Final Year Project Report on

DB4n6 - SQLite File Carving, Standardization and Data Visualization

Submitted in the partial fulfillment of the degree

of

Bachelor of Technology

in

Computer Science and Engineering

By

SABBISETTI AMARNADH

B180041CS

Department of Computer Science and Engineering

National Institute of Technology Sikkim

Under the Supervision of

DR. ANAND KUMAR MISHRA

Assistant Professor

May 2022

CERTIFICATE BY THE SUPERVISOR

This is to certify that the project report entitled **“ DB4n6 - SQLite File Carving, Standardization and Data Visualization ”** is being submitted by **Sabbisetti Amarnadh (Roll No: B180041CS)**, student in the **Department of Computer Science and Engineering, National Institute of Technology Sikkim,** for the award of the degree of **Bachelor of Technology (B.Tech)**. It is carried out by him under my supervision and guidance. The results embodied in this report have not been submitted to any other University or Institute for the award of any degree or diploma.

Dr. Anand Kumar Mishra
Assistant Professor
Department of Computer Science and Engineering
National Institute of Technology Sikkim
Ravangla - 737139, Sikkim, India

CERTIFICATE BY THE STUDENT

I hereby declare that the work presented in the report entitled **“ DB4N6 ”** is a bonafide record of the work done by me under the supervision of **Dr. Anand Kumar Mishra**, Department of Computer Science and Engineering and that no part thereof has been presented for the award of any other degree.

- I have followed the guidelines provided by the institute in writing the report.
- The report does not contain any classified information.
- Whenever, I have used materials from other sources, We have given due credits to those by citing them in the text of the report and giving the details in the references.
- Whenever, I have quoted written materials from other sources, I have put them under quotation marks and given due credits to the sources by citing them in the text of the report and giving their details in the references.

Dated: 23-05-2022	**Sabbisetti Amarnadh**
Place: Ravangla	**Roll No: B180041CS**

ACKNOWLEDGEMENT

I am tremendously indebted to our supervisor **Dr. Anand Kumar Mishra**, Assistant Professor, Department of Computer Science and Engineering, National Institute of Technology Sikkim for her invariable guidance and assistance through the project. Her advice and suggestions have been prized in the development and progress of the content. Furthermore, the skills and knowledge which I have gained through this project I perceive that as very valuable and significant for my future.

I would also like to thank, **Dr. Pratyay Kuila**, the Head of the Department, Department of Computer Science and Engineering for giving this opportunity to do this project as my final year project for our Bachelor's degree.

Sabbisetti Amarnadh
Roll No: B180041CS
Computer Science and Engineering
National Institute of Technology Sikkim
Ravangla - 737139, Sikkim, India

TABLE OF CONTENTS

Title **Page No.**

CERTIFICATE BY SUPERVISOR . . . . **i**

CERTIFICATE BY THE STUDENT . . . . **ii**

ACKNOWLEDGEMENT . . . . **iii**

TABLE OF CONTENTS . . . . **iv**

ABSTRACT . . . . **vi**

1 Introduction **1**

1.1 **Background** 3

1.1.1 **Computer Forensics** 3

1.1.2 **Memory Acquisition Techniques** 6

1.1.3 **File Carving** 7

1.1.4 **SQLite Database Carving** 9

1.2 **Motivation** 15

1.3 **Problem Statement** 15

1.4 **Problem Objective** 15

2 Literature Survey **16**

3 Proposed Approach **25**

3.1 **Our Contribution** 25

3.2 **DB4n6 Model** 26
3.3 **DB4n6 Working** 26
3.3.1 **Database Carving** 27
3.3.2 **View and Search** 29
3.3.3 **Visualisation data in DB4n6** 30
3.3.4 **Filtering in DB4n6** 30
3.4 **Implementation** 31
3.5 **Case Study** 33
3.5.1 **Case 1** 33
3.5.2 **Case 2** 40
3.6 **Comparison Table** 41

4 Conclusion and Future Work **42**
4.1 **Conclusion** 42
4.2 **Future Work** 43

References **44**

ABSTRACT

Recently, Everyone is using SQLite very widely in mobiles, computers.etc.,. Whatsapp, Skype, Chrome, etc these are mostly using applications.Now, Forensic investigators face issues like a lot of data to analyze, less available tools, etc. In the market, there are a very small number of tools used to analyze these kinds of data but they also don't give the complete report or expected result. Sometimes, Forensic investigator's don't even consider this data as evidence when compared to the large amount they are analyzing. But they don't waste time analyzing this type of data by using multiple tools.There are some tools available in the market, can only do one task file (or) database carving but they don't provide analytic tools to analyze the data which is extracted by using current available tools. We have to use different tools to analyze and visualize data.

In this report, we are given a solution to save time and without using multiple tools by using our new tool named **"DB4N6 Tool"** which is used to carve, visualize and filter the data. By applying carving technique, we can restore the deleted data, missing data, etc.,. Our tool gives a more accurate percentage than other tools. And we visualize result data and filter it very easily.

In this report, we give a complete procedure to do database forensics from scratch (memory acquisition) to data visualization.

Keywords: Memory Forensics, Mobile Forensics, Database Forensics, Data Discovery, SQLite, Carving, Tools and Techniques.

Introduction

Nowadays, everyone spends most of time using mobiles, Computers, etc. Most of the data is stored or recorded in databases, particularly in relational databases. Because relational databases are easy to handle when compared to NoSQl databases. This the reason why almost all types of applications like mobile apps, chrome browser, enterprise-management systems, etc. use relational databases. All over the relational database management system, SQLite the light weighted management system are very easy to install and to use. That is one of the reason most recently used database in the mobile phones are relational.

- SQLite based mobile applications are Whatsapp, Chrome browser, etc.
- SQLite databases are used by many softwares and browsers like chrome in computer.

Nowadays attacks like malware injections and other types of cyber-crimes are resulting in the loss, corruption or modification of the data. And also there are high chances of having our own data being used against us under mischevious circumstances by the attacker, so on and so forth. Sometimes it is pretty challenging to find out what exactly was done to our system.

In this report, a detailed explanation is given on how to find and restore the deleted/ corrupted/ modified data by the attacker by using our new proposed tool and some additional techniques. We are mainly focusing on the SQLite Databases and hence, this entire process is called SQLite Database Forensics.

In this report, we are going to use Memory forensics, Mobile forensics and Database forensics techniques to extract, collect and analyze the data we aimed to study, and explanation is going to be step to step approach of database forensics from scratch (in other words, analyzing the data from a system which is attacked by an attacker). So, Let's start with gathering the data from the victim computer due to an attack by a hacker or malware attack. This process of gathering data is also known as Memory Acquisition. There are many acquisition techniques available, but we are mentioning some techniques to acquire data for analysis.

Firstly, we have to gather data from the victim computer or mobile. For acquiring data form certain devices, we can use different types of techniques classified under Hardware based techniques and Software based techniques. The process of acquiring or gathering data is also known as Memory Acquisition. We implemented some techniques to get data from victim computer or mobile. We are about to discuss those things in below sections.

By performing a few memory acquisition techniques we get data in the formats .RAW, .MEM, .DMP, .dd, etc., which are the formats of byte files. These type of files are accessed by using some third-party tools like HexViewer, etc. By using these kind of tools we can extract hexadecimal codes of images with formats .png, .jpg, .jpeg, etc., if we know the HEX Header and Footer of file type. This is also known as Image file Carving. A brief explanation of carving can be seen in below sections.

For our research work, we have to extract .sqlite/ .db/ .sqlite-wal/ .sqlite-journal files. These are the files we have to extract and analyze. By doing so we can recover the database and then we can do some forensics on it.

We use BelkasoftX Forensic Tool to extract the database files. Belkasoft is a well known forensic tool which is main used only for file carving. It has ability to carve all type of files which are in dump file.

