

NTFS - Part 3 (보고서)

날짜	@2024년 2월 13일
사람	ohnahee

Chapter 8. 파일시스템 분석

8.1. 파일시스템이란

[데이터 분류](#)

[필수 데이터와 부가 데이터](#)

[분류를 통한 분석](#)

8.2. 파일시스템 범주

[분석 기술](#)

8.3. 내용 범주

[전반적 정보](#)

[분석 기술들](#)

[영구 삭제 기술](#)

8.4. 메타데이터 범주

[전반적정보](#)

[분석기술](#)

8.5. 파일 이름 범주

[전반적인 정보](#)

[분석 기술](#)

[영구 삭제 기술](#)

8.6. 응용프로그램 범주

[파일시스템 저널](#)

8.7. 응용프로그램 수준 검색 기술

[응용프로그램 기반 파일 복구 \(데이터 흔적찾기\)](#)

[파일타입 정렬](#)

8.8. 특정 파일 시스템들

[요약](#)

Chapter 9. FAT 파일시스템

9.1. 소개

9.2. 파일시스템 범주

[전체적인 개념](#)

[분석 기술과 고려사항](#)

9.3. 내용범주

[할당 알고리즘](#)

[분석 기술](#)

[분석고려사항](#)

[분석 시나리오](#)

9.4. 메타데이터 범주

[분석 기술](#)

[분석 고려사항](#)

[분석시나리오](#)

9.5. 파일이름범주

[할당알고리즘](#)

[분석기술들](#)

[분석고려사항](#)

[분석시나리오 \(2가지\)](#)

9.6. 큰 그림

[파일 할당의 예](#)

[파일 삭제 예제](#)

9.7. 다른주제

[파일 복구](#)

타입결정

일관성 검사

Chapter 10. FAT 데이터 구조

10.1. 부트섹터

10.2. FAT FSINFO

10.3. FAT 영역

10.4. 디렉토리 엔트리

10.5. 긴 파일명 디렉토리 엔트리

Chapter 8. 파일시스템 분석

8.1. 파일시스템이란

- 데이터들을 장기적으로 저장하고 빠르게 검색할 수 있도록 하는 시스템

데이터 분류

- 여러개의 파일 시스템을 조사하고 기본적인 참조모델을 이용하여 타입들을 쉽게 비교할 수 있다
- 모델은 5가지로 분류된다

파일시스템범주

- 전체 파일 시스템의 정보
- 특정 파일 시스템을 위한 지도

내용범주

- 실제 내용을 구성하는 데이터
- 데이터 유닛

메타데이터 범주

- 파일을 설명하는 데이터
- 저장위치, 크기, 접근 및 수정시간, 접근 제어 등을 표시
- 파일 이름과 내용은 표시하지 않는다

파일이름 범주

- 파일명이 할당된 데이터
- 디렉토리 내용에 위치

응용프로그램 범

- 특별한 기능을 제공하는 데이터 포함
- 파일을 읽고 쓰는데는 필요가 없지만 통계나 저널과 같이 특별한 기능에 사용

필수 데이터와 부가 데이터

필수데이터

- 파일을 저장하고 가져오는 데 필요한 것들
- 파일 내용, 파일명을 저장하는 주소, 메타데이터 구조에서 이름으로 연결하는 포인터

부가데이터

- 단지 편리함을 위해 존재하는 데이터
- 접근시간, 허용권

*필수데이터가 비신뢰적이면 읽어오는데 문제가 발생

부가데이터는 상관 X

분류를 통한 분석

EX) JPG 확장자 파일 찾을 경우 → 파일 이름 범주 분석

특정값으로 파일을 찾을 경우 → 메타데이터 범주 분석

8.2. 파일시스템 범주

- 어떻게 이 파일 시스템이 고유한지, 중요한 데이터들이 어디에 위치했는지를 구분
- 파일시스템 첫번째 섹터 표준 데이터 구조체에 저장
- 이 단계에서는 다른 범주의 데이터 구조체 위치를 찾아야 하므로 파일시스템 분석의 모든 유형들을 필요로 함

- 이 범주의 특정 데이터를 손실하거나 잃어버리면 추가 분석이 어려울 수 있다

분석 기술

- 무결성 검사
 - 파일시스템이 있는 볼륨 크기와 파일시스템 크기를 비교
 - 두 개를 비교했을 때 볼륨 크기가 파일 시스템의 크기보다 더 크다면 볼륨 슬랙이라는 공간이 생긴것
 - 이 공간에는 데이터를 숨길 수 있다
- TSK 에는 파일시스템 범주 데이터를 포함하는 `fsstat` 도구가 있음

8.3. 내용 범주

- 데이터 저장을 위한 파일과 디렉토리를 할당하는 저장공간

전반적 정보



데이터 유닛 주소 지정방법

데이터 유닛 할당 방법

파일시스템이 손상된 데이터 유닛 처리 방법

논리적 파일 시스템 주소

- 한 섹터는 물리적/상대적(논리적) 주소가 존재한다
- 논리적 볼륨 주소 내에서 데이터 유닛을 구성하기 위해 , 연속적인 섹터를 그룹화 하기위해 논리적 파일 시스템 주소를 부여한다

