MFT & INDX Slack Analysis

proneer

proneer@gmail.com

http://forensic-proof.com

Security is a people problem

개요

1. 슬랙 공간

2. MFT 슬랙

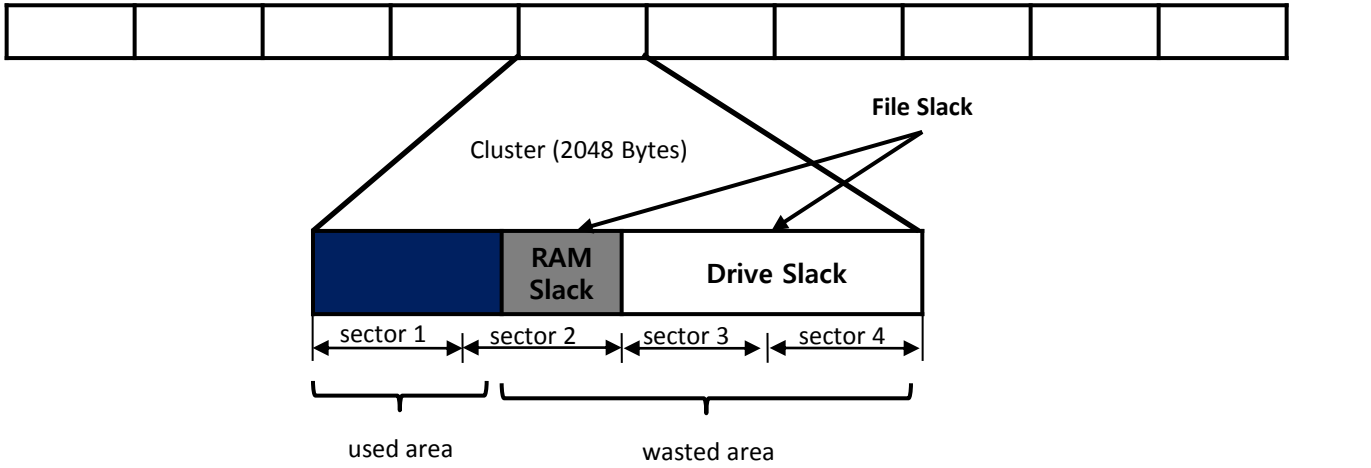
3. INDX 슬랙

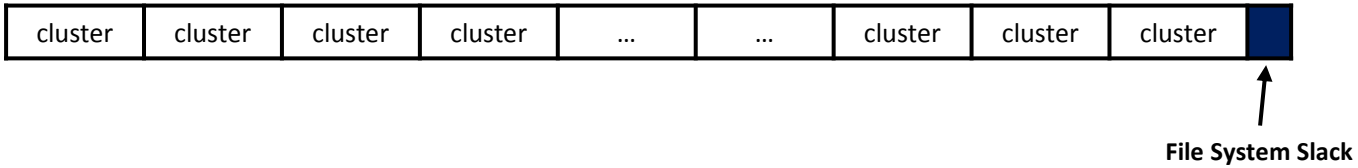
슬랙 공간

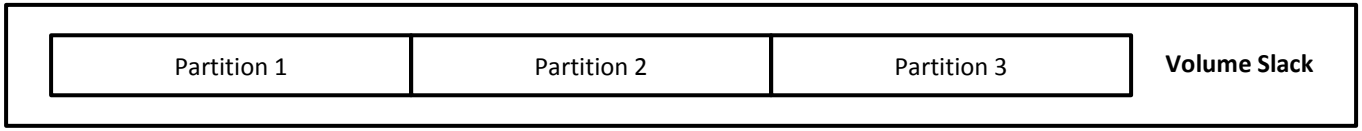
슬랙 소개

- **슬랙 (Slack)**
  - 물리적인 구조와 논리적인 구조의 차이로 발생하는 낭비되는 공간
    - ✓ 램 슬랙
    - ✓ 드라이브 슬랙
    - ✓ 파일시스템 슬랙
    - ✓ 볼륨 슬랙
    - ✓ MFT 슬랙
    - ✓ INDX 슬랙
    - ✓ … …