For our research work, we have acqurie or gather or collect all kinds of SQLite database related files which file types .db, .sqlite, .sqlite-wal, .sqlite-journal. By using SQLite Viewer feature in BelkasoftX tool we can get this things manually.
Therefore, we created our own datasets by doing all the above steps.

Now we can use our tool ”DBForensic” to carve the data from multiple databases and reconstruct those databases.We can also view those databases database level, table level, page level, row level data. After that we can also use filter option between the databases or tables.

We can do all above things by using multiple tools. For database carving, we have to use tool like DBCarver, Drinkwater, etc.,. then it will give .csv or .xml formatted files as output then we have to use some other tools for visualizing and filtering.

Without using much time and space, By using our tool we will get result directly with out using any extra space and less time complexity and giving more accuracy in recovery of missing, deleted and unallocated data (by doing carving we get more accurate data). We given detailed explanation in further sections.

1.1 Background

1.1.1 Computer Forensics

Computer Forensics is the process of using computer science knowledge for collecting and analyzing data from computers (or) networks (or) storage devices, presenting it as evidence which is admissible in a court. Computer forensic technique used to determine and reveal computer or network related technical criminal evidence.

Criminal Activities:-

- Committed against- individuals, companies, institutions
- Involves - computer, network, digital devices, mobile technology, Internet.
- By cyber criminals or hackers.
- Email frauds, Internet frauds, Identity frauds, theft of financial data, Malware attacks, Phishing, Distributed DoS attacks.
- Cyber Extortion - demanding money to prevent a threatened attack.
- Crypto Jacking - mine cryptocurrency

Some of the recent common crimes :- Email Spoofing, Sending Malicious File Applications, Social Engineering, Cyber Bullying, Identity Theft, Job (Banking) Frauds.

Forensic Process:-

- Collection
- Examination
- Analysis
- Reporting

Basic rules:-

1. Without altering and damaging data, We have to collect evidence.
2. Originally seized data size should be always equal to data we are analysis, we have authenticate evidence as same as data we about analyze.
3. Without modifications evidence, we have to analyze the data.

Types of Computer Forensics

1. **Database Forensics:-**

   It is the forensic investigation of databases and their metadata. By analyzing the data, timestamps of queries we applied in the database we are able to monitor the changes in the database. By doing so, we can identify what kind of data was stolen, modified, or corrupted by the attacker.

2. **Email Forensics:-**

   The recovery and analysis of emails and other information such as schedules, contacts, chats, conversations are contained in email platforms such as Gmail, Hotmail, Yahoo, Bingo, etc.,.

3. **Malware Forensics:-**

   Sifting through code to find and analyze potentially harmful programs. Trojan horses, ransomware, and different viruses are examples of such programs.

4. **Memory Forensics:-**

   Collecting information stored in a computer's random access memory (RAM) and cache.

5. **Mobile Forensics:-**

   Examining mobile devices in order to retrieve and analyze information such as contacts, incoming and outgoing text messages, photographs, and video files.

6. **Network Forensics:-**

   Monitoring network traffic with technologies like a firewall or intrusion detection system to look for proof.

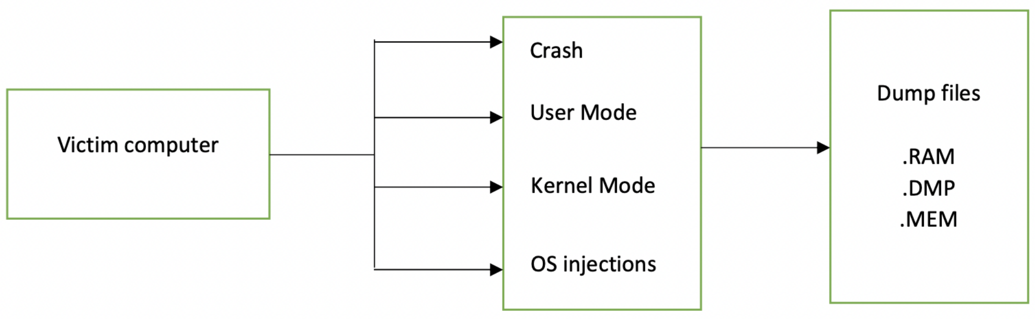


Figure 1.1: Memory Acquisition Techniques.

1.1.2 Memory Acquisition Techniques

There are so many techniques. we are mentioning some which are very widely and easily usable techniques.

Crash Dump:

If the machine goes down, the Windows operating system can program the file to be updated in the memory dump. The system state and main memory are intercepted, dumped on the offending machine, and the rest of the CPU information is stored in the machine registry for later analysis.

User Mode Technique:

User mode technique is used to get the instance data of the memory (RAM). Execution completes fairly quickly, controlling only the inclusion of processes in memory addresses. User-mode applications feature high availability (HA). It is recommended that the associated device be run using an external flash drive to reduce machine shortages and deal with Windows-based operating systems.

Kernel Based Technique:

Most of the investigators use this method. Because it will create multiple forensic copies of the RAM to deal with user-mode technique. Easy to implement by directly crashing the OS. To crash the OS, there are some settings we have to change in Windows. By doing so, it can create RAM Dump in ROM but it is unnamed. For naming and formatting that data we use nofoolexist.exe software to run. Then we can get .DMP file at the given location while we are doing settings. This entire process is known as Kernel mode exercise.

Operating System Injection:

This technique is also known as Body Snatcher. This technique is used in multiple cases like if the victim OS is uninstalled or affected by some characteristics. This method is independent of the OS. If the system has an OS, we perform it in that case, using software named OSforensics V9 (https://www.osforensics.com/download.html). Then run with administrative controls. Then it will dump what type of data we want to dump. In our case, we want physical memory. So, we get a .mem file.

1.1.3 File Carving

Carving is an low-level and an irreplaceable technique widely used in data recovery and digital forensics. By using carving, we essentially perform a low-level scan of the media for various artifacts, looking for signatures—specific sequences of bytes, characteristic of this or that type of data.

- Renamed, relocated, hidden data by performing carving, we ensure that files that have been slightly modified (e.g. change of original location, name, or extension) will be found on the media despite the changes the user applied in an attempt to conceal data. This also applies to data hidden inside containers and other forms of hidden data.

- Deleted data in unallocated space, free space, and slack space.

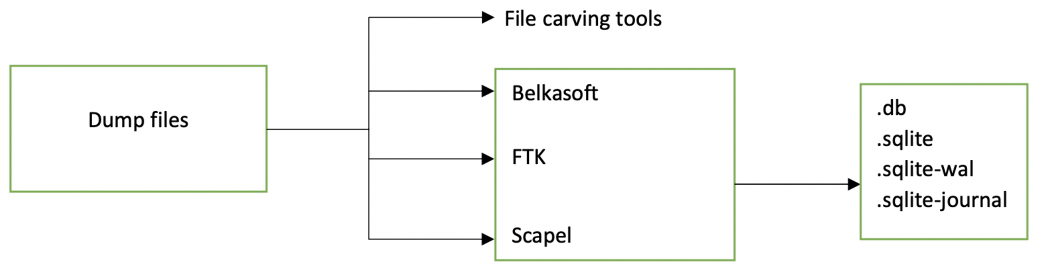


Figure 1.2: File Carving.

- Unallocated space is an another important implication of carving is searching for data inside slack space and unallocated space.
- Unallocated space is not recognized by the operating system and are generally unusable in regular means, unless you expand an existing partition or create a new one to include that space.
- We can see unallocated space in Disk Management.
- Free space: It is in allocated space. (*deleted file)
- Slack Space: It is also in allocate space. (*remnants files,hiding data)

We can carve disk, disk image. We can also do partial disk carving that present in hard drive and flash drive but we can not do in SSDs and EMMC.

File Carving Methods

we introducing some basic methods used for file carving. **Header Signature:** Carving looks for a stable signature of file header. header signature should be unique and longer.

Footer Signature:

Carving is looking for a footer signature, which determines the end of a file. It is less reliable because it can be easily override or located end of the disk due to fragmentation.