데이터 유닛 할당 정책

- 운영체제마다 데이터 유닛 할당 정책이 다름

1. 적용

- 파일 시스템의 첫 데이터 유닛부터 검색하여 사용가능한 데이터 유닛을 찾는 것

2. 다음 적용

- 마지막 할당 그 이후부터 할당하는 것

3. 자동 맞춤

- 파일 데이터 크기에 맞는 연속적인 데이터 유닛 검

손상된 데이터 유닛

- 손상된 데이터 유닛을 표시할 수 있는 기능이 파일시스템에 존재
- 표시하고 나서 이 공간에 데이터를 할당하지 않음
- 하드디스크 스스로 불량섹터를 감지하고, 교체한하기 때문에 표시 기능이 필요하지는 않음
 - 기능을 약용할 수 있으니 수집도구의 불량섹터 보고, 손상된 리스트 비교를 통해 더해진 섹터를 찾을 필요가 있다

분석 기술들



내용 범주 데이터 데이터 분석 방법

데이터 유닛 보기

- 특정 파일에 할당, 특별한 의미가 있는 증거의 주소를 알아내려 할 때 사용
- 논리적 파일시스템의 주소에 접근해서 도구를 이용해 바이트나 섹터 주소를 계산
- 그 도구는 해당 위치를 파악 해 데이터를 읽음
- hexa 편집기
- TSK의 dcat 도구

논리적 파일 시스템 수준 검색

- 파일의 내용만 알고 위치를 모를 때
- 특정 문구나 특정 파일 헤더를 일일이 찾아다녀야 한다
- 비할당, 할당 공간을 모두 추출하여 비트맵을 만들어 비할당 공간만 찾아다니며 지워진 팜리를 복구한다

데이터 유닛 할당순서

- 할당정책을 결정하는 운영체제 조사

일관성 검사

- 모든 데이터 범주를 분석하는 기술
- 정확히 하나의 할당된 메타데이터 엔트리 지점을 정확히 갖는지 검증
- 해당하는 메타 데이터 엔트리가 없는 데이터 유닛을 고아 데이터 유닛이라고 부름

영구 삭제 기술

- 파일에 할당된 데이터 유닛과 모든 비할당 데이터 유닛에 0이나 난수를 채워두는 것
- 응용프로그램 수준으로 영구 삭제를 하기는 힘들다

→ 어차피 운영체제가 0이나 난수를 써야하는데 시간차가 존재함

- 이 기술이 쓰이면 증거를 찾기 힘들다
 - 영구 삭제 도구를 찾고 접근했는지 알아내야함
 - 복구가 불가능 할 시 임시파일 복사본을 대체로 찾는다

8.4. 메타데이터 범주

- 데이터를 부가 설명해주는 데이터가 존재하는 곳
- 메타 데이터와 파일명 분석을 구분하지 않고 두 범주를 통해 분석한다

전반적정보



주소지정방법

슬랙공간

지워진 파일 복구

압축 파일

암호화 파일

논리적 파일 주소

- 파일이 가지는 데이터 유닛 = 논리적 파일 주소

논리적 볼륨 주소 ≠ 논리적 파일 시스템 주소 ≠ 논리적 파일 주소

슬랙공간

- 파일 크기와 그 파일이 할당된 데이터 유닛 사이의 갭
- 컴퓨터에서의 슬랙 공간은 두가지
 - 파일의 끝, 파일이 끝나는 섹터의 끝 사이
 - 보통 운영체제는 0으로 섹터를 채움
 - 파일 내용을 포함하지 않은 섹터
 - 비사용 섹터로서 이전에 삭제하나 파일의 데이터가 남아있을 수 있다

메타데이터 기반 파일 복구

- 지워진 파일을 복구하기 위한 방법
 - 메타데이터 분석
 - 응용프로그램 분석

압축 및 스파스 파일

- 일부 파일 시스템들은 데이터를 압축된 형식으로 저장하여 좀 더 적은 데이터 유닛들이 디스크를 채우도록 함
- 최소한 3개의 수준을 가지고 있음
- 압축 수준
 - 높은 수준 : 한 파일 형식의 내부 데이터를 압축
 - 중간 수준 : 외부 프로그램이 전체 파일을 압축해서 새로운 파일 생성
 - 낮은 수준 : 파일시스템이 데이터를 압축하는 것

암호화 파일

- 허가되지 않은 사용자가 파일에 접근하는 것을 막기 위해 내용을 암호화 하여 저장하는 것
- 파일 내용이 아닌 접근 시간같은 부가 데이터는 암호화가 불가능
- 전체 볼륨을 암호화 시 파일 내용 암호화 가능

분석기술



파일내용 보기

데이터 검색

지워진 파일 위치 찾기

메타 데이터 검색

- 메타데이터를 분석하면 특정 메타데이터 구조체에서 가리키는 파일명을 발견할 수 있다
- TSK istat 도구 → 메타데이터의 데이터 구조체 값들을 보여줌

논리적 파일 보기

- 해당 파일의 메타 데이터를 찾으면 그 파일에 할당된 데이터 유닛을 읽어서 내용 확인 가능
- 파일이 항상 데이터 유닛을 다 채우지는 않기 때문에 슬랙공간이 생기는 것을 주의
- TSK istat 도구는 메타데이터 구조체에 할당된 데이터 유닛의 내용을 보여줌
 - -s 옵션은 슬랙공간
 - -r 옵션은 지워진 파일 복구 시도

논리적 파일 찾기

- 특정 키워드를 가지는 파일 찾기
- 앞에서 배운 논리적 파일시스템 검색은 데이터 유닛이 연속적이지 않으면 검색이 어려웠지만 메타데이터 이용시 한 세트로 검색하게 되어 가능함
- 오직 할당된 데이터만 논리적 파일 주소를 갖는다
- 분석도구가 슬랙공간을 검색하는지도 파악해야함
 - DFTF 테스트 이미지를 통해 확인

