SPaRe: 데이터베이스 스키마 패턴 분석을 이용한 SQLite 복원기법

이 성 일[1] 이 준 락[2] 염 흥 렬[3*]

◈ 목 차 ◈

1. 서 론

2014년 가트너 선정 10대 전략기술에 만물 인터넷(IoE: Internet of Everything)이 포함 되었다. 2012년 사물인터넷(IoT: Internet of Things)이 포함된 이래 ICT(Information Communication Technology) 분야에서 폭발적인 관심을 받고 있다. 최근에는 IoE기술범주에 사람(People), 사물(Things), 장소(Places), 정보(Information) 등이 포함되어 PC와 모바일 기기를 넘어 다양한 기기가 인터넷을 통해 연결되는 초연결사회로 발돋움하고 있다 [1].

IoT환경에서는 IoT를 구성하는 스마트폰, 타블렛, 웨어러블(Wearable)기기 등 다양한 디바이스들이 사용자의 인지 또는 간섭 없이 서로 간의 연결 및 통신을 통해서 사용자에게 서비스를 제공한다. 협소한 저장공간을 가지는 IoT기기의 특성상, 디지털 형태로 변환된 데이터의 효율적인 저장 및 관리가 중요하다. SQLite는 경량화 DBMS(Database Management System)로 스마트폰 등 IoT기기에서의 사용에 최적화되어 있어 IoT 응용에서 널리 사용되고 있다 [2].

디지털 포렌식(Digital Forensic)은 PC나 노트북, 휴대폰 등 각종 저장매체 또는 인터넷 상에 남아 있는 각종 디지털 정보를 분석해 범죄 단서를 찾는 수사기법이다. 최근 IoT기기 사용자들은 통화기록, 이메일 접속기록 등 디지털정보를 IoT기기를 통하여 접근한다. 특히 저장공간이 협소한 IoT기기의 특성상 데이터의 삭제가 빈번하게 이루어진다. 더욱이 포렌식 수사에서의 증거은닉을 위해 용의자가 인위적으로 데이터를 삭제하는 경우도 있어 삭제된 데이터를 데이터베이스에서 복원하는 것은 포렌식 기술에서 중요한 연구분야라 할 수 있다 [3].

최근 포렌식 연구분야에서 SQLite의 데이터 베이스 영역에서 삭제데이터의 복원기법에 관련된 연구가 진행되어 왔다 [4-8]. SQLite 데이터베이스 복원기법 연구는 Pereira에 의해 근원을 찾을 수 있는데, 저널파일을 이용하여 파이어폭스3의 삭제된 레코드를 복원하는 방법이 제안되었다 [4]. [5]에서는 SQLite의 삭제된 오버 플로우 데이터를 복구하는 기법을 제안하였다. [6]에서는 SQLite의 비 할당영역[1]에 잔존하는 삭제 레코드를 복원하는 기법을 제안하였다.

기존에는 SQLite헤더를 분석하여 비 할당영역을 탐색하고 해당 영역에 잔존하는 삭제 데이터를 복원하였으나, 이 복원기법은 레코드가 개별 삭제된 경우에는 복원이 불가능하다는 한계점이 있다.

본고에서는 기존연구의 한계를 극복하기 위한 방법으로 SPaRe (Schema Patten Recovery)를 제안한다. SPaRe는 SQLite 데이터베이스의 스키마 패턴을 분석하여, 할당 및 비 할당 영역에 잔존하는 개별 삭제데이터를 복원할 수 있는 기법이다. SPaRe는 기존과 같

1 ETRI부설연구소 선임연구원
2 강원대학교 교수
3 순천향대학교 교수

1) 비 할당영역은 비 할당공간(Unallocated Area)와 비 할당 블록(Free Block)의 2가지 종류가 있다.

이 그룹 단위로 삭제된 비 할당영역에 잔존하는 데이터를 복원할 수 있으며, 추가적으로 할당영역에 잔존하는 개별 삭제데이터를 복원할 수 있다.

본고의 구성은 다음과 같다. 2장에서 관련 연구를 살펴보고, 3장에서 SQLite의 데이터베이스 저장형식을 간략하게 분석 기술한다. 4장에서 제안하는 스키마 패턴을 이용한 삭제된 레코드복원 기법에 대해 기술한다. 5장에서 실험결과를 기술하고, 6장에서 본고를 맺는다.