File Size:

Carving is done by fetching the file length from the header by looking for x bytes.we will give limit of fetching. Sometimes size is false in header.

File Carving vs Data Carving:

if header file is damaged due to any reason it impossible to get back, but by data carving it is possible.
Data carving is very efficient in terms of small data ...

Working of File Carving

Steps to carve the data

1. connect your data source to the computer to be used for running forensic analysis (data source must be carvable).
2. choose what you are going to carve, taking into account the amount of time you have, the types of data you are looking for, and the data source itself.(include both allocated space and unallocated space).
3. if you are short on time and/or looking for specific data, you can choose which parts of the disk which you want to carve.
4. you would need to analyze the results—whatever carving was able to find and recover.

Belkasoft, FTK and Scalpel is some popular carving tools.

1.1.4 SQLite Database Carving

It is the forensic investigation of databases and their metadata. By analyzing the data, timestamps of queries we applied in the database we are able to monitor the changes in the database. By doing so, we can identify what kind of data was stealed, modified,

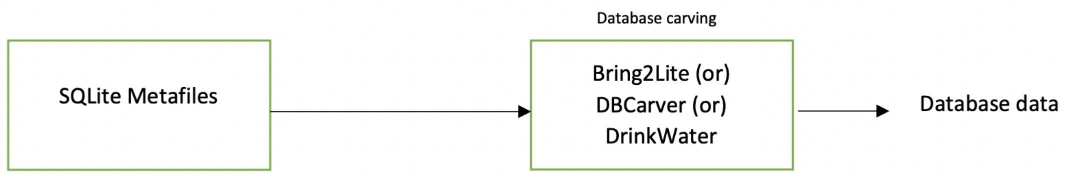


Figure 1.3: Database Carving.

or corrupted by the attacker. if some data is deleted in sqlite database, we can recover or restore it. Because if we delete a data in database it is not actually deleted but it is labelled as deleted. Sometimes, the deleted data stores at unallocated space or stored at same location but it is labelled as 'deleted'.

Let us explore SQLite Database,

Fundamentals of SQLite

It can shows how the data is stored in structural manner.

SQLite Pages:

Pages are very smallest entity which we can address the location in the whole database. It contains data(data in rows of a table). If the page is not enough to contain all row data of a table in a single page then the database will create similar sized pages to store the row data.

SQLite Pragmas:

Practically, it is not a default option. we have set this mode up then only, it enables. It is very useful mode in the forensic point view. Because:-

1. **secure_delete:-** The pragma secure_delete only has three options: 0, 1, and FAST. The FAST option is an unique setting for this pragma. As a result, forensic artefacts are on the freelist. Once this pragma is configured as 1, all deleted data and pages are replaced with zeros.

2. **auto_vacuum:-** The 2nd pragma is known as auto_vacuum. The database removes or deletes all useless or unused pages and does not keep in free list, if this pragma is on.

3. **journal_mode:-** It is also an optional mode in the SQLite database.It consists 6 options :

   (a) DELETE

   (b) TRUNCATE

   (c) PERSIST

   (d) MEMORY

   (e) WAL

   (f) OFF

   All of those alternatives create extra files with a "-journal" extension, with out the WAL and OFF switches. If the pragma is about to WAL, a document with the extension "-wal" could be created. The WAL document is not deleted in any case, however due to the fact it's far constant in size, SQLite overwrites the web page whilst the give up of the WAL document is reached.

SQLite File Format

we get this information form SQLite website [1] Generally, We all know that RDBMS files are organised in the B-tree data structured form. SQLite file is classified into two parts:

1. **Header:-**

   It occupies first 100 Bytes of SQLite file. It contains page size, database file size, text encoding, etc.,.

2. **Body:-**

   It starts with first page which contains structure known as SQLite master table.

Master table contains all type of info regarding the database such as table, index schemas, etc.,. through the page ID in first page.

Types of SQLite Pages

1. **Freelist Trunk Page:-**

   It is subdivided into 4 bytes page pointers. The first 4 bytes of the pointer will point to the another trunk page of the freelist. The page number of a freelist leaf page is indicated by each consecutive reference. At offset 32, the SQLite header stores a pointer to the first freelist trunk page.

2. **Freelist Leaf Page:-**

   It is a free page used to create or add new records. It contains only allocated data that can be stored.

3. **B-Tree Interior Page:-**

   It contains pointers which coonect with the children B-tree pages and to the right most child's page number.

4. **B-Tree Leaf Page:-**

   Leaf pages are the only page which contains active data.

Temporary files and Types

One of SQLite's distinguishing qualities is that each database is contained within a single disc file. SQLite is now also suitable for usage as an application file format. This makes SQLite easier to use as transferring or backing up a database is as simple as copying one single file. However, although a full database is stored in a single disc file, SQLite uses several temporary files to operate a database.
Nine distinct types of temporary files used by SQLite:

1. TEMP databases

2. Super-journals

3. Transient indices
4. Transient databases used by VACUUM
5. Rollback journals
6. Write-ahead Log (WAL) files
7. Shared-memory files
8. Materialisations of views and subqueries
9. Statement journals

WAL format

Write-Ahead-Log format is basically a roll-forward journal that is committed but not yet applied to the main database. It basically records transactions and its format details are described in the WAL format subsection of the main file format document. The file is used in place of the rollback journal when SQLite is operating in WAL mode. The main purpose of the WAL file is to implement atomic commit and rollback. The main purpose/ context behind WAL files is that after log records describing the changes have been flushed to permanent storage, the changes to data files where tables and indexes live must only be written after they have been recorded. The write-ahead logging (WAL) mechanism in SQL Server ensures that no data changes are written to disc before the accompanying log record is written to disc. This keeps a transaction's ACID characteristics intact. It is the industry standard methodology for ensuring that all database changes are appropriately logged in the sequence in which they occur. It assists in the preservation of data integrity by easing the recovery procedure in the event of a database crash.

Journal Files and Types

They are 3 type of journal files. They are

Super Journals:

The super-journal file is used as part of the atomic commit process when a single transaction makes modifications to numerous databases that have been added to a single database connection using the ATTACH command. All the associated auxiliary databases that were updated during the transaction are listed in the super-journal file. Once the super-journal file is destroyed, the multi-database transaction commits.

Rollback Journals:

The rollback journal is always in the same directory as the database file and has the identical name as the database file except for the addition of the eight letters ”-journal.” It is a temporary file that SQLite uses to enable atomic commit and rollback. Though there are exceptions to this rule, the rollback journal is usually created and destroyed at the start and end of a transaction.

Statement Journals:

If an alternate dispute resolution mechanism is utilized, the statement journal is omitted. It is only produced for a UPDATE or INSERT statement that may alter several rows in a database and may encounter a constraint or a RAISE exception within a trigger, requiring partial results to be undone. If the UPDATE or INSERT statement is not inside BEGIN...COMMIT and there are no other active statements on the same database connection, no statement journal is produced since the standard rollback journal can be utilized instead.

These are the most basic fundamentals we have to know to understand how our tool works in the backend.

1.2 Motivation

Attacks such as malware injections and other sorts of cybercrime may result in data loss, damage, or change. Also, there's a good probability that our own data may be utilized against us by the attacker under dubious circumstances, and so on. Finding out exactly what was done to our system might be difficult at times. Though there are numerous methods and techniques to eradicate this issue, there is always a scope of betterment and boost in accuracy of backtracking the cause of attack. This stood as a major motivation to come up with yet another technique which is hybrid in nature and promising as per the work.

1.3 Problem Statement

Developing a database forensics tool capable to extract and carve data, reconstruct it into the required file formats, visualize it in the best way understandable and capable enough to allow filtering of data to retain the required part(s).

1.4 Problem Objective

- The main objective is to understand the reason and working of the attack that happened, for which the data is carved in such a manner that all the suspicious logs present in the memory gets filtered accordingly.
- The filtered logs need to be reconstructed in order to make them suitable enough to perform visualizations.
- For this there are certain codes that are mentioned in the form of pseudo codes in this report.
- Using these, the data is converted into the required format, making it suitable enough to visualize. Then visualization techniques are applied on the data.