램,드라이브,파일시스템,볼륨 슬랙

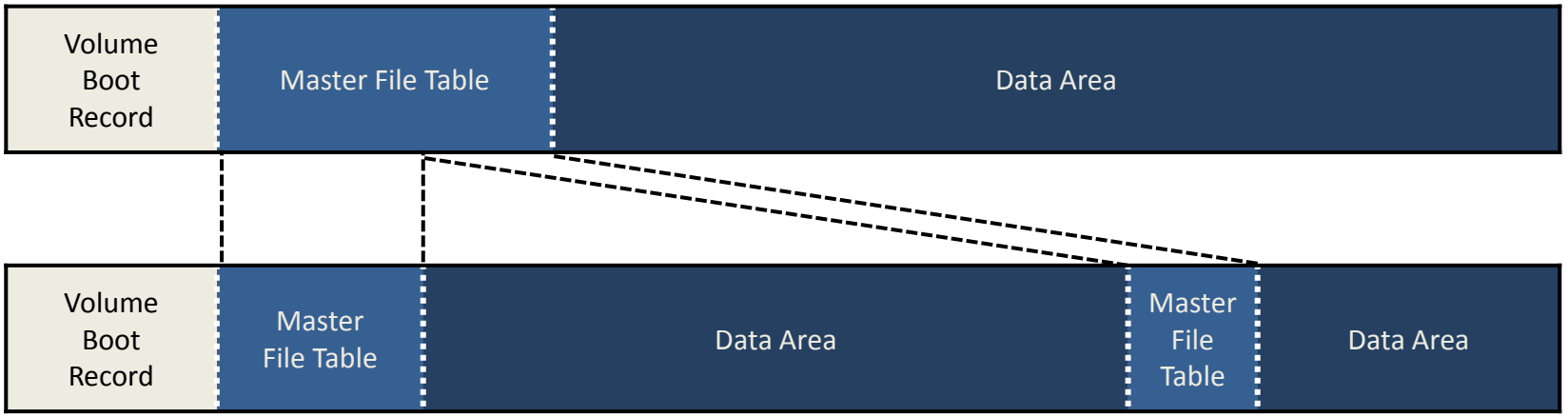






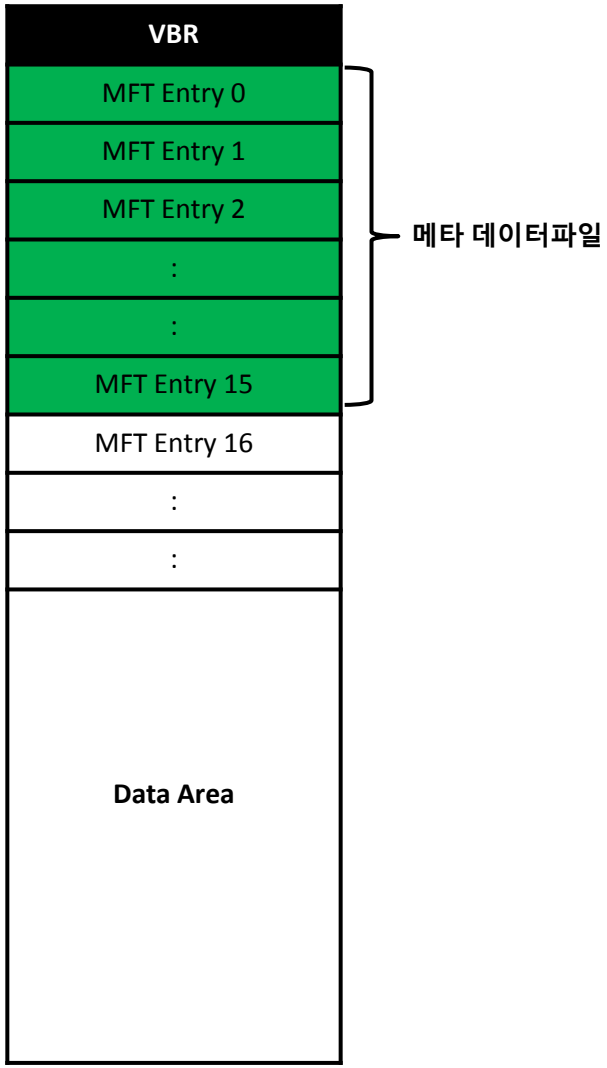
MFT 슬랙

MFT 구조



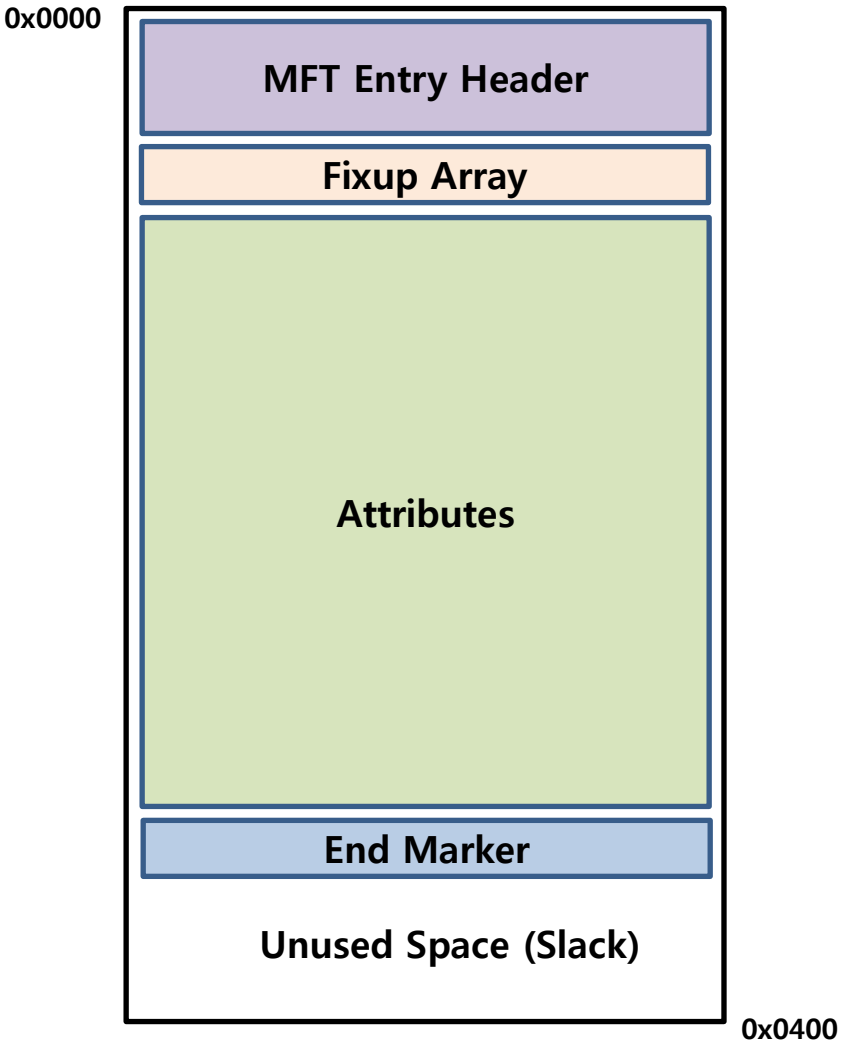
- NTFS는 파일, 디렉터리, 메타정보를 모두 파일 형태로 관리
- 각 파일의 위치, 속성, 시간정보, 이름, 크기 등의 메타정보는 MFT Entry라는 특별한 구조로 저장
- MFT(Master File Table)는 NTFS 상에 존재하는 모든 파일의 MFT Entry의 모음
- MFT 영역은 파일시스템 상의 파일 수에 따라 동적으로 할당
- 일반적으로 볼륨의 12.5% 정도가 MFT 영역으로 할당
- MFT Entry 0 ~ 15번은 파일시스템 생성 시 함께 생성되는 예약된(특별한 용도) 영역

MFT 구조



Entry 번호	Entry 이름	설명
0	$MFT	MFT에 대한 MFT Entry
1	$MFTMirr	$MFT 파일의 일부 백업본
2	$LogFile	메타데이터(MFT)의 트랜잭션 저널 정보
3	$Volume	볼륨의 레이블, 식별자, 버전 등의 정보
4	$AttrDef	속성의 식별자, 이름, 크기 등의 정보
5	.	볼륨의 루트 디렉터리
6	$Bitmap	볼륨의 클러스터 할당 정보
7	$Boot	볼륨이 부팅 가능할 경우 부트 섹터 정보
8	$BadClus	배드 섹터를 가지는 클러스터 정보
9	$Secure	파일의 보안, 접근 제어와 관련된 정보
10	$Upcase	모든 유니코드 문자의 대문자
11	$Extend	$ObjID, $Quota, $Reparse points, $UsnJrnl 등의 추가적인 파일의 정보를 기록하기 위해 사용
12 – 15		미래를 위해 예약
16 -		포맷 후 생성되는 파일의 정보를 위해 사용
-	$ObjId	파일 고유의 ID 정보 (Windows 2000 -)
-	$Quota	사용량 정보 (Windows 2000 -)
-	$Reparse	Reparse Point 에 대한 정보 (Windows 2000 -)
-	$UsnJrnl	파일, 디렉터리의 변경 정보 (Windows 2000 -)

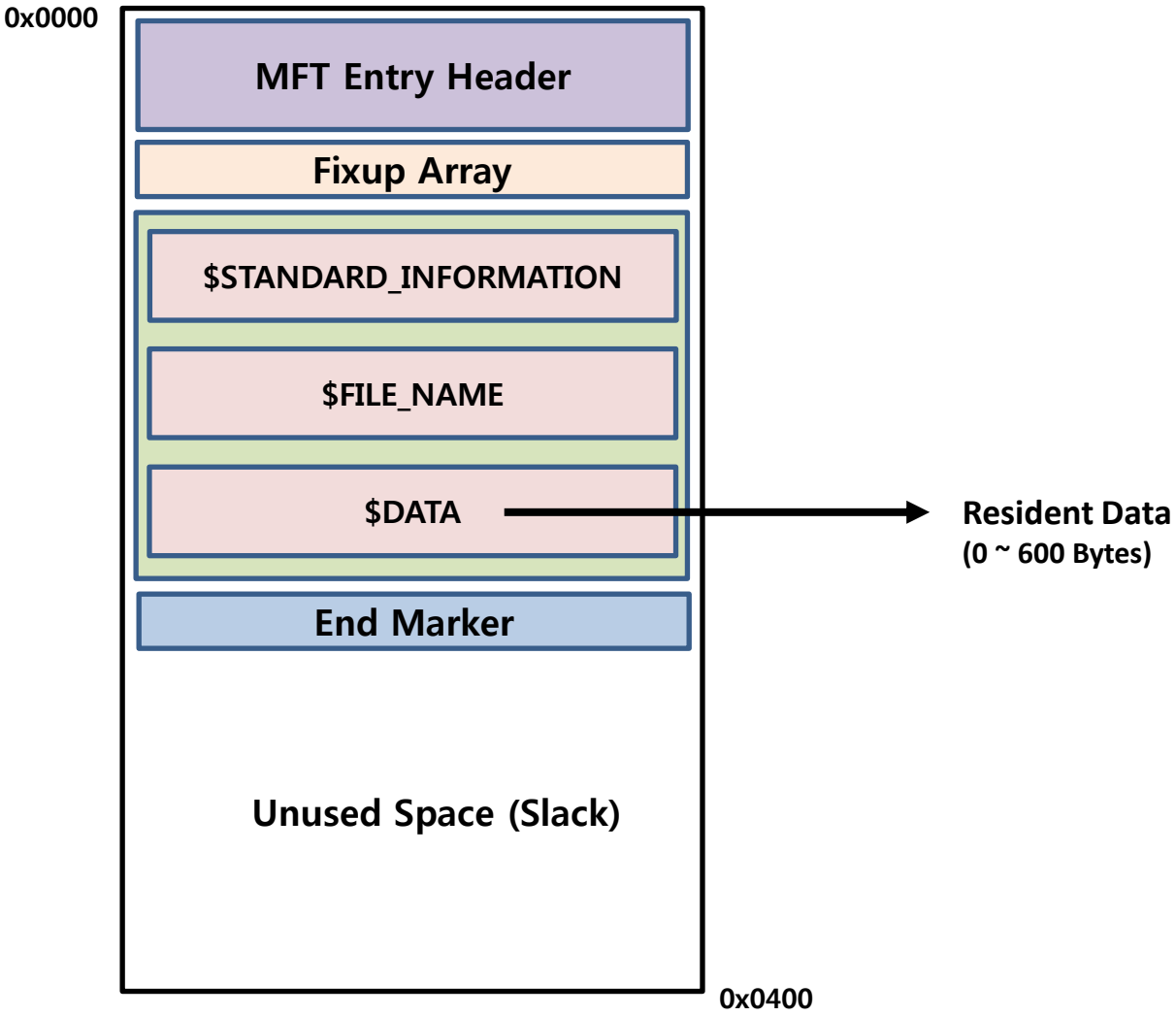
MFT 레코드(엔트리)


0x0000
MFT Entry Header
Fixup Array
Attributes
End Marker
Unused Space (Slack)
0x0400

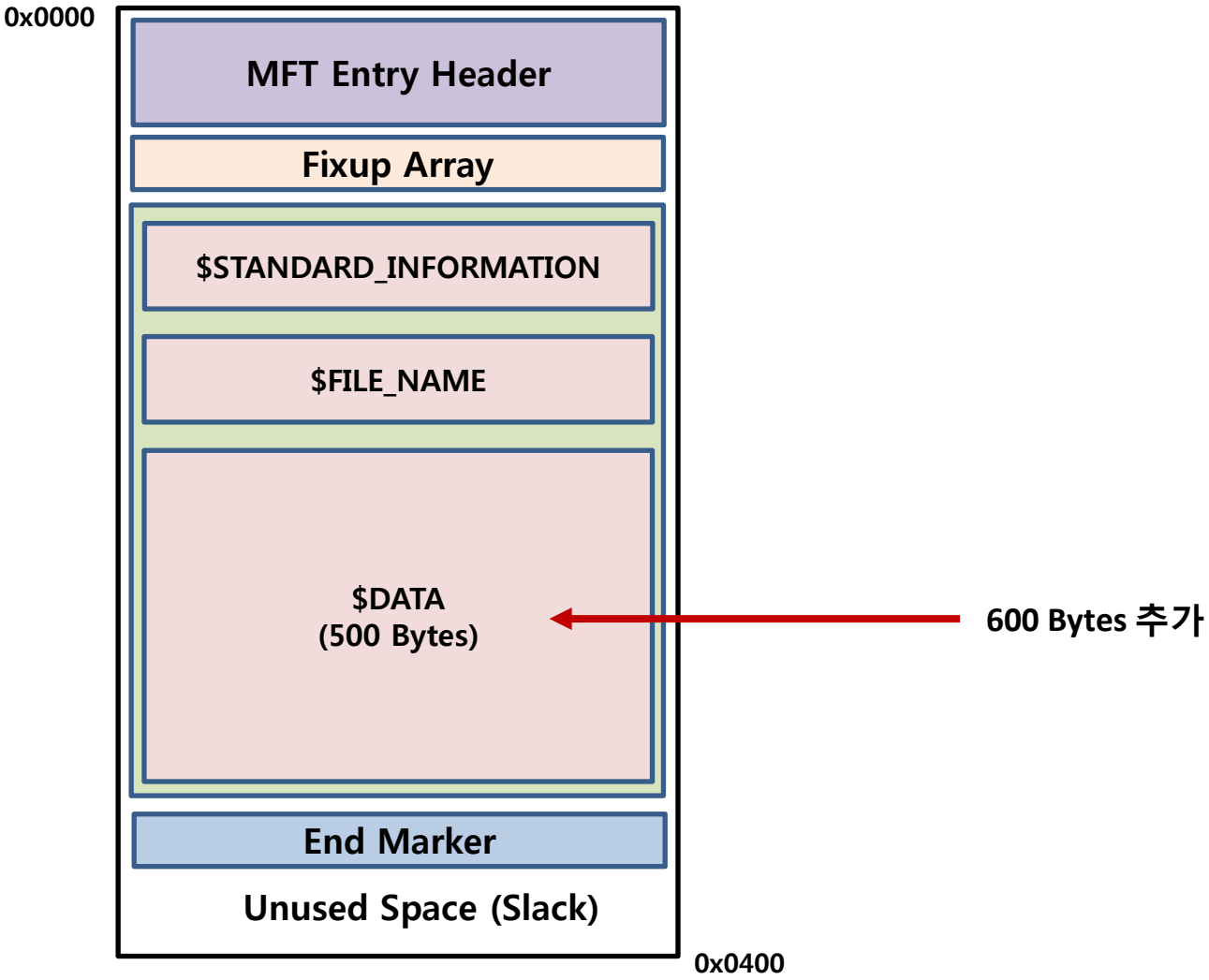
MFT 레코드(엔트리) ➔ 속성(Attributes)