2. 관련 연구

2.1 Journal파일을 이용한 복원

SQLite는 데이터베이스 쿼리를 처리하는 과정에서 journal파일을 생성하여 중간단계 입출력 과정을 저장한다. M. T. Pereira는 journal파일을 분석하여 파이어폭스3의 삭제된 레코드를 복구하는 기법을 제안하였다 [4]. M. T. Pereira의 연구결과는 SQLite 데이터베이스 삭제데이터 복원기법 연구의 시발점이 되었으며, 이후 많은 journal파일을 이용한 트랜잭션 기반 SQLite 데이터베이스 복원연구가 진행되었다.

2.2 SQLite 비 할당 영역의 삭제 데이터 복원

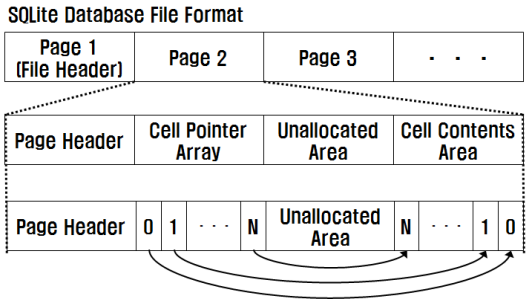
S. Jeon의 연구 [6]는 SQLite 데이터베이스 데이터 저장방식을 분석하여 목록 삭제가 수행된 데이터가 SQLite 비 할당영역에 잔존하는 점에 착안하여 레코드 복원기법을 제안하였다. [5]에서는 유사한 방식으로 오버플로우 데이터를 복원할 수 있는 기법을 제안하였다. [5][6] 연구는 SQLite의 비 할당영역을 탐색함으로서 빠르게 목록 삭제된 레코드를 복원할 수 있는 장점이 있으나 개별 삭제를 통해서 할당영역에 잔존하고 있는 데이터는 복원할 수 없다. 우리가 제안하는 SPaRe의 경우 비 할당영역의 목록 삭제된 데이터의 복원이 가능하며 할당영역에 잔존하는 개별 삭제 데이터를 SQLite 데이터베이스 스키마 패턴분석을 통해 파싱하여 복원할 수 있다.

2.3 SQLite 로그를 이용한 복원

SQLite는 트랜잭션의 원자성과 지속성을 보존하기 위해 물리적 로깅 기반의 복원을 제공한다. SQLite 로깅은 한 페이지 내의 일부 데이터가 수정되더라도 전체 페이지를 저장하기 때문에 로그의 큰 단점이 있다. [8]에서는 논리적 로깅 기반으로 작업의 종류 및 입력값만을 저장하여 생성되는 로그의 크기를 줄일 수 있는 방법인 Delta-WAL을 제안하였다. Delta-WAL은 SQLite의 기본제공 복원 효율을 제고한 방법으로 본 연구와는 독립이다.

3. SQLite 데이터베이스 저장형식 분석

3.1 SQLite 데이터베이스 파일



(그림 1) SQLite 데이터베이스 파일 구조

SQLite 데이터베이스 파일의 구조를 (그림 1)에 도식하였다. SQLite 데이터베이스 파일은 페이지(Page)단위로 구성된다. 첫 번째 페이지는 헤더(Header) 페이지로 데이터베이스 파일 헤더 정보와 테이블 정보 등 18종류의 메타데이터 정보가 기록된다 [2].

페이지의 종류는 Leaf페이지, Internal페이지, Overflow페이지, Freelist페이지 등이 있으며, Leaf페이지에 실제 레코드가 저장된다.

페이지는 페이지 헤더와 셀 포인터 배열, 비 할당영역, 셀 콘텐츠 영역으로 구분된다. 셀(Cell)이란 페이지 내부에서 관리하는 하나의 정보 단위이며 하나의 셀이 곧 하나의 레코드 관련 정보가 된다.

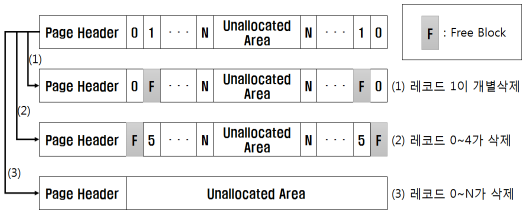
3.2 SQLite 레코드 삭제

SQLite 데이터베이스에서의 레코드의 삭제는 아래와 같은 3종류의 유형으로 구현된다. 응용 프로그램별로 채택하고 있는 레코드의 삭제방법은 다양하며, 대표적인 삭제방법 구현 예를 (표 1)에 소개하였다.

