

101 - In Vehicle Infotainment

Team Information

Team Name : HSPACE

Team Member : Jinung Lee, Beomjun Park, DoHyeon Kim, Soyoung Cho

Email Address : hspacedigitalforensicslab@gmail.com

Teams must:

- Provide a detailed, step-by-step description of their problem-solving approach to ensure reproducibility by another examiner.
- List all tools used to arrive at their conclusions.

Tools used:

Name:	HashTab	Publisher:	Implbits Software
Version:	v6.0.0.34		
URL:	https://hashtab.softonic.kr/		

Name:	DB Browser for SQLite	Publisher:	
Version:	3.13.1		
URL:	https://sqlitebrowser.org/		

Step-by-step methodology:

문제 풀이에 앞서, dfchallenge.org에 공지된 문제 해시와 다운로드 받은 문제 해시를 비교함으로써 분석 대상이 동일한 파일임을 증명한다.

Hash Value (MD5)
• IVI.zip : 8c4616f8b6e39dc6f2f66d2746ba956d

Figure 1. dfchallenge.org에 공지된 문제 해시(MD5) 값

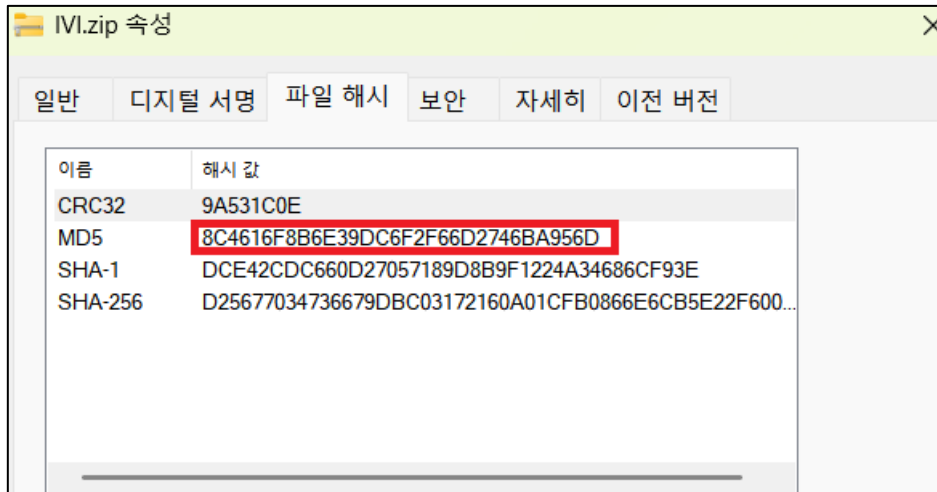


Figure 2. HashTab을 통해 확인한 문제 해시(MD5) 값

압축을 해제하면 두 개의 파일을 확인할 수 있으며, 파일 정보는 [Table1], [Table2]와 같다.

File Name	Kakaoi_DB.db
Size	53,248 bytes
MD5	9DF47E0D0F680675CAF882DF26E61A1E

Table 1. Kakaoi_DB.db 파일 정보

File Name	radio.db
Size	229,376 bytes
MD5	9750BCA6E4FB5855C15B17651E170569

Table 2. radio.db 파일 정보

Kakaoi_DB.db는 총 9개의 Table을 갖고 있으며, 사용자별로 차량에서 재생한 음악의 히스토리와 즐겨찾기 정보를 저장하고 있다.