속성 식별값		속성이름	설명
16	0x10	**$STANDARD_INFORMATION**	파일의 생성.접근.수정 시간, 소유자 등의 일반적인 정보
32	0x20	**$ATTRIBUTE_LIST**	추가적인 속성들의 리스트
48	0x30	**$FILE_NAME**	파일 이름(유니코드), 파일의 생성.접근.수정 시간
64	0x40	**$VOLUME_VERSION**	볼륨 정보 (Windows NT 1.2 버전에만 존재)
64	0x40	**$OBJECT_ID**	16바이트의 파일, 디렉터리의 고유 값, 3.0 이상에서만 존재
80	0x50	**$SECURITY_DESCRIPTOR**	파일의 접근 제어와 보안 속성
96	0x60	**$VOLUME_NAME**	볼륨 이름
112	0x70	**$VOLUME_INFORMATION**	파일 시스템의 버전과 다양한 플래그
128	0x80	**$DATA**	파일 내용
144	0x90	**$INDEX_ROOT**	인덱스 트리의 루트 노드
160	0xA0	**$INDEX_ALLOCATION**	인덱스 트리의 루트와 연결된 노드
176	0xB0	**$BITMAP**	$MFT와 인덱스의 할당 정보 관리
192	0xC0	**$SYMBOLIC_LINK**	심볼릭 링크 정보 (Windows 2000+)
192	0xC0	**$REPARSE_POINT**	심볼릭 링크에서 사용하는 reparse point 정보 (Windows 2000+)
208	0xD0	**$EA_INFORMATION**	OS/2 응용 프로그램과 호환성을 위해 사용 (HPFS)
224	0xE0	**$EA**	OS/2 응용 프로그램과 호환성을 위해 사용 (HPFS)
256	0x100	**$LOGGED_UTILITY_STREAM**	암호화된 속성의 정보와 키 값 (Windows 2000+)