(표 1) 응용 프로그램별 레코드 삭제 구현방법

구현방법	대표 응용프로그램
0으로 덮어씀	파이어폭스3, 사파리, 크롬
삭제 영역 제거	아이폰 OS (문자 개별삭제)
비 할당영역에 포함	스카이프, 아이폰 OS (문자 그룹삭제, 통화기록)

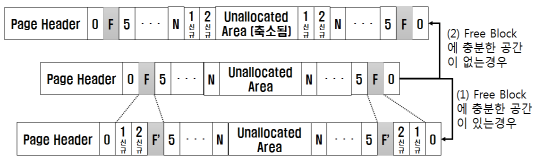
삭제된 데이터를 다른 데이터로 덮어쓰는 경우에는 어떠한 경우에도 복원이 불가능하며, (표 1)의 첫 번째와 두 번째 유형에 해당한다. (표 1)의 세 번째 유형은 데이터 영역을 비 할당 영역에 포함시키고 실제 데이터는 데이터베이스 파일에 잔존하여 복원이 가능하다.



(그림 2) SQLite 레코드삭제 구현방법 (유형 3)

(그림 2)는 세 번째의 경우를 좀 더 상세하게 구분한 것이다. 1개 이상의 레코드가 삭제되면 해당 데이터가 저장되어 있던 공간을 Free Block으로 구분하고 실제데이터는 남아있게 된다 (그림 2-1, 2). 그러나 레코드의 전수 삭제(통화목록 등)가 일어나게 되면 해당 영역을 비 할당공간(Unallocated Area)으로 포함시킨다.

3.3 SQLite 신규 레코드 삽입



(그림 3) SQLite 신규레코드의 기록

새로운 레코드가 기록되어야 할 경우, SQLite는 우선적으로 Free Block에 데이터의 저장 공간이 충분한지를 검사한다. 만약 공간이 충분하다면 Free Block에 새로운 레코드가 기록이 되며(그림 3-1), 만약 공간이 충분하지 않으면 비 할당공간에서 신규 레코드를 위한 영역을 생성한다 (그림 3-2)에 신규 레코드가 기록되는 과정을 도식하였다. 복원이 어려운 경우는 신규 레코드의 삽입으로 Free Block의 데이터가 덮어씌워진 경우이다. (그림 3-1)에서는 Free Block공간 중 일부가 신규 레코드로 덮어씌워졌으며 (남겨진 부분은 F'로 표기) 레코드의 길이가 가변인 점을 고려하면(3.4에 기술) 실제적으로 몇 개의 레코드를 복원할 수 있는지 알 수 없다.

3.4 SQLite 레코드의 자료형

SQLite 데이터베이스에서 스키마 자료 형에 대한 종류는 Signed Integer(정수형), IEEE Float(실수형), Integer 1(상수 1), Integer 0(상수 0), BLOB(바이너리형), TEXT(문자형)가 있다. SQLite는 데이터베이스 파일의 공간효율성을 위해 가변길이 타입을 정의하여 사용한다. 각 스키마 자료 형에 대한 페이로드 타입과 데이터의 크기는 (표 2)와 같다.

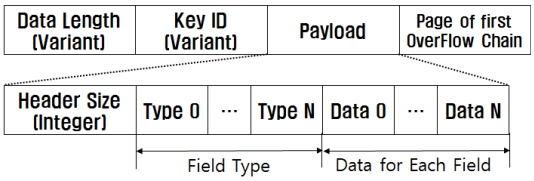
(표 2) SQLite 레코드 자료형

자료형	타입	크기(B)
Signed Integer	0~6	0~8
IEEE Float	7	8
Integer 1	8	0
Integet 0	9	0
BLOB	12이상의 짝수(가변)	(N-12)/2
TEXT	13이상의 홀수(가변)	(N-13)/2

4. SPaRe: 스키마 패턴을 이용한 복원

4장에서는 SPaRe(Scheme Pattern Recovery)에 대해 기술한다. SQLite는 데이터베이스 파일에 트리형의 자료구조를 이용하여 레코드를 기록한다. 3장에서 살펴본 바와 같이 셀 포인터가 실제 레코드를 담고 있는 셀을 가리키고 있다. SQLite셀은 헤더와 페이로드로 구성되어 있으며 헤더를 분석하여 페이로드에 저장된 레코드의 필드의 개수와 자료형(필드타입)을 알 수 있다.