Literature Survey

Some researchers are working on our project related topics like Database Forensics, Memory Forensics, etc.,. We did some research on some new research models or structural analysis. we took some concept from research paper which we mentioned below.

Wagner et al.[2] developed a prototype, that demonstrate that the toolkit follows the state-of-the-art design used by current famous marketing tools and offers easy to interpret database artifact search capabilities. All forensic tools give the result format according to their file system or tool defaults, there are so many tools and file system. So, Wager came with a solution of the which standardizing the result format and structure for all file system, databases, file format. Wagner give a standard database storage format which is also known as Database Forensic File Format (DB3F). Wagner et al. created a tool which can used to view data and filter data in DB3F. That is named as Database Forensic Toolkit (DF-Toolkit), that enables the analysis of data stored in our database forensic format are presented.

From Wagner et al.[2], we need take some part of concept to our new tool in the sense of for viewing and filtering purposes. We are recreating a part of this tool concept as per our tool requirement.

Wagner et al.[3] developed the database carving tool which has a capability to recover the all data form the backup database file and also it can restore the deleted data. It has a ability to carve MySQL, PostgreSQl etc,., to recover all type of data.

Meng et al.[4] suggests making use of a structural approach; to analyze the deletion behavior of SQLite that depends on different database parameters or modes like secure_delete, auto_vaccum, journal_mode,etc.,. These things can affect database data in various ways. The main focus of Meng et al.[4] is on the investigation of the digital traces of SQLite deleted database data which is saved in various locations like freelists, unallocated space, etc.,. Meng et al.[4] applied his logic and implemented with common database datasets to prove accuracy is more than other database carving tools.

We are using same concept as Meng et al.[4] instead of using wagner et al.[3]. Because Wagner et al.[3] has ability to carve different types of database but we are only focusing on SQLite and also we want more recovery data accuracy. We choose Meng et al.[4] concept. so we can get more accurate recovered data.we are giving some other related researches.

Sangjun et al.[5] gives basic idea to recover the deleted data of database. The idea is to gather data from schema table regarding the database structure (B-tree structure). By using this info to search all datatype cells and match them with remaining cells.To confirm deleted cell is recover.

Ramisch et al. [6] did some research on retrieval of deleted. They restore by using expired indexes. Using we can expired records by using B-Tree Leaf pages (Index pages).By doing so,we retrieve data but there are sometime we can't be restore the SQLite database data.He included sanjun et al. [5] to get deleted data in different point of view. By analysis these data we can classify what data is exactly recovered by doing the process.

Liu et al. [7] came with a brilliant idea which mainly focuses on the WAL files (Write-

Ahead Log file which is extra file of main SQLite database file).The procedure examines both the page type and the serial kinds. Two databases were created to test the functionality of the built programme. The checkpoint to release data from the WAL to the SQLite database was not triggered on the first database, but it was triggered on the second. It give good result according to WAL file based only. But it is total extra all deleted cells.

Drinkwater et al.[8] was one of the first to explain a SQLite DBMS database carving approach (Drinkwater). The SQLite Parser from Guidance Software implements a lot of what Drinkwater talked about; it reconstructs both allocated and unallocated SQLite data (Guidance Software, 2018a). The results of SQLite Parser are returned in the form of a new SQLite instance (i.e., a single database file).

Case et al. [9] have presented current state of the memory forensics techniques which is evolved from many a long time but when we compared to evolution of Operating System designs is slow. That makes issues and limitation of every memory forensic analysis techniques are increase and also unable to detect some issues in memory acquisition.Some of the issues are Page smearing which is caused by using more than 8 GB RAM that gives non consistency memory capture means content change when we acquiring data form page tables. To avoid this problem we use smear-aware acquisition tools which makes continuous allocation and deallocation of kernel drivers while acquiring data from memory. And some other similar to it is Page Swapping, Demand Paging. In Windows, after updating the hibernation file is overwritten so it causes losing information about the previous process of RAM. Forensics lacking at detection of activity of powershell. In Linux android, both are similar when we compare kernels. Leveraging a kernel module database due rapidly increasing of different types of kernel modules or driver of several distributions of Linux operating systems.For avoid this issue we use Avoiding acquisition modules that hinder structured analysis. Case et al.[4] have also presented limitation of memory forensic in current generation technologies Apple ios, IoT, Chromebooks etc.

Lapso et al. [10] have represented the visualization forensic techniques which is helpful for investigator because usage of electronic devices increase than data storage in that device also increases than investigation process increases. By using memory visualization technique, we can able to see timeline visualization of non-volatile storage device by using triage method. By using InfoVis method with visualization tools, we are able to see local and global view in a single canvas. Addition to this, we apply whitelisting algorithm which will shirk interested type data form the visualization. This method is used for consistent and repetitive collected data in live computer. By doing this, the whitelist algorithm says that the application is affected by malware or not by comparing the same application from white list. If the process of application looks unique it will add to white list with AppID. Then percentage of occurrence of application to get more efficient visualization.

Dewald et al. [11] is developing an approach to identify and recover the files in a Ext4 file system which is used by the Linux operating system where the super blocker is overwritten or corrupted. Generally, super blocker stores the metadata(like number, size of blocks, no.of nodes, reserved blocks etc.,.) of the file system which is contained by the block groups (128 bytes). If superblocker is overwritten then the metadata is no good for recovering the file system. So, we use the file carving for specific file types, we carve for nodes which contain filename, timestamp etc.,. by using a pattern-based search method like heuristic search pattern. After using pattern search method then mostly got the file system but it contains duplicate files , false positive files, empty file, inodes with same parent inode or child inode , some files are not fully recovered. After using the sleuth kit tool which uses both content mode and metadata mode simultaneously, throughout the process of file system recovery. The process steps are Initialisation, Inode carving, Directory tree, Regular files and Files without content. Evaluation of the dataset is based on the parameters like timestamps, access rights, no.of hard links, extent flag and header, file type. These parameters are used in the pattern combination. Then we get the file system and verify the result with the original dataset. Then check runtime performance. It takes only 30 seconds to retrieve data (128 bytes) from a corrupted file system.

Sebastian et al. [12] presented a standardized corpus of SQLite files that can be used to evaluate and benchmark analysis methods and tools. The corpus contains databases that use special features of the SQLite file format or contain potential pitfalls to detect errors in forensic programs. The results show that there is no perfect tool for the analysis of SQLite, since all struggle with different peculiarities.

Lanterna et al. [13] gives the detailed explanation of forensic analysis of deduplicated file systems. Generally Deduplication reduces the data stored in the device which means removing the duplicate files. It is mainly used when backup or archive the data in the device.It is a post process for data saving. NTFS has the Microsoft Windows server which is add-on to the windows file system.The process is to split file into small fragments which is named as chunks but every should hash algorithm(Murmurhash3 used in OpenDedup, SHA256 used in windows file system).then store all chunks as chunk shore. Then it looks like a chunk sequence. Generally file systems contain nodes then more than one node will store in each chunk(var length). We can extract or identify the chunk by using the fingerprints on the Rabin fingerprint algorithm, which uses a sliding window to compare the original content and the content gathered from chunks. Then the result will be the common content. We can use that data as evidence in forensics.

Quadri et al. [14] have presented methods used to acquire data from secondary storage of victim machines. Here we use File System Forensic techniques, because most of the evidence is from the File system. The data/files storage structure in the file system is divided into 5 categories: 1. File System Category which can find locations of the other categorical data. Here we use Volume Slack which sizes of files contain to analyze for forensic 2.Content Category which contains storage locations of data saved files, it forms clusters of same sized files. Here we use cluster viewing technique to know the address of evidence. 3.Metadata Category which is descriptive data in address given by when cluster is allocated. Here we use consistency checks to find hidden data or internal errors at slack space which contain unused byte files allocated by OS. 4. Filename category which contains names of file here we use common investigation technique to get data and consistency check to confirm with other gathered metadata. 5.Application Category which unused data created by OS or crush occurred in OS while running application but by Journal of events (log files) and analysis this data we get some evidence.