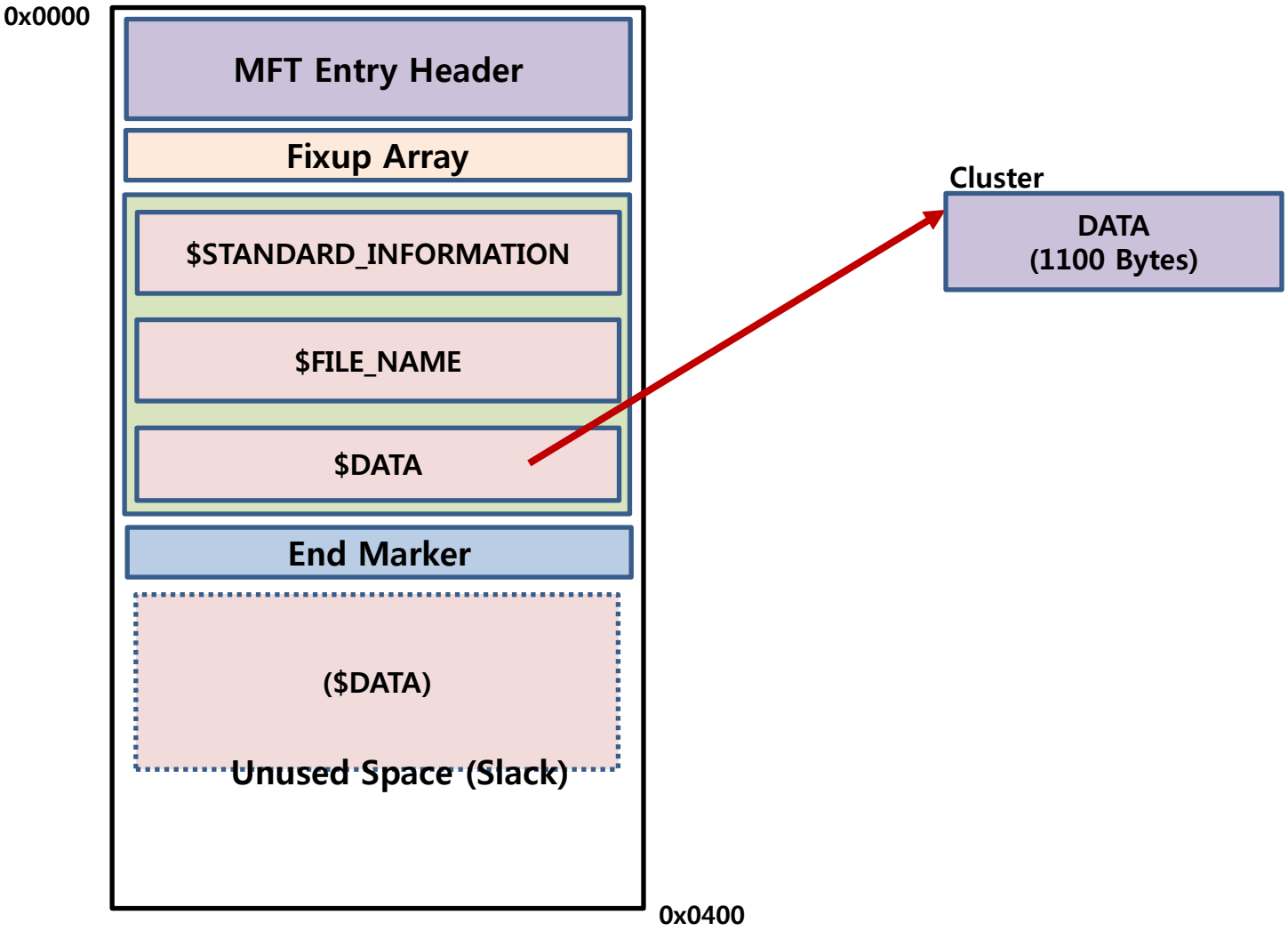
MFT 레코드(엔트리) ➔ 일반 파일



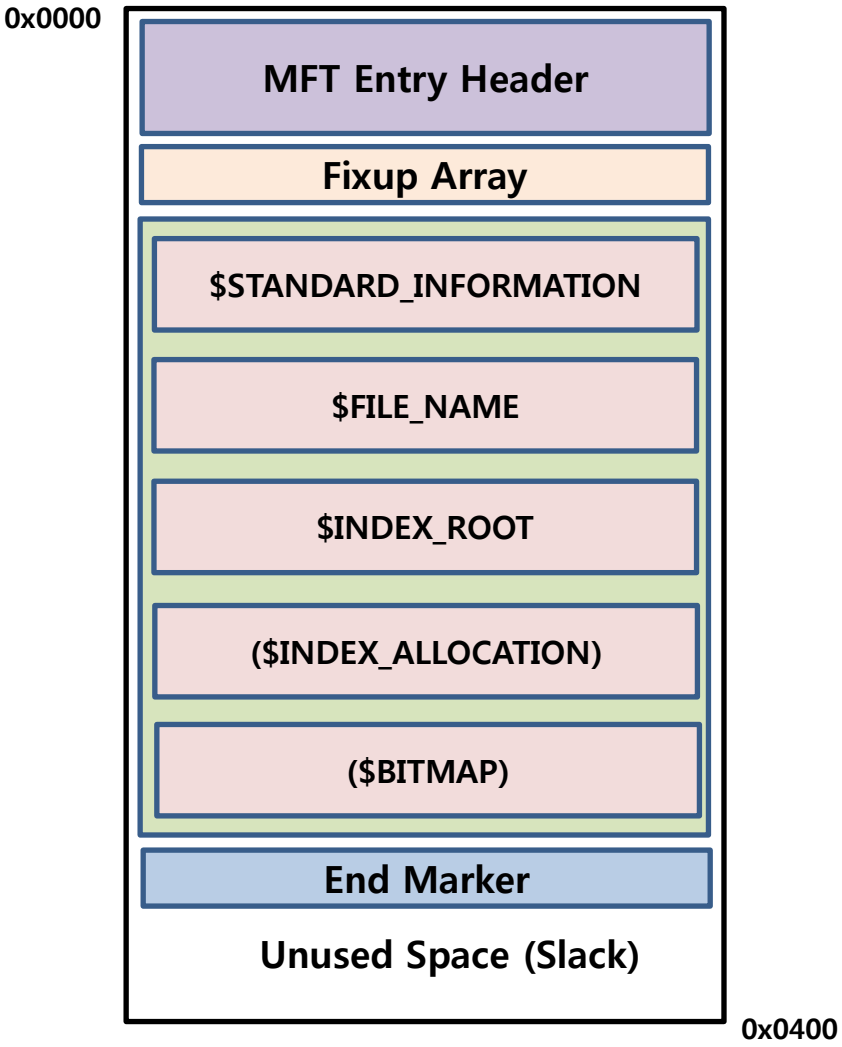
MFT 레코드(엔트리) ➔ 일반 파일



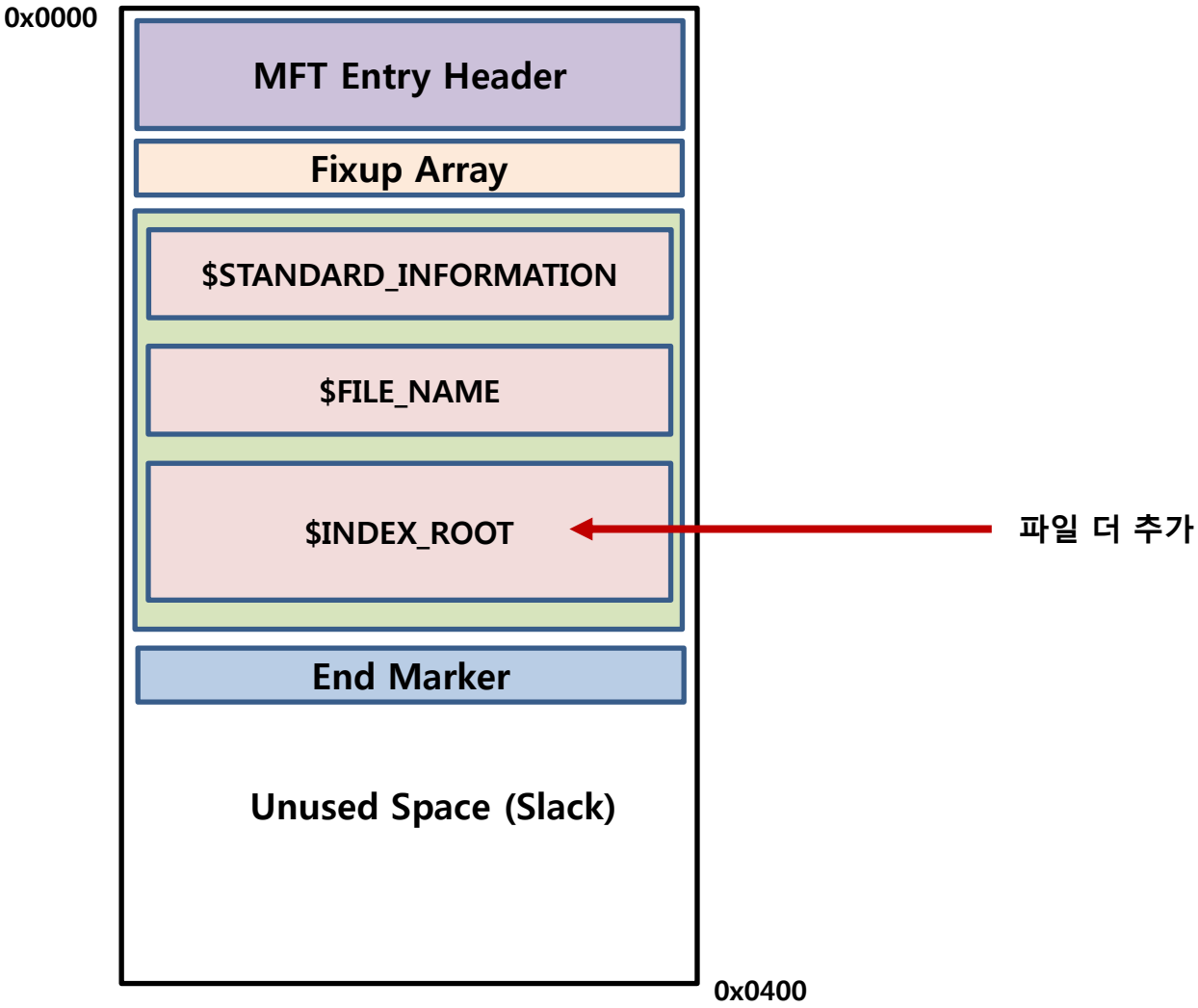
MFT 레코드(엔트리) ➔ 일반 파일



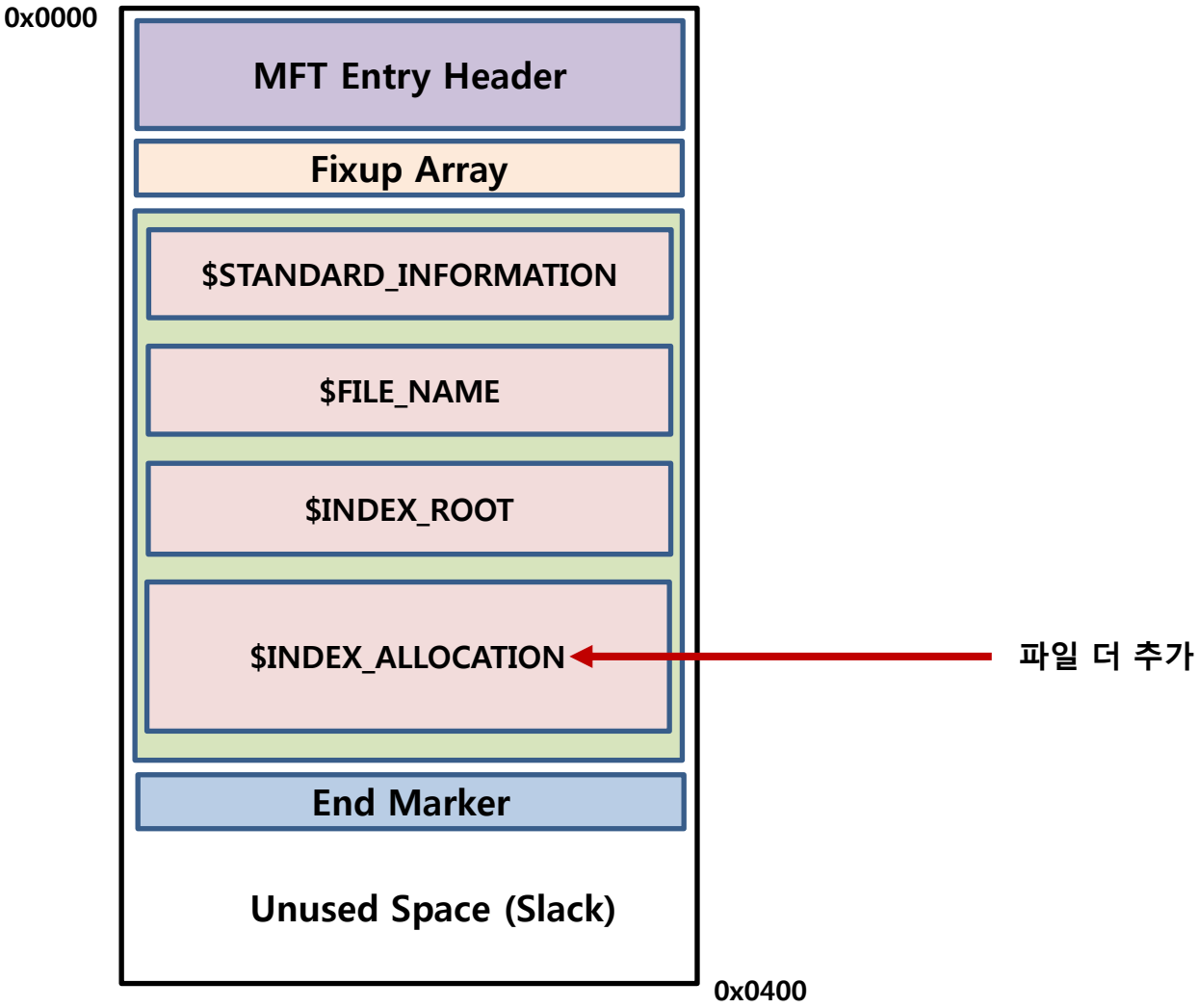
MFT 레코드(엔트리) ➔ 디렉터리



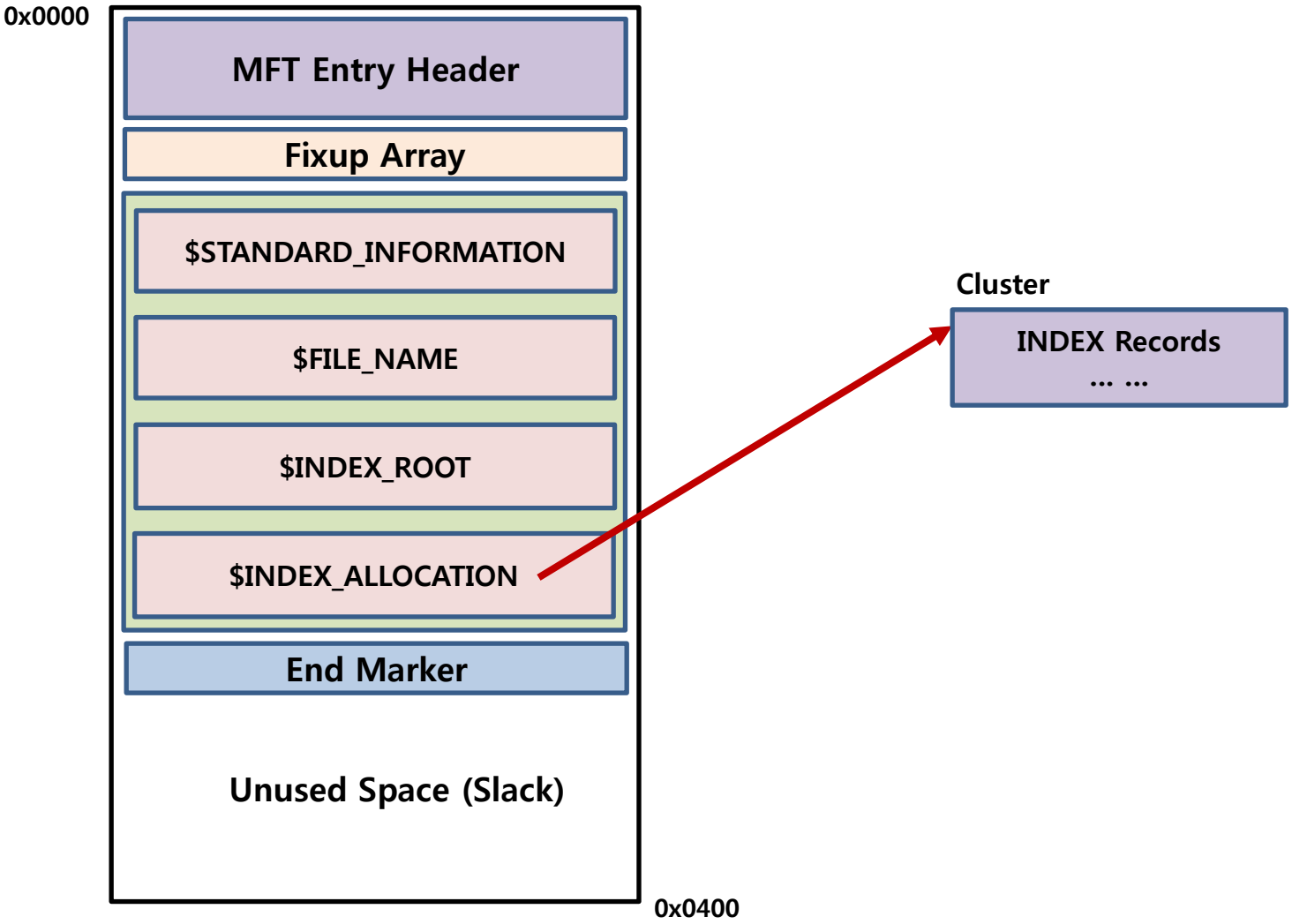
MFT 레코드(엔트리) ➔ 디렉터리



MFT 레코드(엔트리) ➔ 디렉터리



MFT 레코드(엔트리) ➔ 디렉터리



MFT 슬랙 데이터

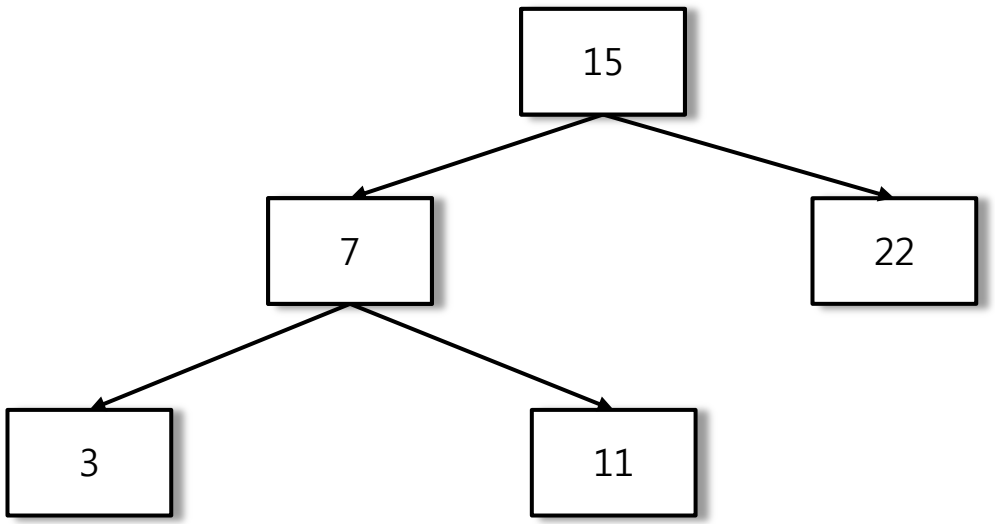
- **일반 파일의 경우**
  - 파일의 내용 ➔ $DATA

- **디렉터리의 경우**
  - 디렉터리 내의 이전/삭제/현재 파일 목록 ➔ $INDEX_ALLOCATION

INDX 슬랙

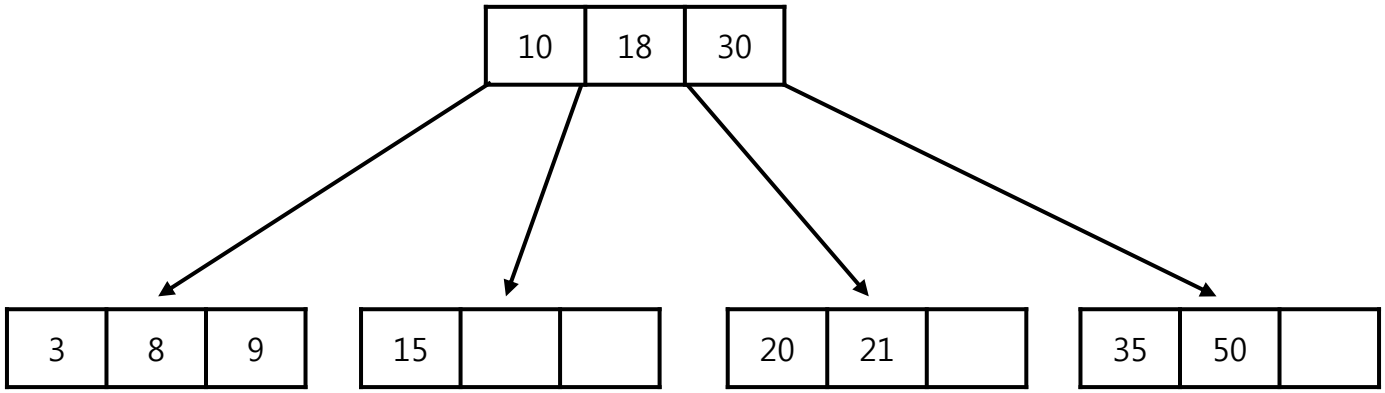
인덱스 구조