4.1 삭제 레코드 헤더 파싱



(그림 4) SQLite 신규레코드의 기록

(그림 4)는 SQLite에서 실제 레코드가 저장되는 Leaf셀을 도식한 것이다. 셀의 페이로드 영역에 레코드가 기록되며 페이로드는 헤더, 필드별 자료형, 그리고 데이터로 구성된다.

SQLite 데이터베이스에서 삭제 레코드의 필드별 자료형의 표현형은 (표 3)과 같다.

(표 3) SQLite 레코드 자료형

스키마 자료형	Type 필드 표현형태		자료형 길이(B)
	16진수	10진수	
Signed Integer	0x00 ~ 0x06	0 ~ 6	1
IEEE Float	0x08	8	1
Integer 1	0x09	9	1
Integet 0	0x07	7	1
BLOB	0x0C 이상 짝수	12 이상 짝수	1~9
TEXT	0x0D 이상 홀수	13 이상 홀수	1~9

(표 3)은 SQLite에 바이트단위로 기록된 스키마 패턴이(헥사값) 레코드데이터의 자료형 정보를 의미할 수 있음을 의미한다. 예를 들어, SQLite 데이터베이스 파일의 페이지에 기록된 0x00은 Signed Integer자료형의 타입 값일 수도 있다. 그러나 0x00은 정수 값 0의 1바이트 표현일 수도 있다. 제안하는 SPaRe는 SQLite 페이지를 전수 조사하여, 페이지의 모든 위치에 저장된(혹은 부분적으로 잔존하는) 헤더를 검색하여 레코드를 파싱한다. 우리는 이를 '삭제 레코드 헤더 파싱'이라 부른다.

4.2 파싱 알고리즘

(표 4) Message 테이블의 스키마 구조

```
CREATE TABLE Message; {
    RowID,      INTEGER     PRIMARY KEY,
    Date,       INTEGER,
    Address,    TEXT,
    Text,       TEXT,
    Data,       BLOB,
}
```

SPaRe는 3.1에서 기술한바와 같이 헤더페이지를 통해 페이지에 저장된 레코드의 스키마 구조를 알 수 있다. 예를 들어, (표 4)의 CREATE쿼리를 통해 생성된 Message 테이블의 스키마는, Integer, Integer, Text, Text, Text가 된다. 이를 표현하는 SQLite 데이터베이스 필드 표현형 중 하나는 "0x08 08 0D 0D 0C"이며, 또 다른 필드 표현형은 "0x09 08 0D 0D 0C"이다. 우리는 이와 같은 필드 표현형을 '스키마 패턴'이라 부른다. SQLite는 가변길이 자료형을 지원하므로 Message 테이블의 스키마 패턴은 이밖에도 여러 가지가 있을 수 있다. SPaRe는 페이지를 전수 조사하여 레코드의 스키마 패턴[2]과 부합하면 이를 레코드로 간주한다.

(표 5)는 SPaRe의 스키마 패턴 파싱 알고리즘을 도식한 것이다. 알고리즘은 페이지에서 스키마패턴의 길이와 같은 모든 패턴에 대하여 부합여부를 조사한다.

2) SQLite에서는 가변길이 자료형을 지원하므로 스키마패턴은 여러 종류일 수 있다.

SPaRe에서는 우선 FIFO큐 형태의 agenda에 조사 페이지에서 등장하는 모든 비트패턴 중 탐색하고자 하는 스키마패턴과 길이가 같은 모든 패턴을 삽입한다. agenda에서 원소를 추출하여 이를 스키마패턴과 비교하고, 부합할 경우 스키마패턴길이 만큼 agenda에서 원소를 추출하여 버림으로서 불필요한 비교를 피한다. 제안하는 알고리즘은 수행측면에서 최적화가 가능하나 계산집적도가 높지 않은 SQLite의 특성을 고려하여 본고에서는 이를 기술하지 않는다. 알고리즘을 통해 탐색한 헤더정보를 통해 페이로드 데이터영역을 읽어옴으로서 삭제된 레코드를 복원한다.