Hejazi et al.[15] have presented new types of techniques to extract the sensitive information (User ID, name, address) from the windows physical memory. Application/Protocol footprint analysis uses FileZilla code to concat string(type of sensitive data ex. "password") and string for dialog box to a function as parameter. After processing we get all prefixed information of string, by classifying common applications source code it we could get footprints which are used for investigators.Call Stack analysis produces XML file where we can get sensitive function (in stack) and parameter. By locating every thread related stack and also stack frame for all functions in stack after understanding and analysis of functions we can construct an address table of process image.After comparing with forensic sensitive functions we can extract stack function parameters which means sensitive information.
Zoubek et al. [16] developed a practical approach to selectively delete data objects in a forensically sound manner. To face critical appraisals of selective deletion it is clearly stated that the approach presented here provides a forensically sound deletion process for data objects.

Hussain et al. [17] along with some research fellows, in 2013 worked on the objective of simulating a scenario in which there occurs a dark web cyber crime, and then analyzed the attack and acquired the evidence. Their work was mainly on TOR browser, since the location of the user remains unknown mostly. It was mainly to locate the hacker and observe his activities online. They used AndroKit, which provides a unified platform to perform android browser forensics. AndroKit can be a very useful component of cyber threat intelligence. It is also beneficial for web browser vendors for strengthening user data privacy. Apart from that, they analyzed registry, memory and storage on Windows 10 while on Android 10, and analyzed storage, zram (swap partition), and memory for potential artifacts.

Shree et al.[18] have presented the different techniques used in memory acquisitions from OS.They are two categories in memory acquisition :Hardware-based techniques and Software based techniques. In Hardware based techniques, 1.Steady hardware which is installed in physical devices (called "Tribble"). It is the best way to get data without interfering with the OS in the victim system.once tribal device activated malware is blocked by HOS. 2.Hardware bus : It is a very easy technique to collect data by using PCI, USB, FireWire. It uses the Hess mechanism to get raw data from the victim system. In Software-based techniques, 1.Virtualization which saves volatile data of VMware based VM with .vmem or .vmss in host machine running directory. 2. Crash Dump: to get a memory dump from RAM when the OS Suddenly stops. 3.User-mode exercise technique uses the PMDump tool to get the memory address of a victim machine and obtain a single address for RAM data. 4.Kernel- mode exercise is used to implement the kernel-mode software and applications which can get data from RAM. 5.Operating system Injection/Body Snatcher gives access to facilities to get rid of issues using software based mechanisms. This technique promotes the atomic memory images.In this paper they also mentioned some memory analysis mechanisms such as process and thread analysis, cryptographic key analysis, Network analysis, Open file analysis, System state analysis, and analysis tools which are available in the market.They also mentioned some malware analysis using memory forensics:main two types, 1. Static analysis which is very simple.it easily eliminates metadata of malware and collects important data of malware. 2.Dynamic analysis which is further classified into two phases. One is basic dynamic analysis which runs single time and analyzes the nature of malware and the other one is advanced dynamic analysis which makes the malware modify itself to become anti malware software.

Serhal et al. [19] aimed to explain a developed model that was trained and tested on a dataset consisting of the metadata of nearly 2 million files extracted from devices running Android OS and linked to real terrorism cases. This research effort presents a robust approach to triage files extracted from a smartphone, which can also be considered a proof of concept to develop ML-based triage tools for files extracted from smartphones. It also proposed the use of ML to classify files extracted from smartphones based on their metadata.

Popular tools, such as The Sleuth Kit, FTK, and EnCase uses data files to present this metadata.As a result, our data-base forensic storage format was created to store not only the records that can be retrieved live, but also the DBMS metadata, which users may not always have access to via the DBMS API.

There are several database carving tools available, but none of them produce a consistent output for storing their results. At the page level, these technologies investigate and rebuild database forensic artefacts. All row-store relational DBMSes use pages as the minimum read/write unit (usually 4 or 8 KB).

In a series of CSV files, DBCarver delivered much of the metadata as well as the allocated and unallocated user entries. Percona's Recovery Tool for InnoDB (Percona, 2018) restores MySQL DBMS files, but we don't consider it a forensic tool.

FTK and The Sleuth Kit store case information in SQL databases, and we also follow same approach.

The data must first be saved in a relational schema that is appropriately designed. Some forensic SQLite tools (for example, Guidance Software's SQLite Parser) output results as a SQLite DBMS file, which can be filtered directly with SQL. It does not, however, include the forensically significant metadata outlined by Wagner et al. (2015), which we believe should be included. As a result, just re-creating the database management system is insufficient because it only offers data and not metadata.

Proposed Approach

3.1 Our Contribution

We are introducing a tool named **"DB4n6 Tool"**,

- It has a ability to extract data from SQLite metadata files like .sqlite, .db, .sqlite-wal.
- It has the ability to carve the database and gives the data.
- The output format of result is JSON file but Data saved in standard format "DB3F" (by Wagner et al. [2]) and also gives LOG files.
- It has ability automaticly reconstruct the SQLite database.
- We can view data of reconstructed SQLite database.
- It has an ability to add DB3F formatted JSON files to view the data.
- we can also filter the data in SQLite database by using SQLite Queries on the database.

We are creating a tool named **"DB4n6"**, It has the ability to extract data from SQLite metadata files like .sqlite, .db, .sqlite-wal. It can carve the database and give out the relevant data. The output format of result is JSON file but the data is saved in standard format "DB3F" by Wagner et al.[2] which also includes LOG files. It can also automatically reconstruct the SQLite database. We can view data of reconstructed SQLite database. It

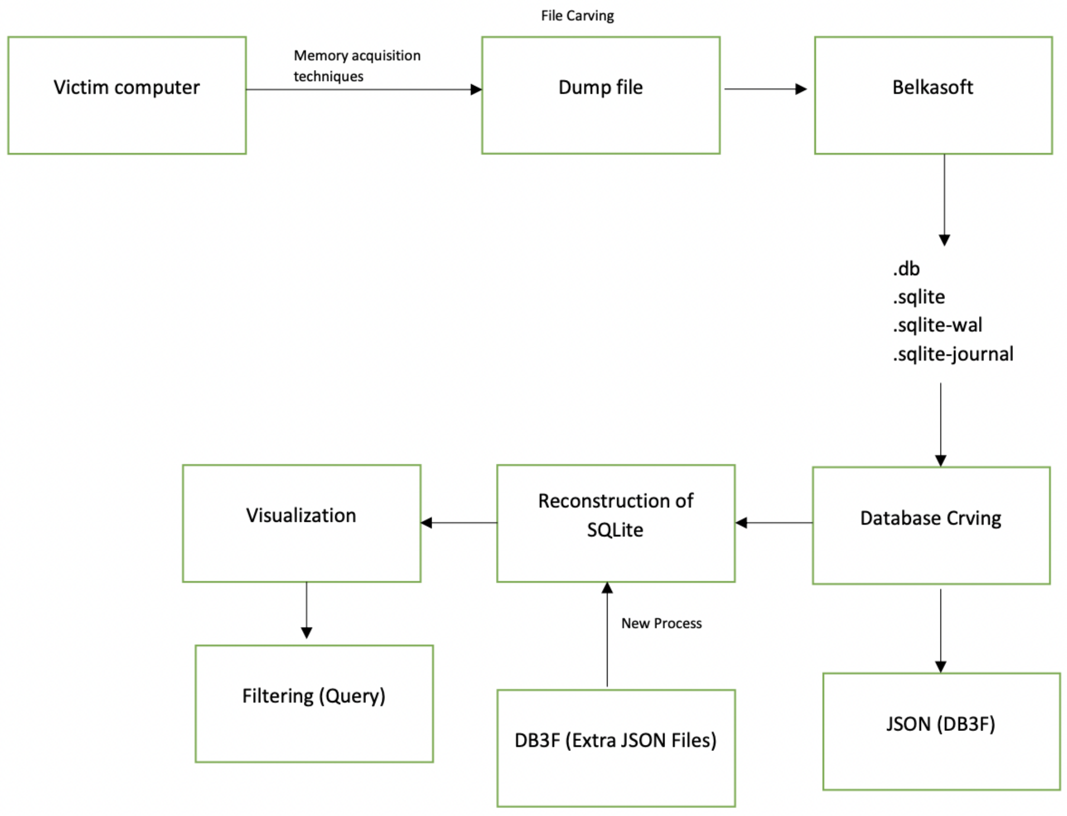


Figure 3.1: Model of DB4n6

has the capability to add DB3F formatted JSON files to view the data. We can also filter the data in SQLite database by using SQLite Queries on the database.