비할당 메타데이터 분석

- 지워진 내용을 다시 찾는 경우
- TSK ils 도구
 - 비할당 데이터 구조체 목록을 보여줌

메타데이터 속성 검색 및 정렬

- 특정 시간을 기준으로 찾는 경우 부가적인 정보를 이용할 수 있다 이 때 메타데이터 이용
- 시간은 변경오디기 쉽지만 실마리를 제공하기도 함

- TSK mactome 도구
 - 파일들의 타임라인을 만드는데 유용
 - 첫번째 열 : 시간 스탬프
 - 두번째 열 : 파일 크기
 - 세번째 열 : 내용 수정 시간/내용접근시간/메타데이터 변경시간
 - *실제 정보는 더 많음
- 데이터 유닛을 발견했다면 데이터 유닛주소로 메타데이터 엔트리를 검색할 수 있다
 - 이걸 통해서 같은 파일의 데이터 유닛들의 주소를 알 수 있다

데이터 구조체 할당 순서

- 두 엔트리의 시간 차이를 아는 것은 운영체제마다 너무 다르기 때문에 어려움

무결성 검사

- 무결성 여부를 판단하는 필수 데이터
 - 데이터 유닛 주소
 - 크기
 - 각 메타데이터 엔트리의 할당 상태
- 파일 이름 범주 데이터를 이용하는 방법도 있음
 - 할당된 모든 디렉토리 엔트리들이 그것을 가리키는 할당된 이름이 있는지 검증하는 것

영구 삭제 기술 (메타데이터 범주에서)

- 영구 삭제 시간, 크기, 데이터 유닛주소들에 0이나 난수를 채워 넣음

8.5. 파일 이름 범주

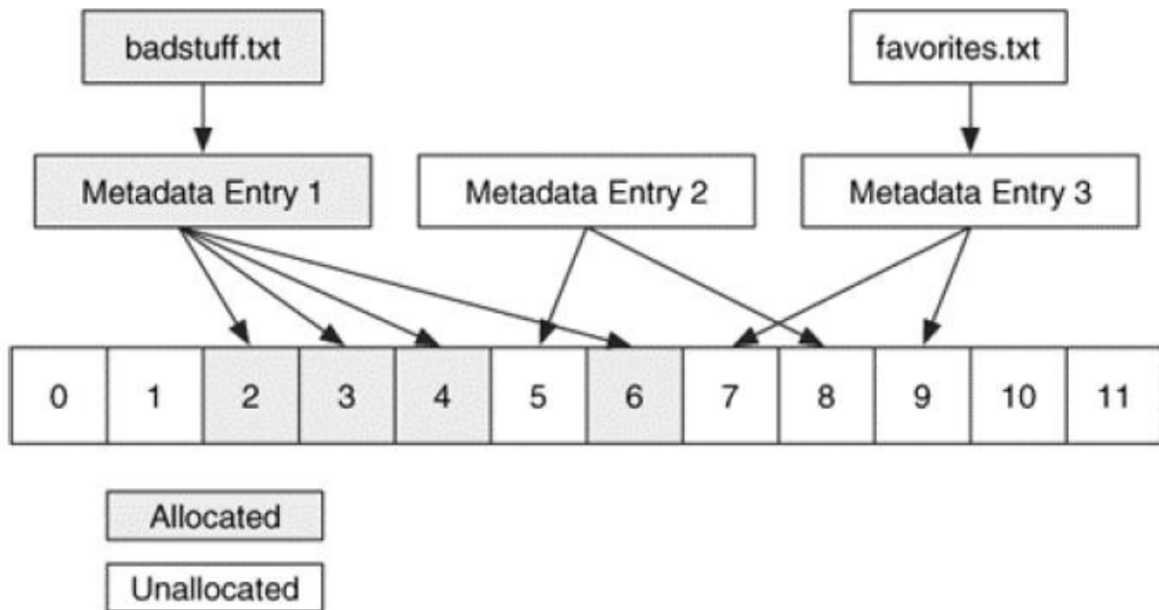
전반적인 정보



파일 이름 범주의 전반적 개념
이름에 기초한 파일 복구 방법

파일 이름 기반 복구

- 메타데이터 기반 복구로 파일명을 복구하기 위해 **삭제된 파일명**과 **메타데이터 주소**를 사용함



두개의 파일명과 3개의 메타데이터 엔트리가 있는 예

분석 기술

*파일 이름 범주에서 수행될 수 있는 분석 기술

파일명 목록 작성

- 파일명을 할당하기 위함
- 파일과 디렉토리 이름을 목록화 함
 - 이름, 경로, 확장자 기반으로 증거를 검색할 때 사용
- 더 많은 정보를 얻기 위해 파일의 메타데이터 주소를 사용함
- 루트 디렉토리 위치를 파악해야함
- TSK fls 도구
 - 할당되고 지워진 파일명을 목록화 해준다

파일명 검색

- 검색하는 과정은 파일명을 목록화 하는 것과 비슷
- 또 다른 검색은 메타데이터 엔트리에 할당된 파일명을 검색하는 것
- TSK ffind 도구가 이러한 작업을 도와줌

일관성 검사

- 모든 이름들이 메타데이터 구조체를 가르키는지 확인
- 한 파일에 여러 파일명이 있는 경우도 있음
 - 이때는 메타데이터 구조체에 여러 파일명이 가리키고 있을수도 있음