(표 5) 파싱 알고리즘

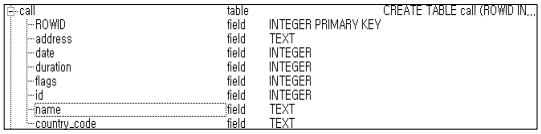
```
Parsing algorithm for SPaRe
input:
  page y of length l_y
  schema pattern x of length l_x
output:
  found schema pattern(s) in y
  location(s) of found schema pattern
begin
 /* constructs agenda (queue) */
 for i from 0 to l_y − 1 do
   add substring y_i…y_{i+l_x−1} to agenda
 end

 location ← 0
 while agenda is not empty do
   given a substring s ← pop(agenda) do
   location++;
     if s=x then
       while l_x − 1 > 0
         pop(agenda)
         location++;
       end
       found schema pattern += s
       location(s) of found scheme pattern += location
     end
   end
 end
end
```

5. 실험결과

이 장에서는 4장에서 기술한 SPaRe를 이용한 삭제 레코드 복원 기법을 구현하여 실험한 결과를 기술한다.

5.1 환경

우리는 iPhone 6 (iOS version 7.x, SQLite version 3.7.x)를 사용하여 실험하였다. 사용된 SQLite 데이터베이스 파일은 통화기록이다. 포렌식 조사에서 용의자 혹은 사용자가 증거은닉을 위해 인위적으로 통화기록을 '부분삭제'하였다고 가정한다. iOS 7.x의 통화기록 테이블의 스키마정보는 (그림 5)과 같다. 통계적으로 유의미한 실험결과를 얻기 위해 랜덤하게 통화기록정보를 삭제하여 실험을 반복하였다.



(그림 5) 통화기록 테이블의 스키마 정보

5.2 결과

	Status	Address	Date	Duration	Flags	ID	Contry
75	삭제	0109062	2011/08/01 22:35:00	0	Send(5)	85	450
76	삭제	032542	2011/08/16 15:01:14	9	Recevie(4)	-1	450
77	삭제	0107799	2011/08/16 13:35:55	0	Unknown(2...	5	45
78	삭제	0107799	2011/08/16 09:12:50	18	Recevie(4)	-1	450
79	삭제	01	2011/08/16 08:13:46	0	Unknown(1...	-1	450
80	삭제	+824258	2011/08/14 16:45:56	20	Send(5)	-1	450
81	삭제	+824258	2011/08/14 16:45:18	24	Send(5)	-1	450
82	삭제	0105434	2011/08/14 13:26:00	0	Send(5)	104	450
83	삭제	0102470	2011/08/11 21:38:35	72	Send(5)	7	450
84	삭제	0108972	2011/08/11 15:16:23	0	Send(5)	-1	450
85	삭제	0103023	2011/08/10 22:28:29	31	Send(5)	-1	450
86	삭제	0107799	2011/08/10 22:27:39	25	Recevie(4)	-1	450
87	삭제	0103023	2011/08/10 22:07:17	0	Send(5)	-1	450
88	삭제	0103023	2011/08/10 21:47:44	60	Send(5)	-1	450
89	삭제	0103537	2011/08/10 20:11:45	74	Send(5)	68	450
90	삭제	0109648	2011/08/10 18:42:01	35	Recevie(4)	-1	450
91	삭제	0107799	2011/08/10 16:41:55	44	Send(5)	86	000
92	삭제	0107799	2011/08/10 16:32:37	0	Recevie(4)	-1	450
93	삭제	028289	2011/08/10 14:07:42	42	Recevie(4)	-1	450
94	삭제	0109648	2011/08/10 14:01:57	66	Send(5)	19	450
95	삭제	0108807	2011/08/10 13:30:26	0	Recevie(4)	-1	450
96	삭제	0108807	2011/08/10 13:20:14	0	Recevie(4)	-1	450
97	삭제	0109256	2011/08/10 12:45:01	4	Send(5)	123	450
98	삭제	0109256	2011/08/10 12:42:37	0	Send(5)	123	450

(그림 6) SPaRe를 통한 통화기록 복원결과

(그림 6)은 SPaRe를 통하여 통화기록을 복원한 결과를 보여준다. SPaRe는 통계적으로 98%이상의 삭제 레코드 복원율을 보였다. 실험결과의 심층 분석결과, 복원이 되지 않은 2%는 '개별삭제'영역에 데이터가 덮어씌워짐으로서 복원이 원천적으로 불가능한 경우였다.

