화 된 상태인 마운트 동작을 발생시키지 않는다. 따라서 여러분의 드라이버가 드라이브 C:에서 마운트 된 Logical Volume을 나타내는 Device Object에 attach 되기를 원하거나, 드라이버에서 제공된 이름이 "₩Device₩C:"라면, IoAttachDevice()는 우리가 기대했던 동작을 처리할 것이라고 확신하지 못한다. 왜냐하면 "₩Device₩C:"가 정의한 물리 Device Object에 attach 된 Device Object를 가지고 마무리 될 수 있기 때문이다.

그러나, 새로 생성한 드라이버가 항상 감시하고자 하는 드라이브에서 마운트 된 Logical Volume을 담당하는 File System Driver의 Device Object에 항상 attach 되기를 바란다면, DesiredAccess 값을 FILE_READ_ACCESS와 같은 종류로 설정함으로써 드라이버에서 직접

IoGetDeviceObjectPointer() 함수를 호출할 수 있다. 이러한 DesiredAccess 값은 어떠한 Logical Volume도 감시 대상이 되는 장치에 마운트 되지 않았다면 I/O Manager는 마운트 과정을 실행한다. 그리고 반환된 Device Object 포인터는 마운트 된 Logical Volume을 담당하는 Device Object를 참조한다. 그 다음 File System Driver는 IoAttachDeviceByPointer() 함수나 IoAttachDeviceToDeviceStack() 함수를 호출함으로써 attach를 요청할 수 있다.

일단, 개발자가 IoAttachDeviceByPointer(), IoAttachDeviceToDeviceStack(), IoAttachDevice()의 세 함수 중 하나를 성공적으로 호출하면, I/O Manager가 attach 동작을 마무리 했다는 것을 확신할 수 있다.

개발자는 attach 동작을 요청할 때마다 주의해야 한다. 왜냐하면 감시 대상이 되는 Device Object에 전달되는 모든 새로운 IRP는, IRP를 처리해야 하는 장치에 전달되는 것이 아니라 새로 attach 된 드라이버로 전달되기 때문이다. 따라서 이런 요청들을 바로 처리할 수 있도록 준비를 하거나, 그렇지 않으면 모든 초기화를 완료할 때까지 IRP를 대기 상태(block)로 두어야 한다.