영구 삭제 기술

- 영구 삭제도구는 구조체로부터 이름과 메타데이터 주소를 지운다
- 과정
 - 1) 새로운 파일 이름을 삭제되는 구조체에 덮어 씌움
 - 2) 이름 목록을 인지하고 기존의 파일명으로 덮어씌운다

8.6. 응용프로그램 범주

파일시스템 저널

- 컴퓨터가 갑자기 꺼지는 경우 (충돌) 시스템은 불안정 상태가 된다
- 이러한 상황을 위해 운영체제는 파일시스템을 점검하고 잃어버린 포인터와 손상된 흔적을 찾는 프로그램을 실행한다
- 점검이 잘 되도록 하기 위해서 파일시스템은 저널을 구성한다
- 시스템이 손상되면 이 저널을 읽고 충돌 전 완성되지 않은 엔트리 위치를 확인한다

8.7. 응용프로그램 수준 검색 기술



응용프로그램 계층과 분석을 위해 할당된 파일을 구성하고 삭제한 파일을 복구하는데 사용되는 기술 소개

응용프로그램 기반 파일 복구 (데이터 흔적찾기)

- 데이터 흔적
 - 데이터 한 묶음을 알려진 파일 유형의 시작과 끝에 맞는 시그니처를 검색하는 과정
 - 이 결과는 시그니처 중 하나를 포함하는 파일들의 집합
 - 보통 비할당 공간상에서 수행
 - 수사관은 메타데이터 구조체와 그것들에 연결이 없는 파일들을 복구
 - EX) jpeg 이미지를 찾기위해 비할당 공간을 추출하고 이 이미지의 헤더를 찾는 카빙 도구를 실행해 헤더와 footer 의 데이터 추출

foremost

- 분석도구
- 미가공 파일 시스템이나 디스크 이미지를 각 파일 타입의 시그니처 항목이 담긴 설정 파일을 기반으로 분석
- 알려진 헤더값
- 파일 최대 크기
- 헤더값의 대소문자 여부
- 파일타입의 확장자
- 추가적인 파일 마지막값

파일타입 정렬

- 파일 시스템에 파일을 구성하는데 사용
- 모든 그림들과 실행파일을 한번에 그룹화 가능

8.8 특정 파일 시스템들

- 나머지 장에서는 FAT, NTFS, Ext2/Ext3, UFS 파일 시스템의 저장 방법을 조사하고 다섯가지 범주로 분류하여 설명할 것

요약

- 디지털 조사의 거의 모든 증거는 파일 시스템 분석으로 찾기 가능
- 찾으려는 데이터 타입에 따라 어떤 분석기술을 사용해야하는지 생각

Chapter 9. FAT 파일시스템

9.1. 소개

- File Allocation Table
- 일반 운영체제에서 볼 수 있는 가장 간단한 파일 시스템
- 각 파일과 디렉토리를 데이터 구조체에 할당하는 것 → 디렉토리 엔트리
- 파일 다음 클러스터와 클러스터의 할당 상태를 식별하기 위해 사용
 - 내용과 메타 데이터 범주 모두가 해당 구조체에 포함될 수 있다



- 예약된 영역
 - 범주 데이터 포함
- FAT 영역
 - 주 FAT / 부 FAT / 백업 FAT
- 데이터 영역
 - 파일, 디렉토리 내용을 저장하는 클러스터

9.2. 파일시스템 범주



FAT 이 데이터를 저장하는 위치
그 데이터를 분석하는 방법

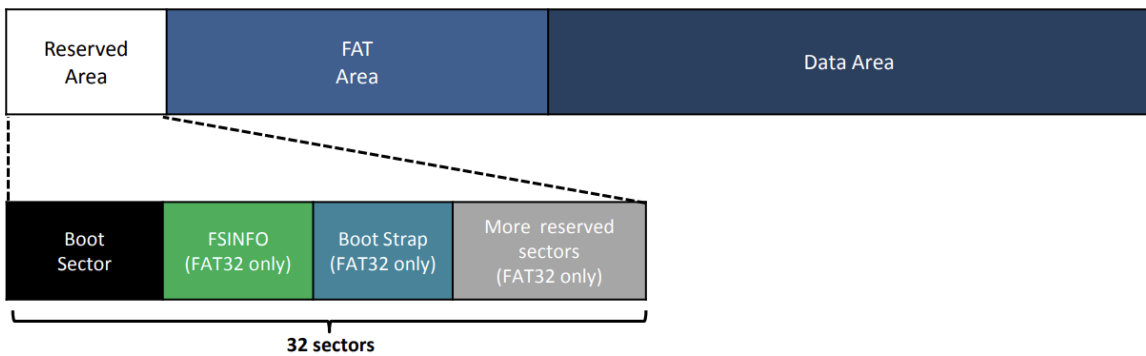
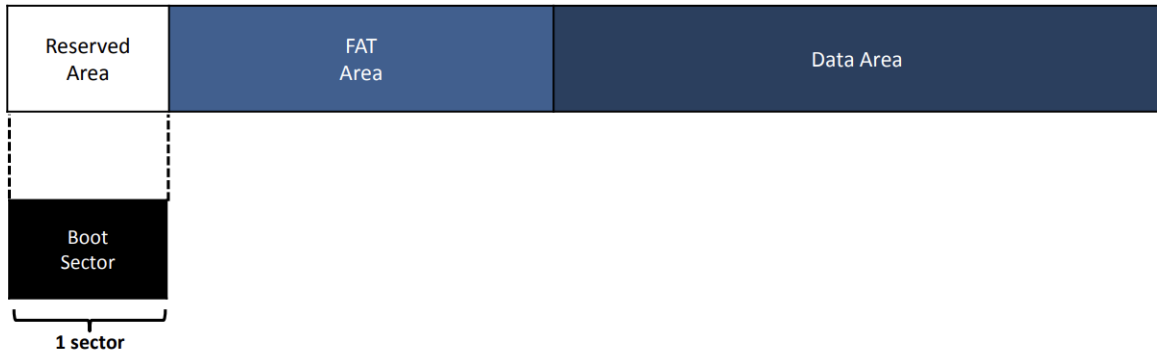
- 파일 시스템 범주 데이터
 - 파일시스템 설명 용도
 - 그 데이터들은 다른 데이터 구조체들을 찾는 데 사용