데이터베이스 (9)	
> android_metadata	CREATE TABLE android_metadata (locale TEXT)
> soundhound_favorites	CREATE TABLE soundhound_favorites (id INTEGER PRIMARY KEY AUTOINCREMENT, data_url TEXT, image_url TEXT, data TEXT, trpe INTEGER, trpe INTEGER, trpe INTEGER)
> soundhound_history	CREATE TABLE soundhound_history (id INTEGER PRIMARY KEY AUTOINCREMENT, data_url TEXT, image_url TEXT, data TEXT, trpe INTEGER, trpe INTEGER, trpe INTEGER)
> soundhound_history_guest	CREATE TABLE soundhound_history_guest (id INTEGER PRIMARY KEY AUTOINCREMENT, data_url TEXT, image_url TEXT, data TEXT, trpe INTEGER, trpe INTEGER, trpe INTEGER)
> soundhound_history_user1	CREATE TABLE soundhound_history_user1 (id INTEGER PRIMARY KEY AUTOINCREMENT, data_url TEXT, image_url TEXT, data TEXT, trpe INTEGER, trpe INTEGER, trpe INTEGER)
> soundhound_history_user2	CREATE TABLE soundhound_history_user2 (id INTEGER PRIMARY KEY AUTOINCREMENT, data_url TEXT, image_url TEXT, data TEXT, trpe INTEGER, trpe INTEGER, trpe INTEGER)
> soundhound_update_history	CREATE TABLE soundhound_update_history (id INTEGER PRIMARY KEY AUTOINCREMENT, is_update INTEGER)
> soundhound_xml_data	CREATE TABLE soundhound_xml_data (id INTEGER PRIMARY KEY AUTOINCREMENT, data_url TEXT, data TEXT)
> sqlite_sequence	CREATE TABLE sqlite_sequence(name,seq)
인덱스 (0)	
뷰 (0)	
트리거 (12)	
trigger_delete_guest_history	CREATE TRIGGER trigger_delete_guest_history AFTER DELETE ON[soundhound_history_guest] FOR EACH ROW BEGIN REPLACE INTO soundhound_update_history (id, is_update) VALUES (1, 1);
trigger_delete_history	CREATE TRIGGER trigger_delete_history AFTER DELETE ON[soundhound_history] FOR EACH ROW BEGIN REPLACE INTO soundhound_update_history (id, is_update) VALUES (1, 1);
trigger_delete_user1_history	CREATE TRIGGER trigger_delete_user1_history AFTER DELETE ON[soundhound_history_user1] FOR EACH ROW BEGIN REPLACE INTO soundhound_update_history (id, is_update) VALUES (1, 1);
trigger_delete_user2_history	CREATE TRIGGER trigger_delete_user2_history AFTER DELETE ON[soundhound_history_user2] FOR EACH ROW BEGIN REPLACE INTO soundhound_update_history (id, is_update) VALUES (1, 1);
trigger_insert_guest_history	CREATE TRIGGER trigger_insert_guest_history AFTER INSERT ON[soundhound_history_guest] FOR EACH ROW BEGIN REPLACE INTO soundhound_update_history (id, is_update) VALUES (1, 1);
trigger_insert_history	CREATE TRIGGER trigger_insert_history AFTER INSERT ON[soundhound_history] FOR EACH ROW BEGIN REPLACE INTO soundhound_update_history (id, is_update) VALUES (1, 1);
trigger_insert_user1_history	CREATE TRIGGER trigger_insert_user1_history AFTER INSERT ON[soundhound_history_user1] FOR EACH ROW BEGIN REPLACE INTO soundhound_update_history (id, is_update) VALUES (1, 1);
trigger_insert_user2_history	CREATE TRIGGER trigger_insert_user2_history AFTER INSERT ON[soundhound_history_user2] FOR EACH ROW BEGIN REPLACE INTO soundhound_update_history (id, is_update) VALUES (1, 1);
trigger_update_guest_history	CREATE TRIGGER trigger_update_guest_history AFTER UPDATE ON[soundhound_history_guest] FOR EACH ROW BEGIN REPLACE INTO soundhound_update_history (id, is_update) VALUES (1, 1);
trigger_update_history	CREATE TRIGGER trigger_update_history AFTER UPDATE ON[soundhound_history] FOR EACH ROW BEGIN REPLACE INTO soundhound_update_history (id, is_update) VALUES (1, 1);
trigger_update_user1_history	CREATE TRIGGER trigger_update_user1_history AFTER UPDATE ON[soundhound_history_user1] FOR EACH ROW BEGIN REPLACE INTO soundhound_update_history (id, is_update) VALUES (1, 1);
trigger_update_user2_history	CREATE TRIGGER trigger_update_user2_history AFTER UPDATE ON[soundhound_history_user2] FOR EACH ROW BEGIN REPLACE INTO soundhound_update_history (id, is_update) VALUES (1, 1);