- **이진 트리 (Binary Tree)**
  - 한 노드가 최대 2개의 자식 노드를 갖는 트리
  - 노드의 왼쪽 하위트리의 모든 노드값은 키값보다 작음
  - 노드의 오른쪽 하위트리의 모든 노드값은 키값보다 큼



인덱스 구조

- **B 트리 (B Tree)**
  - 차수가 m인 m-원 탐색트리
  - 루트와 단말 노드를 제외한 각 노드는 최소 m/2의 서브 트리를 가짐 (절반 이상이 채워져야 함)
  - 루트는 최소 2개의 서브 트리를 가짐
  - 모든 단말 노드는 같은 레벨



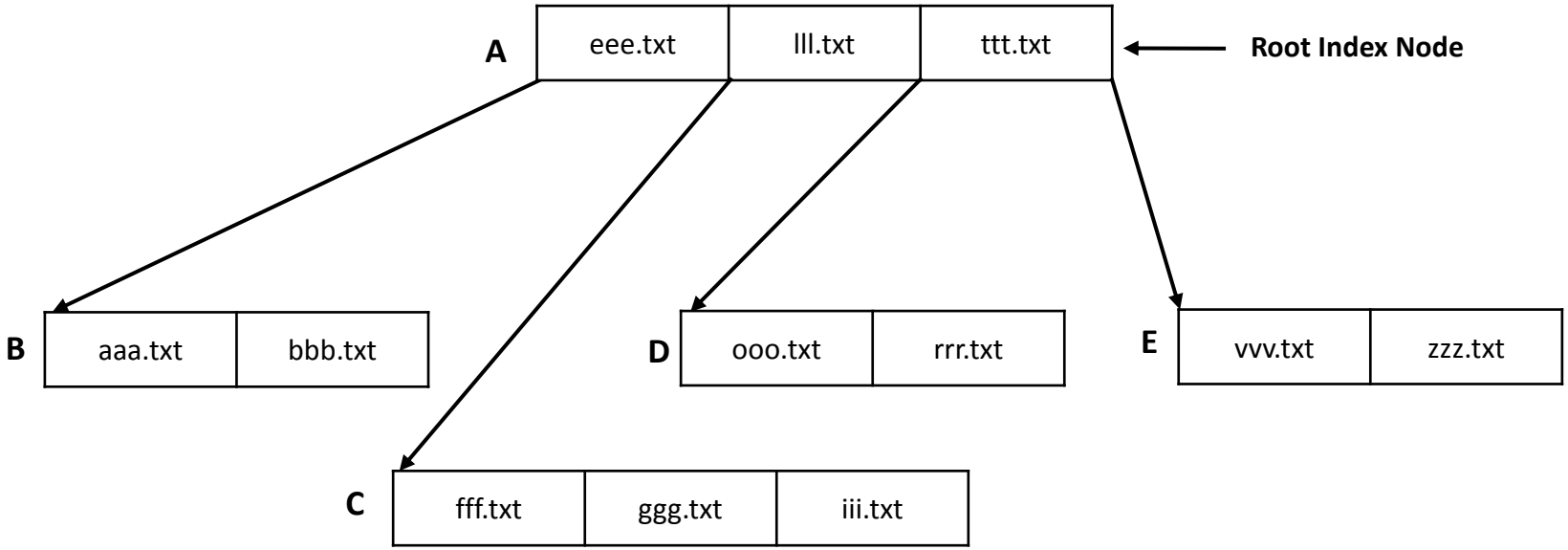
인덱스 구조

- 빠르게 검색이 필요한 데이터는 인덱스 구조로 관리 (디렉터리 등)

인덱스 이름	인덱싱하는 데이터	위치
$I30	$FILE_NAME 속성	디렉터리의 MFT Entry
$SDH	Security Descriptors	$Secure 메타데이터 파일
$SII	Security ID	$Secure 메타데이터 파일
$O	Object ID	$ObjId 메타데이터 파일
$O	Owner ID	$Quota 메타데이터 파일
$Q	Quota	$Quota 메타데이터 파일