전체적인 개념

- 파일 시스템 범주는 부트섹터 데이터 구조체에서 확인 가능
- 부트 섹터
 - 볼륨의 첫 번째 섹터 (예약영역) 에 위치
 - MS에서는 BPB에 속한 첫 번째 섹터에서 데이터 일부를 참조하지만
 - 이 책에서는 부트 섹터라는 용어 사용
 - 부트섹터는 모든 범주에 속하는 데이터를 포함한다

필수 부트섹터 데이터

- FAT 파일 시스템 분석 시 3개의 물리적인 레이아웃 영역 위치를 알아야 함
- 예약된 영역은 섹터 0에서 시작하고 그 크기는 부트섹터가 정한다
- FAT 14, 16 에서 예약된 영역은 1섹터 크기지만 FAT 32에서는 더 많은 섹터를 가짐



부가 부트 섹터 데이터

- 레이아웃 정보 외에도 부트섹터는 많은 부가 데이터를 가지고 있음
- 단지 편리함을 위해 존재하는 데이터도 있기에 부정확 할 수도 있음
- 마지막 식별 레이블은 파일 시스템을 언제 생성했는지 구분하도록 하는 7개 문자로 구성된 볼륨 레이블 스트링
 - 그 볼륨 레이블은 루트 디렉토리에 저장됨

부트코드

- FAT 파일 시스템 부트코드는 데이터 구조체와 밀접한 관련이 있음

분석 기술과 고려사항

- 부트 섹터 정보를 이용해서 예약, FAT, 데이터 영역의 위치를 계산

- 범주 데이터들은 파일 시스템에 대한 구조적인 데이터 제공
- FAT 32는 예약된 공간에 많은 섹터를 할당하는데 실제 백업 부트 섹터, FSINFO 구조체 주 부트섹터 등에서 사용하는 공간은 작음
 - 이 곳에도 데이터 숨김 가능
- 파일시스템과 마지막 볼륨 마지막 사이의 공간인 볼륨 슬랙에도 데이터 숨김 가능

9.3. 내용범주

- 파일이나 디렉토리 내용을 구성하는 데이터를 포함
- FAT 파일 시스템은 그 데이터 유닛을 위해 클러스터라는 용어 사용
 - 클러스터의 섹터 개수는 1, 2, 4, 8.... 2의 제곱이어야 함
 - 각 클러스터는 주소가 있으며 첫 클러스터 주소는 2
 - 0, 1 주소를 가진 클러스터는 존재하지 않음
 - 클러스터들은 파일 시스템 3개 영역 중 마지막에 있는 데이터 영역 부분에 모두 할당 됨

첫번째 클러스터 찾기

- FAT 32 에서 데이터 영역 첫 번째 섹터는 클러스터 2
 - EX) 2,048 바이트 클러스터를 갖는 파일 시스템과 섹터 1,224 에서 시작하는 데이터 영역을 생각한다면
그 클러스터 2의 섹터 주소는 1,224
클러스터 3 섹터 주소는 1,228

클러스터와 섹터 주소

- 클러스터 주소를 데이터 영역에서 사용됨
- 예약된 영역과 FAT영역에서는 다른 주소 체계를 사용하거나 일반적인 가장 낮은 수준의 섹터 주소 체계 (논리적 볼륨 주소) 를 사용한다
- TSK를 포함한 많은 도구에서 보통 한 가지의 주소 체계를 이용해서 사용자에게 보여줌 그렇기 때문에 데이터 영역에 서만 사용하는 클러스터 주소로 통일할 수는 없다 따라서 한가지의 체계로 바꾸어 주어야한다
 - 클러스터 주소를 논리적 볼륨주소와 같은 주소로 변환해야함

클러스터 할당상태

- FAT 구조체를 통해 확인 가능
- 보통 2개의 FAT 복사본이 있고 하나는 파일 시스템의 예약된 영역 이후 시작
- 테이블 엔트리가 0이면 클러스터가 할당되지 않았다는 것

할당 알고리즘

- 윈도우 98, XP에서는 클러스터 할당 알고리즘으로 다음적용을 사용
- 할당 가능한 비 할당 클러스터를 찾기 위해 운영체제는 0값을 가지는 엔트리 확인
- 클러스터를 비할당 하고 싶다면 엔트리 값을 0으로 바꾸면 도미