Figure 3. Kakaoi_DB.db 테이블 정보

radio.db 에는 12 개의 Table 이 존재하고 있으며, FM/AM 라디오 주파수, 방송국 정보, 지역별 채널 설정과 사용자 프리셋을 저장하고 있다.

이름	타입	스키마
테이블 (12)		
am_frequency	CREATE TABLE	am_frequency (frequency text, region1_cd text, region2_cd text, station_cd text, station_loc_cd text)
am_station	CREATE TABLE	am_station (station_cd, station_name text, station_name_e text)
android_metadata	CREATE TABLE	android_metadata (locale TEXT)
check_db	CREATE TABLE	check_db (check_value integer not null)
current_location	CREATE TABLE	current_location (seq_no integer not null, navi_cd text)
fm_frequency	CREATE TABLE	fm_frequency (frequency text, region1_cd text, region2_cd text, station_cd text, station_loc_cd text)
fm_station	CREATE TABLE	fm_station (station_cd text, station_name text, station_name_e text)
presets	CREATE TABLE	presets (band integer not null, preset_no integer not null, frequency text, hd_onoff integer not null, hd_number integer not null, rbds_ps_name text, service_id integer not null)
radio_freq_region	CREATE TABLE	radio_freq_region (freq_region integer not null)
radio_states	CREATE TABLE	radio_states (lastmode integer not null)
region_info	CREATE TABLE	region_info (navi_cd text, region1_cd text, region2_cd text, primary key(navi_cd))
station_location	CREATE TABLE	station_location (station_loc_cd text not null, location text, location_e text, primary key(station_loc_cd))

Figure 4. radio.db 테이블 정보

1. Kakaoi 어플을 이용하여 이 차에서 iKON의 "Love Scenario"를 들었던 시간은 언제 인가?

Kakaoi 어플은 음성 명령을 통해 카카오톡, 음악, 뉴스, 차량 제어 등 다양한 서비스를 제공하는 차량용 인포테인먼트 애플리케이션이다. Kakaoi_DB.db의 soundhound_history_user1 Table 을 확인해보면 두 개의 곡이 재생되었던 기록을 확인할 수 있다.

id	data_url	image_url	data	trpe	trpe_id	title	artist	album
1	http://api.midomi.com:443/v2/?...	http://goscrap.kakao.com/image/v2?...		2		사랑을 했다 (LOVE SCENARIO)	iKON	Return
2	http://api.midomi.com:443/v2/?...	http://goscrap.kakao.com/image/v2?...		2		High High	김태우	신사의 품격 OST Part

Figure 5. soundhound_history_user1 재생기록

iKON의 'Love Scenario' 재생 시간은 '1653103163303'으로 기록되어 있으며, 이는 밀리초 단위의 Unix 타임스탬프이다. 이를 문제에서 요구한 GMT 기준으로 변환하면 '2022-05-21 03:19:23 GMT'이다.

image_url	data	trpe	trpe_id	title	artist	album	time
http://goscrap.kakao.com/image/v2?...		2		사랑을 했다 (LOVE SCENARIO)	iKON	Return	1653103163303
http://goscrap.kakao.com/image/v2?...		2		High High	김태우	신사의 품격 OST Part 1	1653103179374

Figure 6. Love Scenario 재생시간

정답 : 2022-05-21 03:19:23 GMT

2. 이 IVI에 현재 설정된 라디오 주파수, 방송사, 지역은 어디인가?