We are giving above mentioned all abilities to our tool DB4n6 but how it is getting these many abilities.

Now we can understand how DB4n6 works in the back-end.

3.2 DB4n6 Model

3.3 DB4n6 Working

Let us deep-dive into our tool.

3.3.1 Database Carving

We already discussed about the database carving in section[2.4]. Also we discussed about the related works of SQLite database carving in section[5.1].

Extract deleted content

Extraction of SQLite data from the SQLite master table, regular records, freeblocks, unallocated regions, freelist pages, and finally the journal files, WAL files and rollback are the various processing processes.

Extracting deleted content Most of the deleted data fall under the categories of unallocated data, freeblocks, freelist pages, journal files, WAL files, rollbacks, etc. Hence, the SQLite data from the SQLite Master table and these sources is extracted first. Then the SQLite header is to be processed and the master table needs to be parsed. This master table contains all the information about the tables and indices of the database and is stored always on the first page of the database file. The following Algorithm connects all pages of the database file to their respective schema, which is one of the important information about a database.

By running this algorithm, all the pages relate to their respected schema. The SQLite master table must be checked to see if it is a table b-tree interior page or a regular table b-tree leaf page. If the initial database page is a table b-tree interior page, all table b-tree leaf pages must first be obtained through page pointers and then parsed. Now the data is parsed using the outcome of the Algorithm and information of the cell pointer area in each b-tree page. Then from each active page of the database, deleted records that are stored in free blocks or as unallocated data are recovered. We know the schema of the table due to the outcome of the above algorithm, and hence it is possible to assume about the datatypes that are stored in a cell.

The total length of the header is calculated by the formula $L = v + s + b$ Where, v is

Algorithm 1 An algorithm to connect all in the schema

```
Input : schemas, page_size
Output :schemas connected to their pages
 1: $res = [\,]$
 2: for $i\ in\ schema$ do
 3:     if $i > 0$ then
 4:         $Page = extract\_page((i-1) * page_size)$
 5:         if Page == interior btree Page then
 6:             $res[i].append(Page.header.right\_most\_pointer)$
 7:             $cell\_count = Page.header.number\_of\_cells$
 8:             $cells = extract\_cells(Page, cell\_count)$
 9:             $res[i].append(cells)$
10:         else
11:             $res[i].append(i)$
12:         end if
13:     end if
14: end for
15: return $res$
```

Algorithm 2 An algorithm for extracting deleted data stored in freeblocks

```
Input: serial types, estimated header length, free block length
Output: set of solutions that could match the record based on schema
 1: $solutions = [\,]$
 2: $length = len\_freeblock\_contents(serial\_types)$
 3: $content = extract\_content(L, length)$
 4: $solutions.append(content)$
 5: return $solutions$
```

the length of first three variants within a cell s is the number of defined serial types in the schema of current page b is the number of datatypes The difficulty with this computation is that two variables are unknown in general. Because each of the versions has a maximum size of 9 bytes, the variable v can expand to a size of 27 bytes. Second, the schema does not reveal the magnitude of b. We iterate through the variable v, checking every possible value of b at each value of v until we hit a particular threshold. The extracted field of the cell, which is based on serial kinds, is compared to the data types maintained in the current page's associated schema in each iteration step.

Algorithm 3 An algorithm to extract unallocated data as the records of freeblock

```
Input: hexdump of current page
Output: processed unallocated area
1: $len\_header = get\_header\_length(page)$
2: $cellpointerarray\_length = get\_cellpointerarray\_length(page)$
3: $stop = get\_start(page)$
4: $area\_unalloc = extract\_unalloc(len\_header + cellpointerarray\_length, stop)$
5: $deleted\_record\_pointer = NULL$
6: for $x\ inrange(len(area\_unalloc))$ do
7:     $byte = extract\_byte(x)$
8:     if $byte == ""$ then
9:         $deleted\_record\_pointer = x$
10:        $break$
11:    end if
12: end for
13: $result = reduce\_area\_unalloc(area\_unalloc, deleted\_record\_pointer)$
14: return $result$
```

To represent database forensic artifacts, we are using JSON file type.

3.3.2 View and Search

For constructing, we take reference of evidence tree.

For filtering, after constructing database we use SQLite queries to filter the database data.

Evidence Tree

It is most forensic tools that exhibit forensic artifacts in a tree structure (e.g., FTK, The Sleuth Kit/Autopsy, and Encase) use a tree structure to present them. Because database forensic artifacts are fundamentally distinct from traditional forensic objects, they cannot serve as tree nodes.

3.3.3 Visualisation data in DB4n6

DB4n6 has a ability to view all level of data. For exmaple, we view Database level, Table level, Page Level and also Row level data.

Visibility

Forensic tools return a wide variety of data and metadata to users. These artifacts should be organized and presented to users in a manner such that the data can be traversed. This is traditionally done using a representative tree structure where the root nodes are the critical data structures (e.g., disk partitions), the next level nodes are used to store the data objects (e.g., stand-alone files), and all other node levels are used to store information about the carved data objects.

Display Data Objects

Given the ability to see a logical organisation of forensic artefacts in an evidence tree, the user would almost likely want to see the data objects and their contents. A user interface should allow for this type of viewing.

3.3.4 Filtering in DB4n6

Object Filtering

When a user is provided with a large number of data objects, she may wish to filter them down to a smaller subset that is useful to her. For example, if a user is only interested in JPEG files, the appropriate filtering criteria (filetype == ’JPEG’) can be used. A user in a database management system (DBMS) might want to filter objects based on metadata such object type (e.g., table, index, materialised view), number of columns, or object size.

Keyword Searches

In forensic investigations, keyword searches are frequently utilised to locate important evidence. For filtering records, string matches and regular expressions should be provided (e.g., find all mentions of 'Bob').

Reporting

Reports are required to assist analysts in reaching conclusions and presenting their findings. Furthermore, this reporting should make it possible to compare and validate the output of database forensic carving tools. Output is in JSON file.

3.4 Implementation

The data once considered is parsed multiple times during the processing part. Initially we decide several hash values using $.\_extract\_sqlite\_hashes()$. The data is collected and stored, and the header as well as the body of the data is parsed. This involves unpacking of the header and setting a frame offset to each dataframe created.
During the execution of sqlite hashes extraction, we use the header an the count of pages available as the main elements to decide the main page hashes. The parser class contains several methods for several purposes. For example,
$parse\_header()$
$parse\_body()$
$parse\_page()$

- After the parsing is done, the page type is returned but not as a final output. Then the cells are extracted in which the schema offset is taken as an input. The goal over here is to loop over every pointer to a cell.
The indices are traversed once after the other and the $multi\_variant$ function is run on all the cell types and indices. Then the overflow pages are calculated and extracted using

the overflow content.

- Now as far as the journal parsing is concerned, the filename, outname, format and page size are taken as inputs. Parsing is done at header and body level. The header padding is altered depending on the value of counter. The journal headers are unpacked and the page counts are calculated using the journal headers.

- In case of body level parsing, the size of checksum and pagenumber is set to 4. The page counter is configured and the page offsets are calculated by iterating through the pagenumbers. Then the result gets generated using the Journal Parser.