분석 기술

- 특정 유닛을 찾고 그 할당 상태를 결정해서 그 내용으로 수행하는 것
- 데이터 영역에 있는 영역 위치는 섹터 주소를 가지고 찾고 데이터 영역 내의 위치는 클러스터 주소와 섹터 주소 중 하나를 이용할 수 있다
- 만약 모든 비할당 클러스터의 내용들을 추출하기를 원하면 FAT을 읽어서 테이블 값에 0을 가지는 각 클러스터를 추출하면 됨

분석고려사항

- FAT 파일 시스템을 사용하는 많은 디스크들이 하드웨어 수준에서 불량 섹터를 처리하고 운영체제는 그것들을 확인하지 않음
 - 불량클러스터 따로 조사 필요
- 데이터 영역 끝에 클러스터 일부가 아닌 남은 섹터들이 있을 수 있기 때문에 클러스터 크기의 배수가 아니며 이런 곳에 데이터를 숨기거나 이전 파일 시스템의 데이터를 포함할 수 있음
- 비사용 섹터 확인
 - $(\text{전체 섹터수} - \text{클러스터 2의 섹터 주소}) / \text{클러스터 섹터 수}$
- 데이터 숨김이 가능한 다른 곳
 - FAT 구조체 마지막 엔트리와 백업본 시작 사이
 - 백업 FAT 마지막 엔트리와 데이터 영역 시작

분석 시나리오

조사관이 FAT 16 파일 시스템을 조사해야하고 클러스터 812의 첫 섹터를 확인해야 한다고 가정할 경우

1. 섹터 0 → 부트 섹터 찾기
2. 이후 6개의 예약된 영역을 지나 2개의 FAT을 확인 가능
 - 각 FAT에는 249 섹터가 있다
 - 각 클러스터는 32 섹터이고 루트 디렉토리에는 512개 엔트리가 있다
3. 0-5 : 예약된 영역
 - 6-254 : FAT1
 - 255-503 : FAT2
 - 504-535 : 루트 디렉토리 → 512 바이트가 1섹터인데 각 엔트리는 32바이트이므로 총 32개의 섹터가 된다
4. 클러스터 812의 위치 찾기 가능

9.4. 메타데이터 범주

- 파일또는 디렉토리 내용이 저장된 위치, 날짜, 시간 그리고 허가권 같은 데이터가 저장되어있음
- FAT 파일 시스템은 이러한 정보를 디렉토리 엔트리에 저장함
- 파일 시스템은 FAT 구조체를 파일, 디렉토리 레이아웃에 대한 메타데이터 정보를 저장하기 위해 사용

디렉토리 엔트리

- 파일과 디렉토리마다 할당된 구조체
- 32바이트
- 파일 속성, 크기, 클러스터, 날짜, 시간 정보가 들어있음
- 파일명을 저장하기 때문에 메타데이터 뿐 아니라 파일 이름 범주 역할도 함
- 클러스터처럼 고유번호를 사용하지 않음 → 파일의 전체 이름 사용
- 각 파일과 디렉토리에 할당되는 속성은 4개의 부가 속성이 존재
 - 읽기전용
 - 숨김속성
 - 시스템속성
 - 아카이브 속성
- 각 디렉토리 엔트리에는 생성날짜, 수정날짜, 접근날짜 3개가 존재
 - 생성날짜 : 추가 정보로서 0.1초 단위로 저장
 - 접근날짜 : 추가 정보로서 날짜 단위로 저장
 - 수정날짜 : 명세에 의해 요구되는 정보로 2초 단위로 저장

클러스터 연결

- FAT 엔트리가 0이 아니면 할당된 상태임을 알 수 있음
- EX) 클러스터 40, 41, 45에 위치한 파일을 생각하자
 디렉토리 엔트리에서 시작 클러스터 필드 40을 먼저 조사
 클러스터 40에 해당하는 FAT 엔트리는 다음 클러스터인 41을 포함할 것
 파일 크기는 아직 더 많은 클러스터가 존재한다는 것을 보여줄 수 있음
 40 → 41 → 45를 이런식으로 조사하면 마지막 클러스터에서 FAT엔트리는 EOF(End of File) 을 저장하고 있을 것
 이러한 클러스터 순서를 클러스터 연결이라고 부름

디렉토리

- 새로운 디렉토리가 생성될 때 클러스터를 할당하고 0으로 초기화함
- 크기 필드를 사용하지 않으면 항상 0이 되어야 함
- EOF 가 발견될 때 까지 FAT 구조체 클러스터를 따라간다

디렉토리 엔트리 구조

- 디렉토리를 찾아가기 위한 방법은 할당하려는 파일이나 디렉토리의 전체 이름을 사용하는 것
 - 두가지 문제
 - 수사관은 보통 비할당된 엔트리를 찾음 → 비할당할 때 이름의 첫글자는 지워지므로 고유이름이 충돌
 - 디렉토리가 지워져 그 엔트리가 재할당 → 지워진 디렉토리에 파일과 디렉토리에 대한 포인터가 없어 주소를 찾기 못함

할당 알고리즘

- 메타데이터 속 할당된 데이터는 두 유형으로 나뉨

1. 디렉토리 엔트리 할당

- 한 파일이 삭제될때 디렉토리 엔트리의 첫 바이트는 [0xe5]로 설정
- 파일에 할당된 클러스터를 비할당하는 방법은 FAT 엔트리를 0으로 설정하는 것
- 파일의 다음 클러스터 정보는 FAT 엔트리에 존재