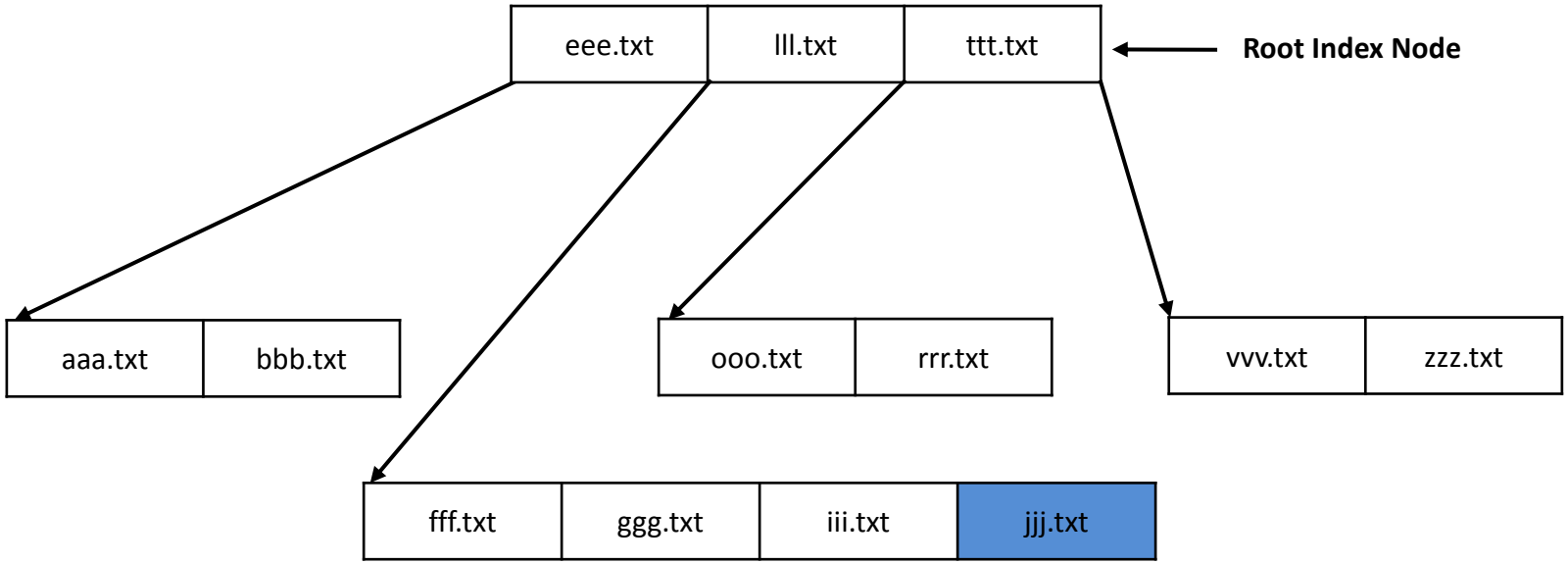
인덱스 구조

- **$I30 (B-Tree)** : B 트리의 노드값은 파일 이름 ($FILE_NAME 속성)



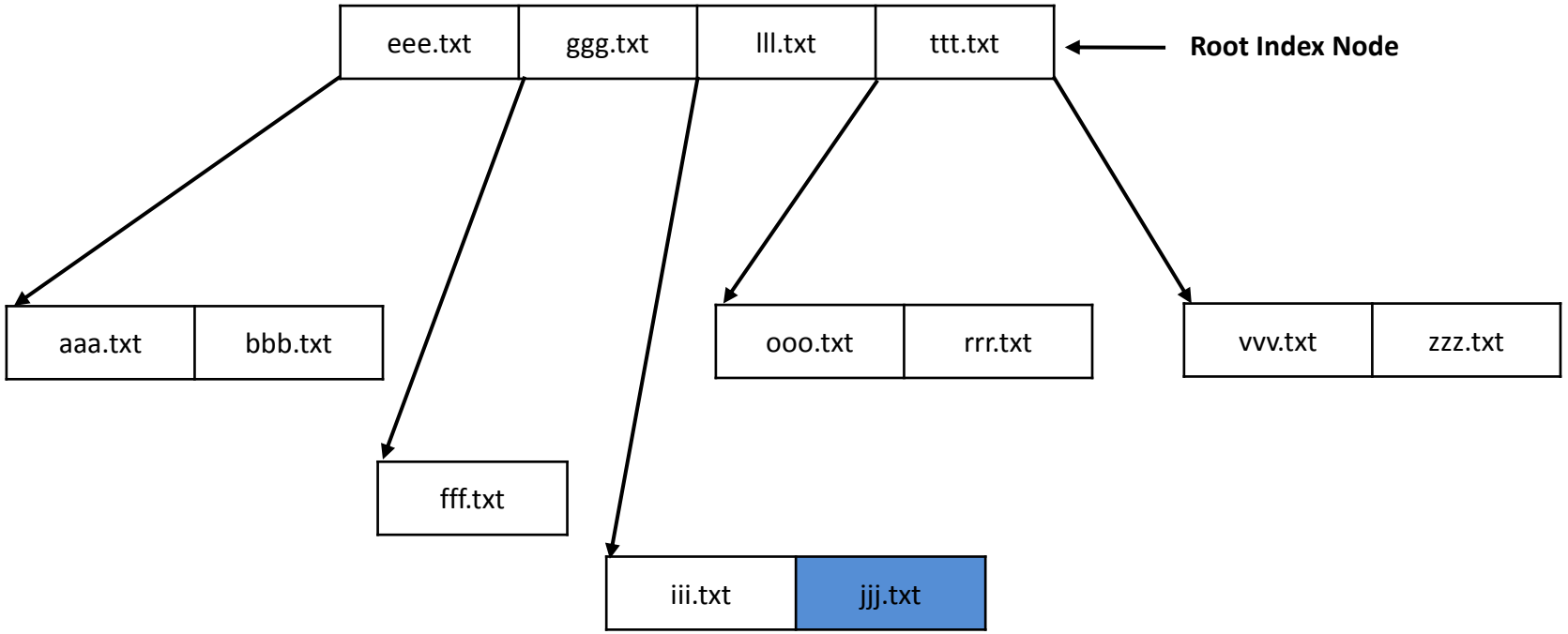
인덱스 구조

- **$I30 (B-Tree)** : jjj.txt 파일 삽입 (1)



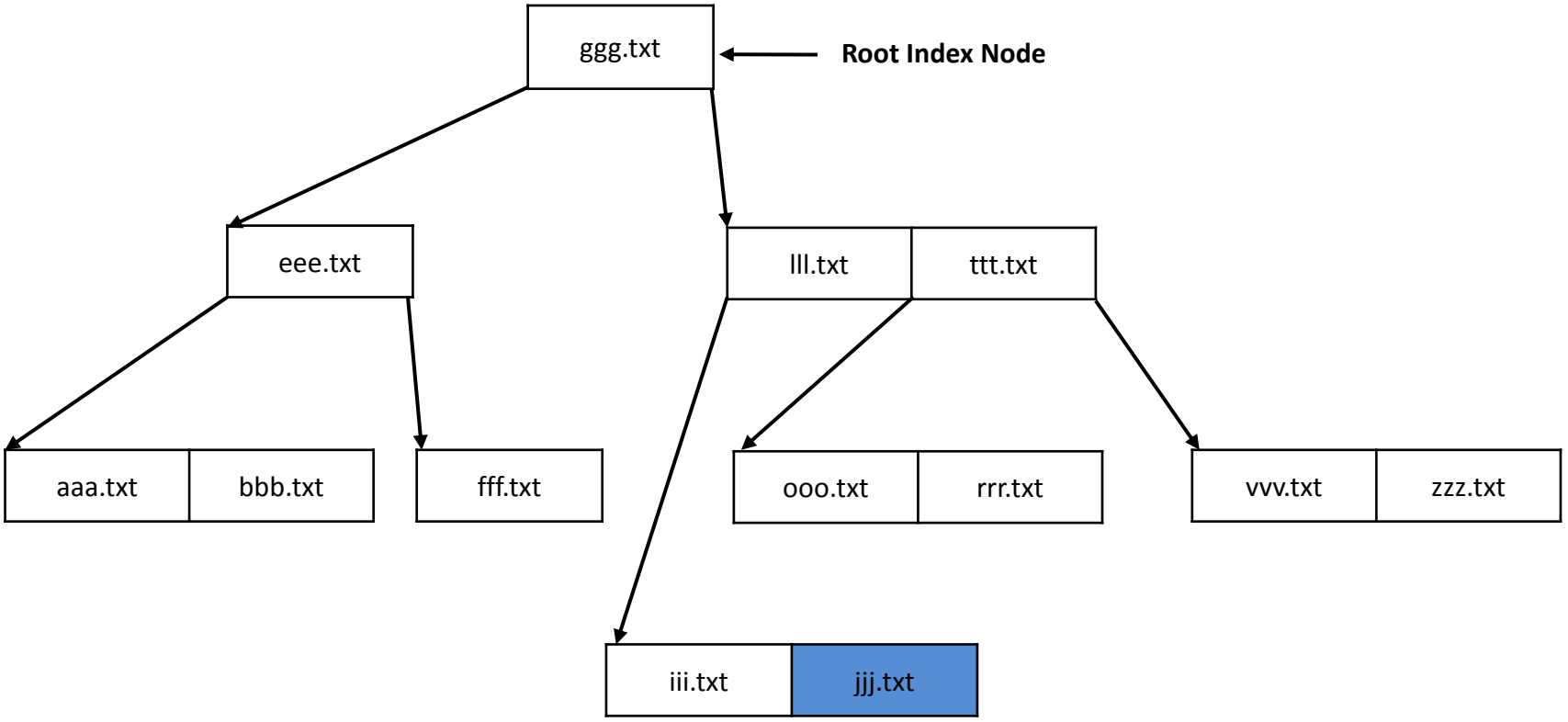
인덱스 구조

- **$I30 (B-Tree)** : jjj.txt 파일 삽입 (2)