현재 IVI에 설정된 위치를 확인하기 위해 radio.db의 current_location 테이블을 분석한 결과, navi_cd 값이 "26320"으로 설정되어 있다. 이는 행정안전부 행정구역코드 체계에 따르면 부산광역시 북구에 해당한다.

테이블(T): current_location					
	seq_no		navi_cd		
	필터	필터			
1	1	26320			

Figure 7. current_location 테이블 내용

해당 지역코드와 매핑되는 지역의 정보는 region_info 테이블에서 확인할 수 있었다. region_info 테이블에서 해당 지역코드와 매핑되는 region1_cd는 "0x0B", region2_cd는 "0x07"로 확인된다.

테이블(T): region_info					
	navi_cd	region1_cd	region2_cd		
	26320	필터	필터		
1	26320	0x0B	0x07		

Figure 8. region_info 지역코드

각 지역별로 수신 가능한 AM 라디오 주파수와 FM 라디오 주파수 정보가 별도의 테이블에 저장되어 있다.

fm_frequency 테이블 분석 결과, 해당 지역코드(region1_cd: 0x0B, region2_cd: 0x07)에서 수신 가능한 두 개의 FM 주파수를 확인할 수 있었다. 각 레코드에는 방송국 식별을 위한 station_cd와 지역 식별을 위한 station_loc_cd 정보가 포함되어 있다.

테이블(T): fm_frequency					
	frequency	region1_cd	egion2_cd	station_cd	station_loc_cd
	필터	0x0b	0x07	필터	필터
1	88.5	0x0B	0x07	0x30	0x1C
2	89.3	0x0B	0x07	0x04	0x1E

Figure 9. fm_frequency 테이블 내 주파수 정보

fm_frequency 테이블을 기준으로 fm_station 테이블과 station_location 테이블을 LEFT JOIN하여 특정 지역코드(0x0B, 0x07)에 해당하는 FM 주파수의 상세 정보를 추출하였다. 아래 쿼리를 통해 현재 지역에서 이용 가능한 FM 주파수별 채널명(한국어/영어), 지역 정보(한국어/영어), 그리고 각 종 식별코드를 모두 확인할 수 있다.

```
SELECT
    'FM' as band,
```

```

f.frequency,
f.region1_cd,
f.region2_cd,
f.station_cd,
f.station_loc_cd,
s.station_name,
s.station_name as station_name_kr2,
s.station_name_e,
s.station_name_e as station_name_en2,
l.location as location_kr,
l.location_e as location_en
FROM fm_frequency f
LEFT JOIN fm_station s ON f.station_cd = s.station_cd
LEFT JOIN station_location l ON f.station_loc_cd = l.station_loc_cd
WHERE f.region1_cd = '0x0B' AND f.region2_cd = '0x07'
ORDER BY f.frequency;

```

Table 3. 현재 설정된 라디오 주파수 및 방송국 정보 조회 쿼리

따라서 현재 설정된 FM 라디오 주파수 및 방송 채널 정보는 다음과 같다.

band	freq	region1_cd	region2_cd	station_cd	station_loc_cd	station_name	location_kr
FM	88.5	0x0B	0x07	0x30	0x1C	KNN 러브FM	부산
FM	89.3	0x0B	0x07	0x04	0x1E	KBS 1FM (Classic FM)	진주

Table 4. 현재 설정된 FM 라디오 주파수 및 방송 채널

반면 am_frequency 테이블에서는 동일한 지역 정보와 일치하는 AM 주파수 대역이 존재하지 않았다. 따라서 현재 설정된 지역에서는 FM 라디오만 이용 가능한 상태로 확인된다.

테이블(T): am_frequency

frequency	region1_cd	region2_cd	station_cd	station_loc_cd
필터	0x0b	0x07	필터	필터

Figure 10. am_freauency 테이블 내 주파수 정보

정답 :

- FM 98.5MHz, KNN 러브FM, 부산

- FM 89.3MHz, KBS 1FM (Classic FM), 진주