2. 시간 값 업데이트

- 시간 값에는 값, 생성, 접근, 수정 시간이 존재
 - 변경되기 쉽고 잘못된 정보를 가지고 있을 수 있음

분석 기술

- 디렉토리 엔트리를 통해 파일의 시작을 알고 다른 클러스터 위치를 확인하기 위해 FAT 구조체를 사용한다

분석 고려사항

- 마지막 접근 및 생성시간들이 변경되기 쉽기때문에 파일을 열기 전 접근시간을 저장하여 이후에 변하지 않도록 해야함
- 윈도우 내 DEFRAG 유틸리티는 디렉토리들을 압축하고 사용하지 않은 디렉토리들을 제거하지 때문에 이 유틸리티가 실행된이후에는 복구가 어렵다

분석시나리오

파일시스템의 생성날짜

- 분석도구를 통해 디렉토리 엔트리를 목록화

삭제된 디렉토리 탐색

- 비할당 공간 추출
 - TSK dls 사용

9.5. 파일이름범주



FAT 이 긴 파일명을 어떻게 처리하는지에 대한 이해

- 파일이름범주 분석은 메타 구조체를 이용하여 파일명을 확인시켜줌
- FAT은 파일명주소와 메타데이터 주소를 구분하지않음 이미 파일명이 메타주소로 사용되기 때문에 이미 메타 데이터 범주에서 파일과 디렉토리 이름의 기본 내용을 설명 완료
- 디렉토리 엔트리 구조체는 파일명으로 8문자, 확장자로 3문자를 이용해 파일명을 저장함
- 파일명이 8문자보다 길거나 이름에 특별한 값이 있다면 디렉토리에 LFN(Long File Name) 타입 추가

- LFN이 있는 파일들 또한 SFN (Short File Name) 디렉토리를 가짐
 - LFN 엔트리는 시간, 크기, 시작 클러스터 정보가 들어있지 않기 때문
- FIs 도구로 분석 가능

할당알고리즘

- 파일명은 메타데이터와 같은 데이터구조체에 저장
- 파일 이름 범주의 할당 알고리즘도 메타데이터 범주와 같음
- 차이점은 **LFN이 SFN의 앞에 위치**해야 한다는 것
- 운영체제는 모든 엔트리들이 들어갈 공간이 있는지 검색
- '첫 번째 적용'에서는 비할당 공간 주변에 다른 비할당 공간이 없으면 건너뛸 수도 있음
- 삭제루틴은 이전에 언급한 것과 같고, 첫 바이트는 0xe5로 교체된다

분석기술들

- 파일 이름범주 분석은 특정 **파일명을 찾는 것**
- FAT 버전에 따라 루트디렉토리의 위치가 다르기 때문에 이 루트디렉토리의 위치를 먼저 파악해야한다 (부트섹터에서)
- 구조체 속성이 LFN 엔트리로 설정되어 있으면 단계적으로 디렉토리를 처리하고 그 내용들이 저장되고 다음 엔트리가 조사됨 이 과정은 일치하는 체크섬을 찾을 때까지 계속되며 엔트리의 할당상태는 첫 바이트를 이용하여 결정된다
- 메타데이터는 디렉토리 엔트리에 위치하기 때문에 확인하기 쉽고, 파일이 삭제되어도 동기가 유지

분석고려사항

- 파일명과 메타데이터는 같은 구조체에 존재하기 때문에 복구하고자하는 파일명이 비할당 엔트리에 항상 존재함
- 만약 LFN을 사용한다면 SFN의 이름의 첫 바이트가 바뀌어도 LFN은 존재하고 있다는 의미
- 만약 파일명을 찾기위해 검색도구를 통해 키워드 검색 동작방법을 사용 시, 긴 파일명에 유니코드(Unicode) 버전을 사용하는지 확인해야 함
 - SFN은 ASCII로 저장되지만 LFN엔트리들은 유니코드로 저장된다
 - LFN엔트리들이 여러 작은 조각으로 나뉘어질 수 있다

분석시나리오 (2가지)

파일명검색

- 도구에서 파일명검색이 이루어지는 것에 대한 이해
- SFN으로 찾을 시 아스키로 이루어진 수정된 이름 검색
- LFN 으로 찾으려면 이름을 묶음으로 변환해주어야함
 - 그 이후 오탐을 줄이기 위해 가장 긴 문자열부터 검색

9.6. 큰 그림



어떻게 하나의 파일이 생성되고 지워지는지에 대한 이해

FAT 파일 시스템에서 파일의 생성을 확인하기 위해 `dir\file1.dat` 파일이 생성되는 과정을 확인
(`dir`디렉토리는 이미 존재/클러스터는 4096바이트/파일크기는 6000바이트)

파일 할당의 예

1. 볼륨섹터 0에서 부트섹터를 읽고 FAT 구조체와 데이터 영역, 루트디렉토리의 위치를 확인
2. `dir1` 디렉토리를 찾기 위해 루트디렉토리에서 각 디렉토리 엔트리를 해석하고, 이름과 속성을 이용해서 `dir1`을 갖는 엔트리를 찾는다. 그것은 여기서 시작클러스터 90
3. `dir1` 시작클러스터 90의 엔트리를 읽고 비할당 엔트리를 찾음
4. 사용가능한 엔트리를 찾고 `file1.txt` 이름을 써서 할당상태를 결정하고 크기와 현재 시간을 적절한 필드에 적는다
5. 우선 FAT 구조체 안을 검색해서 할당할 엔트리를 찾고, EOF로 설정 (밑에 그림에서 200위치)
6. 이후에 클러스터 200에 데이터 내용을 적는다. 이때 4096바이트를 쓰고나니 1904 바이트가 남는다는 것을 알아림 (두번째 클러스터가 필요함)
7. 두 번째 클러스터를 검색하고 할당 (비할당 클러스터를 찾아둠)
8. 첫 클러스터를 위한 FAT엔트리 클러스터 200은 201을 포함하게 변경되고 201은 EOF를 포함하고 클러스터 201에 나머지 1904바이트를 씀