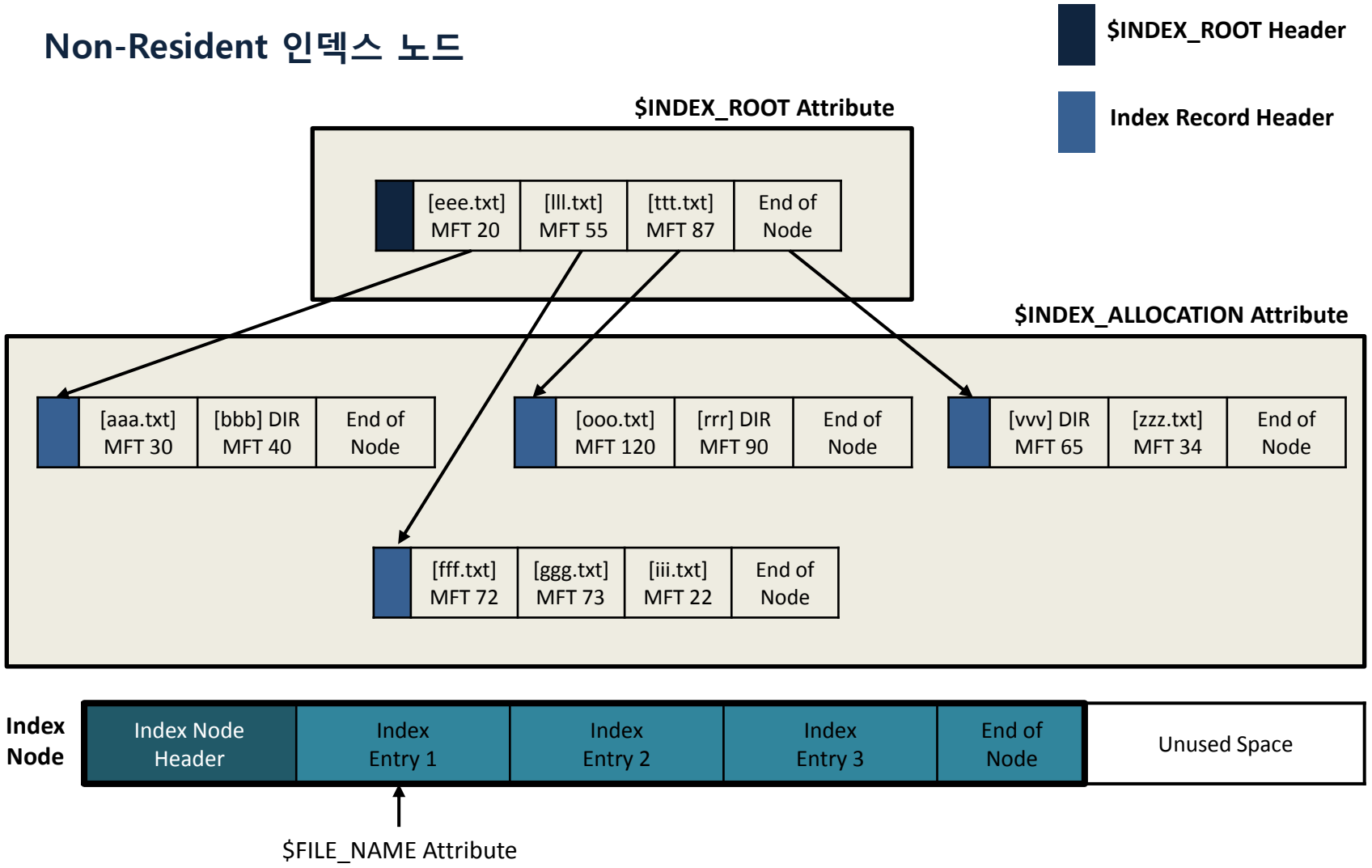
인덱스 구조

- **$I30 (B-Tree)** : jjj.txt 파일 삽입 (3)