- Using $jsonconvert$ function, the directories are configured and os paths are defined. Generation of reports for freeblocks are also done and the data is decoded using $utf-8$.

- In terms of sqlite parsing, the filename, outname, format, and page size are now used as inputs. Parsing occurs at the header and body levels. Depending on the value of counter, the header padding changes. The sqlite headers are unpacked, and the page counts are determined with the help of the sqlite headers.

- Each bit in the sqlite header is classified under a particular format or element i.e., file format read and write versions, reserved and unused spaces at the end of each page, max and min embedded payload fraction, leaf payload fraction, size of database, schema cookies, page cache size, user version, application ids, etc. are aligned to each bit of the sqlite header and collectively they make one header array of size 20.

- Parsing in sqlite is done at freelist leaf pages level, freeblocks level, individual freelist level, etc. using the $PotentiallyParser()$ function. And here as well, report is generated and outname is configured using SQLite parser, path leaf and page numbers of freelist trunk pages.

- Then the cells are extracted using the estimated varint length, freeblocks and schema as inputs. Internally the record types are converted to schemas.

- Then there is a check that is done involving schema and types. By comparing the length of schemas and types, and the elements present in the lists of schemas and types, we return a Boolean that states whether the available schemas and types are the same or not.

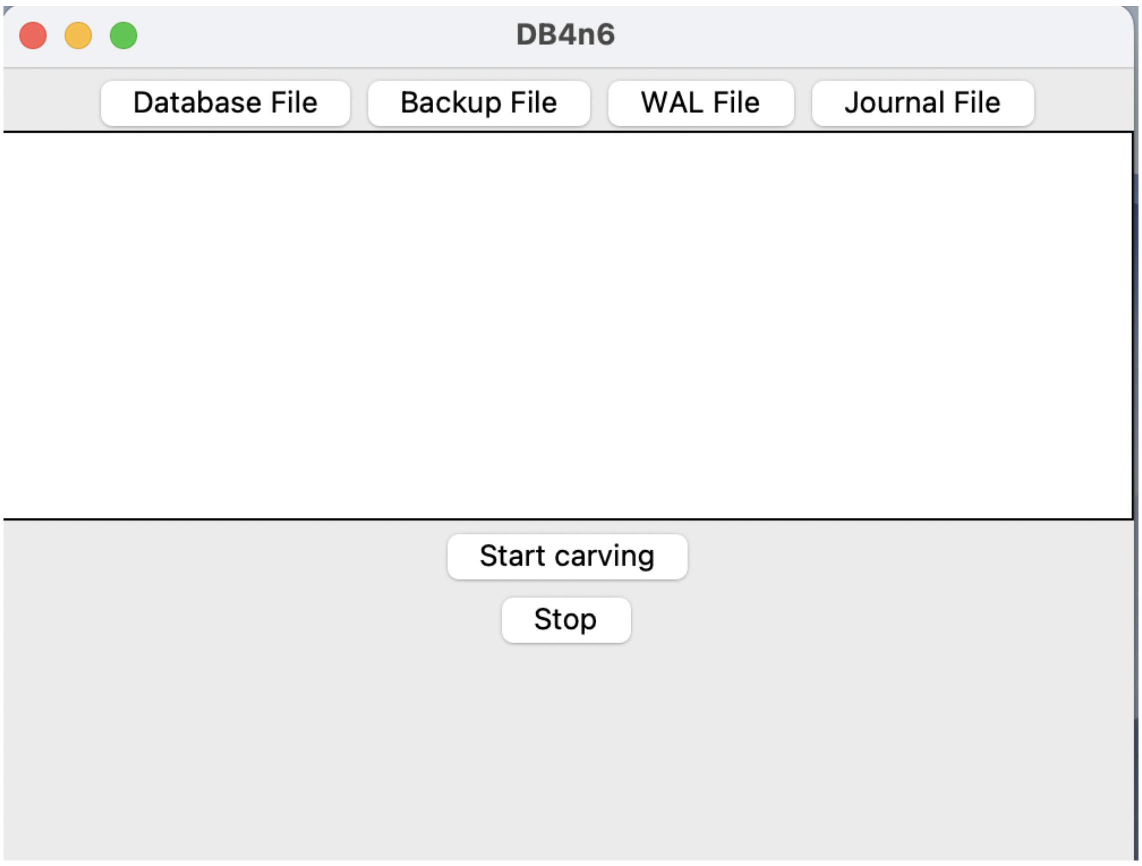


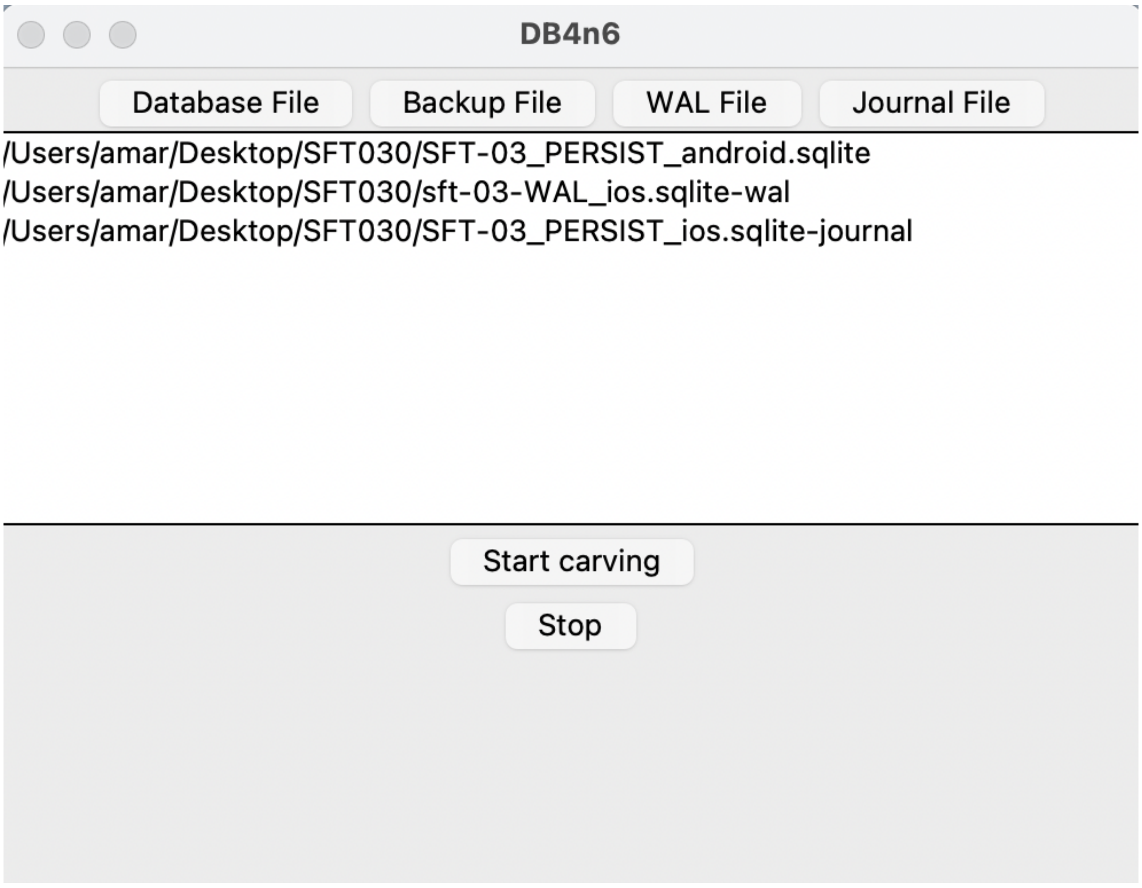
Figure 3.2: Home page of DB4n6

Then the schemas are extracted and decoded using $utf-8$. Thereafter the column names, deleted schemas are extracted row wise, according to the lines in SQL and cell offsets, and the schemas and pages are connected by iterating through the parsed schemas. This is done only one level in depth. Symbols are erased from the sql statements, and the page sizes are obtained using the Magic number and header size.