dir1\file1.txt 파일을 삭제

파일 삭제 예제

1. 볼륨섹터 0에서 부트섹터를 읽고 FAT구조체와 데이터 영역, 루트디렉토리 위치를 확인
2. dir1 디렉토리를 찾기 위해 루트디렉토리에서 각 디렉토리 엔트리를 해석하고, 이름과 속성을 이용해 dir1을 갖는 엔트리를 찾음
3. file1.txt 이름을 갖는 디렉토리 엔트리를 찾기위해 dir1 시작클러스터, 클러스터 90의 내용을 확인한다. 파일의 시작 클러스터 200을 확인할 수 있음
4. 파일에 클러스터 연결을 판단하기 위해 FAT구조체를 확인 그 파일은 클러스터 200과 201에 할당되어있음
5. 클러스터 200, 201의 FAT엔트리를 0으로 설정
6. 첫바이트를 0xe5로 변경해서 file.txt를 디렉토리 엔트리에서 할당을 제거

9.7. 다른주제



어느 범주에서 포함되지 않아 적용 못하는 기술들 설명

파일 복구

- 파일을 삭제할 때 FAT 엔트리를 0으로 설정하고 복구를 위해서는 파일의 시작위치와 크기를 알아야 함
- 클러스터를 읽는 방법
 - 파일의 크기만큼 무조건 읽는 방
 - 데이터 할당 상태를 무시하고 비할당 클러스터만을 읽는 방법

타입결정

- 파일시스템의 클러스터 수를 이용해서 타입구분

일관성 검사

- 파일 시스템 조사 시 손상되거나 숨겨진 데이터가 있는지 검사하는 것

- 손상된 것으로 표시된 엔트리 조사
- 마지막 클러스터의 FAT 엔트리, FAT에 할당된 섹터 끝 사이 공간 조사
- 루트 디렉토리와 그 하위 디렉토리 조사
- 볼륨 레이블로 표시된 디렉토리 엔트리는 시작 클러스터를 갖지 않아야함
- 파일시스템에 볼륨 레이블 엔트리는 오직 1개
- 모두 0이거나 난수 데이터인 디렉토리 엔트리는 영구 삭제 후거나 전에 할당된 엔트리일 수 있음
- NULL 엔트리 조사 필요
 - NULL을 보여주지 않는 운영체제가 많기 때문

Chapter 10. FAT 데이터 구조




FAT을 구성하는 데이터 구조체 분석

데이터 구조체 참

File System Category | FAT Concepts and Analysis

File System Category / FAT Concepts and Analysis from File System Forensic Analysis

 https://flylib.com/books/en/2.48.1/10921_file_system_category.html

10.1. 부트섹터

- 파일시스템의 첫 섹터에 존재
- 여러 파일시스템 범주데이터가 있음
- 볼륨 레이블은 Noname
- FAT 12/16 과 FAT32는 부트섹터의 구조가 다름
 - 첫 36비트는 동일

10.2. FAT FSINFO

- FAT32 파일 시스템에 존재하는 구조체
 - 운영체제가 새로운 클러스터를 어디에 할당하는지 설명
 - 부트섹터에 위치
-

10.3. FAT 영역

- FAT 파일 시스템에서 중요한 역할
 - 두가지의 목적
 - 클러스터의 할당 상태 판단
 - 파일이나 디렉토리 다음에 할당할 클러스터를 찾는데 사용
 - 대체로 한 FAT 파일시스템에는 두개의 FAT 이 존재하고 정확한 번호는 부트섹터에서 준다
 - 첫 FAT은 부트섹터에서 주어진 예약된 영역 섹터 이후에 시작
 - 각 FAT의 크기도 주어지기 때문에 첫번째 FAT 이후 바로 두번째 FAT 시작
 - 두번째 FAT 테이블은 같은 크기의 엔트리들로 구성되나 헤더나 아래값이 없음
 - 클러스터가 할당되지 않으면 그 클러스터는 0
 - 마지막에 있는 클러스터면 엔트리는 마지막표시를 함
 - FAT 12 → 0xff8
 - FAT 16 → 0xffff8
 - FAT 32 → 0xffffffff8 보다 큰 어떤 값이든
 - 파일시스템에서 시작 클러스터는 2번부터
-

10.4. 디렉토리 엔트리

- FAT 디렉토리 엔트리에는 파일, 디렉토리 이름, 메타데이터가 저장되어있음
 - 이러한 엔트리 중 하나는 모든 파일과 디렉토리에 할당
 - 그 파일의 부모 디렉토리에 할당된 클러스터에 위치
 - 그 데이터 구조체에서 이름은 8글자 / 파일은 3글자만 사용
 - 그 이상은 LFN 엔트리 필요!
-

10.5. 긴 파일명 디렉토리 엔트리

- 표준 디렉토리 엔트리는 8자 이름 / 3글자 확장자 이름만 사용
- 한 파일에는 LFN 엔트리 외 일반 엔트리(SFN) 가 존재하며 LFN이 SFN보다 우선시된다