SPaRe의 파싱알고리즘은 복원 가능한 (덮어씌워지지 않은) 모든 잔존하는 데이터를 파싱하였다. (표 6)에 SPaRe와 기존 복원기법과 비교하였다.

(표 6) 기존 복원기법 [6]과 SPaRe의 비교

SQLite 영역		[6]	SPaRe
비 할당공간		복원 가능	복원 가능
Free Block	단일	일부 복원 가능	전체 복원 가능
	연속된	복원 불가	복원 가능
	덮어씀	복원 불가	복원 불가

6. 결 론

본고에서는 SQLite 데이터베이스 파일 구조에서 스키마의 자료 형 패턴을 이용하여 셀의 페이로드 타입을 탐지해 레코드를 복원하는 기법인 SPaRe를 제안하였다. iPhone 6를 이용한 실험에서 기존에 불가능했던 Free Block에서의 삭제데이터의 복원이 SPaRe를 통해 가능함을 보여주었다.

SQLite 데이터베이스의 파일 구조에서 삭제된 레코드 복원 기술은 앞으로도 많은 연구가 진행 될 것이다. 이 과정에서 페이지 내부에 존재하는 레코드를 탐색하기 위한 방법으로 제안한 기법이 활용 될 수 있을 것으로 기대한다. 또한, SQLite 데이터베이스 파일 포맷이 손상되어 정상적으로 인식 할 수 없는 경우에 제안한 기법을 활용하면 활성 레코드와 삭제 레코드 모두 복원 할 수 있어 디지털 포렌식 수사에 도움이 될 것으로 기대한다.

참 고 문 헌

[1] David W. Cearley, "The Top 10 Strategic Technology Trends for 2014" The Top 10 Strategic Technology Trends (2014).

[2] M. Owens and G. Allen, "SQLite", Apress LP, 2010.

[3] G. Palmer, "A Road Map for Digital Forensic Research", First Digital Forensic Workshop, Utica, New York, 2001.

[4] M. T. Pereira, "Forensic Analysis of the Firefox 3 Internet History and Recovery of Deleted SQLite Records", Digital Investigation, Vol. 5. No. 3, pp.93-103, 2009.

[5] K. Lee, S. Yang, W. Hwang, K. Kim, T, Jang, and G. Son, A Recovery Scheme for the Deleted Overflow Data in SQLite Database", 한국정보기술학회논문지, Vol. 10, No. 11, pp.143-153, 2011.

[6] S. Jeon, J. Bang, K. Byun, and S. Lee, "A Recovery Method of Deleted Record for SQLite Database", Personal and Ubiquitous Computing, Vol. 16, No. 6, pp. 707-715, 2012.

[7] S. Lee and H. Yum, "A Recovery Method of Deleted Record Using The Schema Pattern Analysis for SQLite Database ", The workshop for digital forensic technique, 2011. 8.

[8] J. Lee, M. Shin, Y. Jang, and S. Park. "A Novel Recovery Scheme for SQLite Based on Logical Logging", Journal of KIIT. Vol. 12, No. 11, pp. 181-192, Nov. 30, 2014.

◐ 저 자 소 개 ◑

이 성 일 (Lee, Sung Il)
1999년 한밭대학교 컴퓨터공학과 학사
2005년 순천향대학교 산업정보대학원 정보보호학과 석사
2008년 순천향대학교 일반대학원 정보보호학과 박사과정
2008년~ 현재 ETRI부설연구소 선임연구원
관심분야 :포렌식, 네트워크 보안, 자동차 보안
E-Mail : silee@ensec.re.kr

이 준 락 (Lee, Jun-Rak)
1984년 인하대학교 수학과 학사
1986년 인하대학교 수학과 석사
1991년 인하대학교 수학과 박사
1995년 ~ 현재 강원대학교 교수
관심분야 : 해석학, 무선 통신, 정보 보안
E-Mail : jrlee@kangwon.ac.kr

염 흥 렬 (Youm, Heung-Youl)
한양대학교 전자공학 학사
한양대학교 대학원 전자공학 석사
한양대학교 대학원 전자공학 박사
현 순천향대학교 교수
관심분야 : 인터넷 보안, USN보안, IPTV 보안, 홈네트워크 보안, 암호 프로토콜
E-Mail : hyyoum@sch.ac.kr