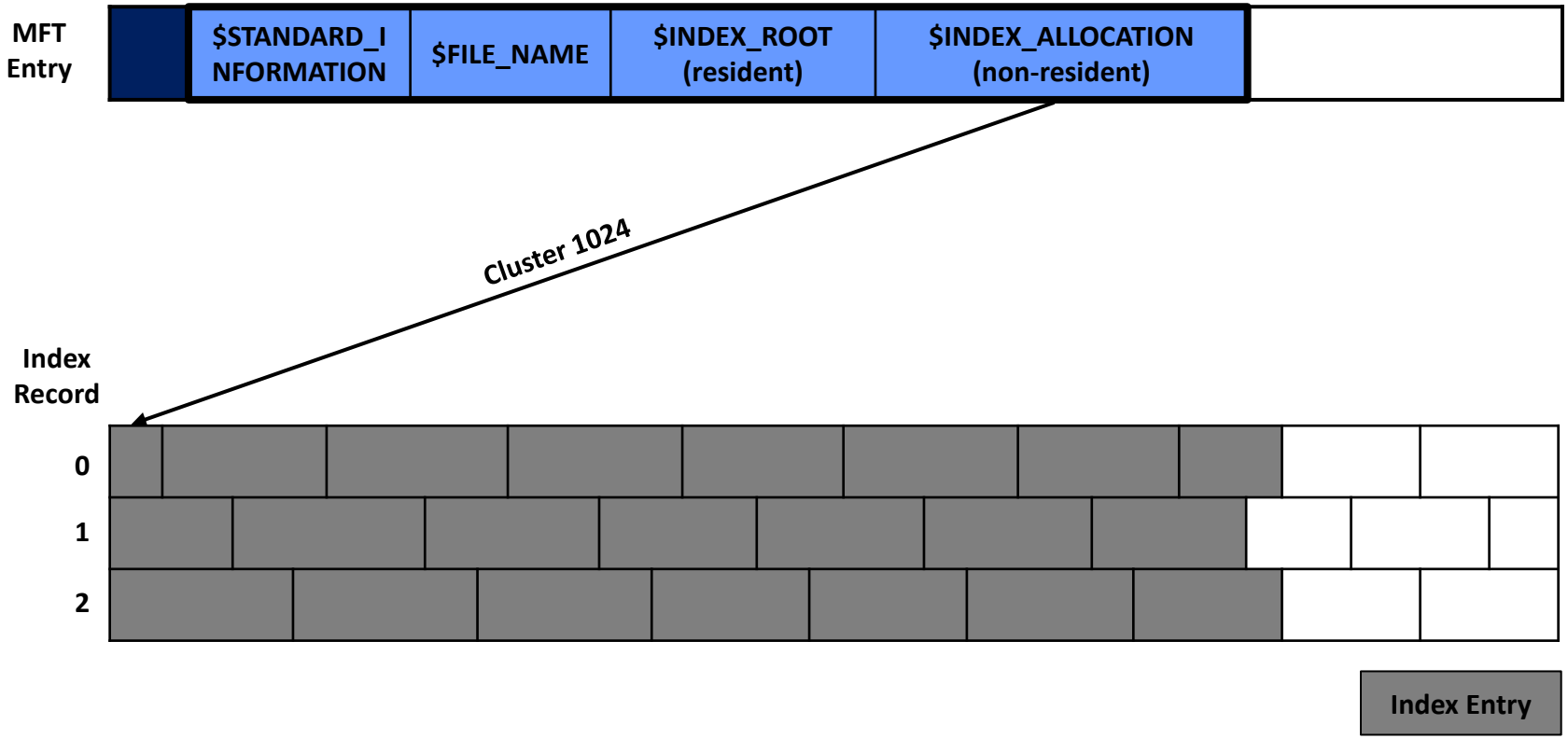
인덱스 구조

- Non-Resident 인덱스 노드



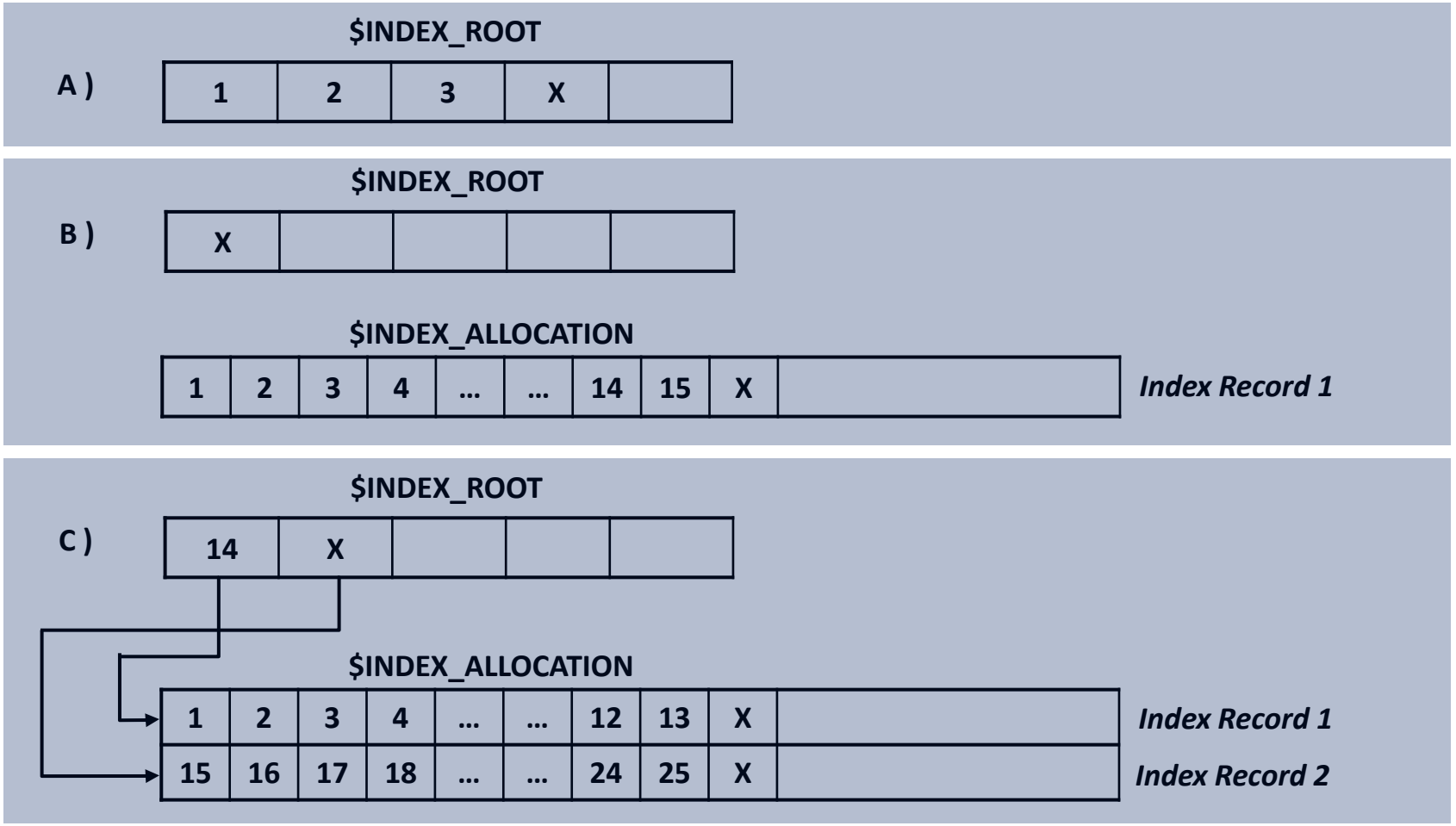
인덱스 구조

- Non-Resident 인덱스 노드



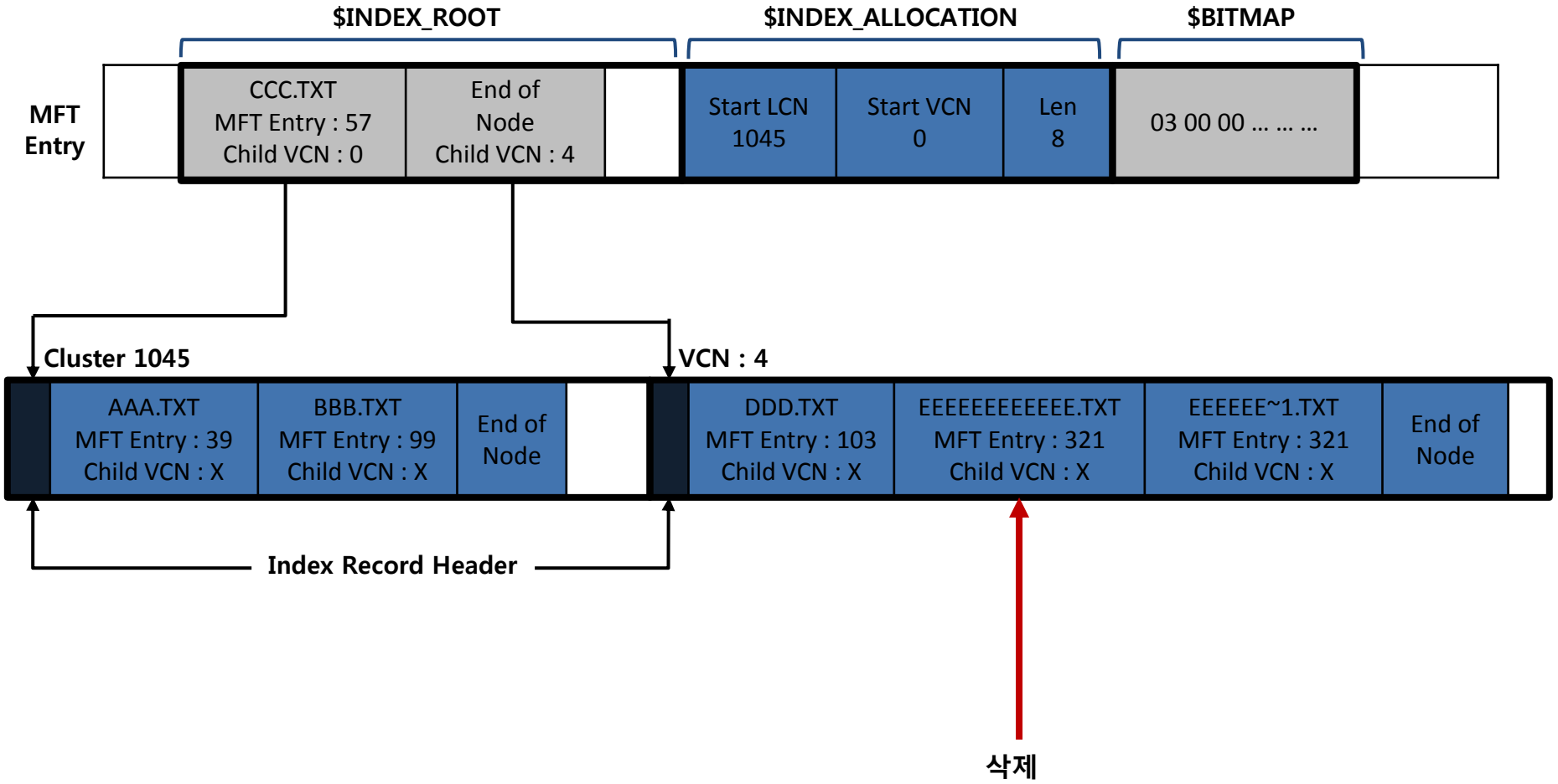
인덱스 구조

- Non-Resident 인덱스 노드

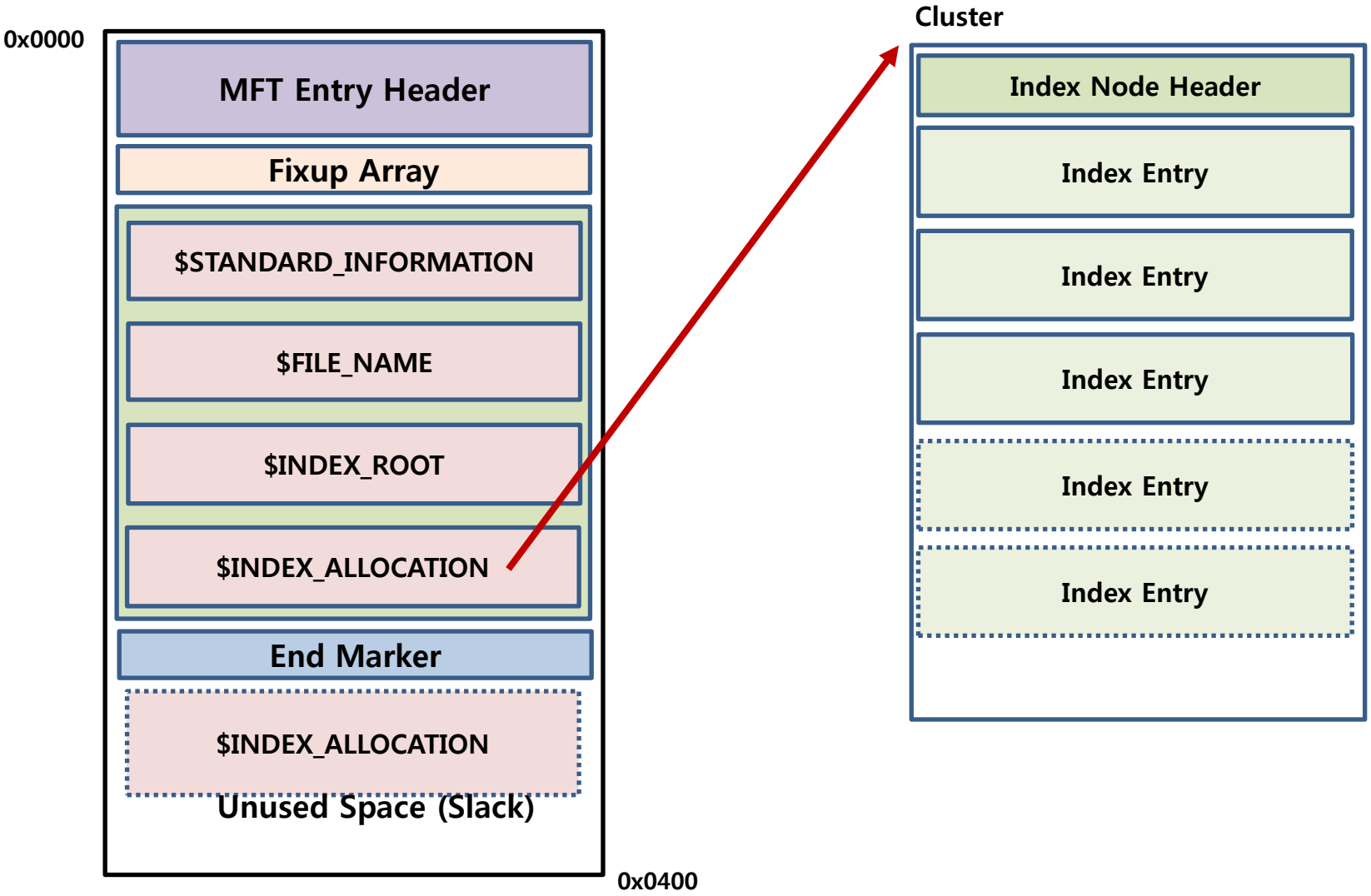


인덱스 구조

- Non-Resident 인덱스 노드



인덱스 구조


0x0000
MFT Entry Header
Fixup Array
$STANDARD_INFORMATION
$FILE_NAME
$INDEX_ROOT
$INDEX_ALLOCATION
End Marker
$INDEX_ALLOCATION
Unused Space (Slack)
0x0400
Cluster
Index Node Header
Index Entry
Index Entry
Index Entry
Index Entry
Index Entry

INDX 슬랙 데이터

- **$I30**
  - 디렉터리 내의 삭제된 파일 목록

그래서...?

어쩌라고

- **MFT 슬랙 분석 도구 ← $MFT 파일만 입력으로 받음**
  - MFT 레코드 분석 기능
  - 전체 경로 출력 기능 ($FILE_NAME의 부모 파일 참조 주소 활용)
  - MFT 슬랙 출력 (하단 도킹 패인 활용)
    - ✓ 파일 : 16진수 데이터 출력
    - ✓ 디렉터리 : 각 파일의 $FILE_NAME 속성 출력
  - 전체 MFT 슬랙 모음 기능

- **INDX 슬랙 분석 도구 ← 활성 상태에서 분석 혹은 파일시스템 이미지 분석**
  - 각 디렉터리 별 슬랙에 존재하는 $FILE_NAME 속성 출력
  - 디렉터리 경로를 옵션으로 받거나 전체 디렉터리의 슬랙 데이터를 csv로 출력

질문 및 답변