3.5 Case Study

3.5.1 Case 1

Hacker hacked the private organisation (like enterprises system, mobile) where SQLite database is widely used. If the system contain very important personal data of customers



/Users/amar/Desktop/SFT030/SFT-03_PERSIST_android.sqlite
/Users/amar/Desktop/SFT030/sft-03-WAL_ios.sqlite-wal
/Users/amar/Desktop/SFT030/SFT-03_PERSIST_ios.sqlite-journal

Figure 3.3: Inputting the file into DB4n6

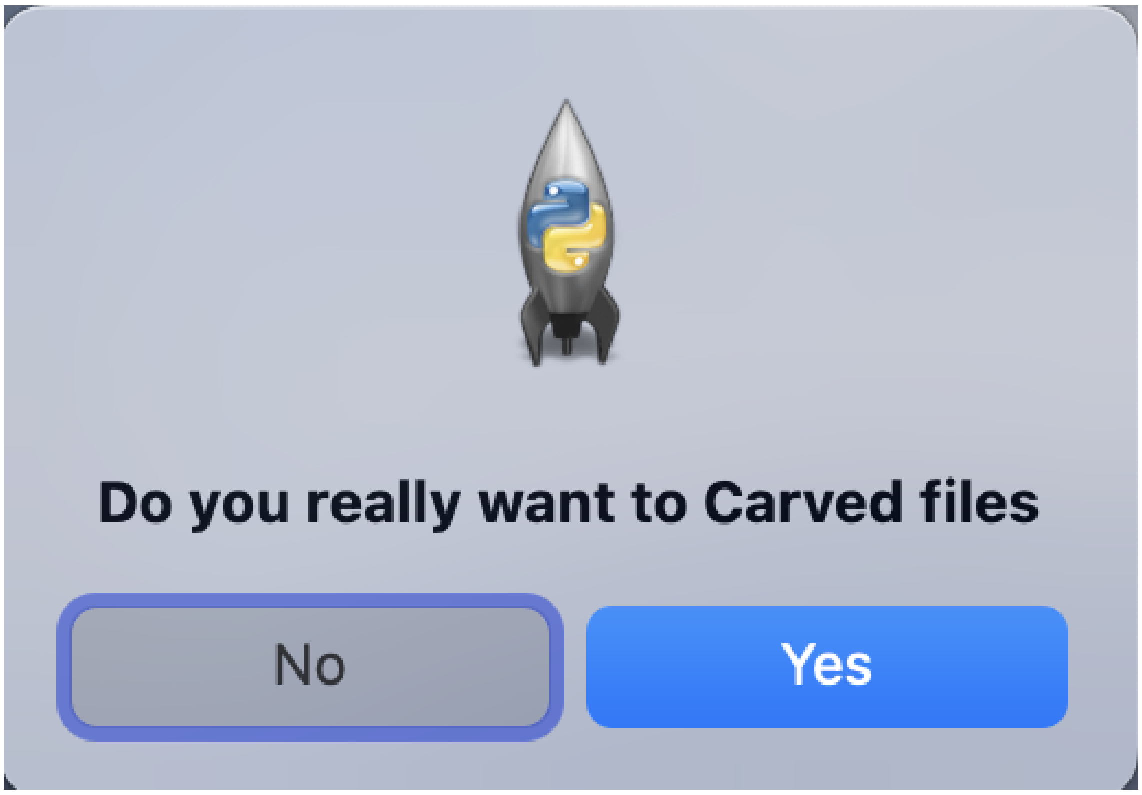
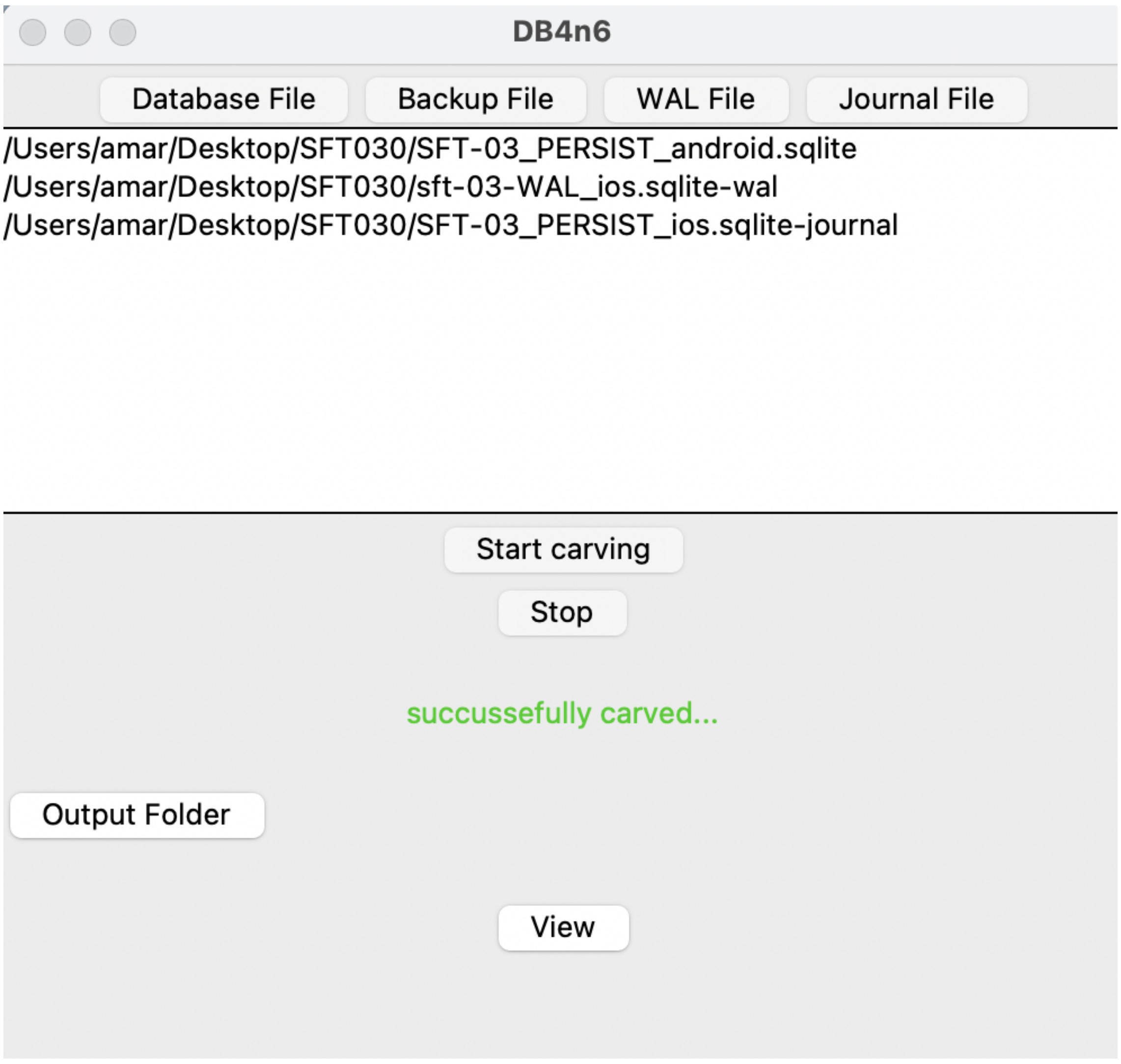


Figure 3.4: carved file in DB4n6

or user. We can acquire the data by using software acquisition techniques like crash dump, Operating system Injection, kernel mode exercise , user mode exercise. By doing so we get RAM snapshot or memory dump. So, we can carve those dump files to extract SQLite related metadata byte files like .db, .sqlite, .sqlite-wal, .sqlite-journal. Here, our tools comes into picture. Our tool DB4n6 first carve the database byte files to extract the data stored in it.

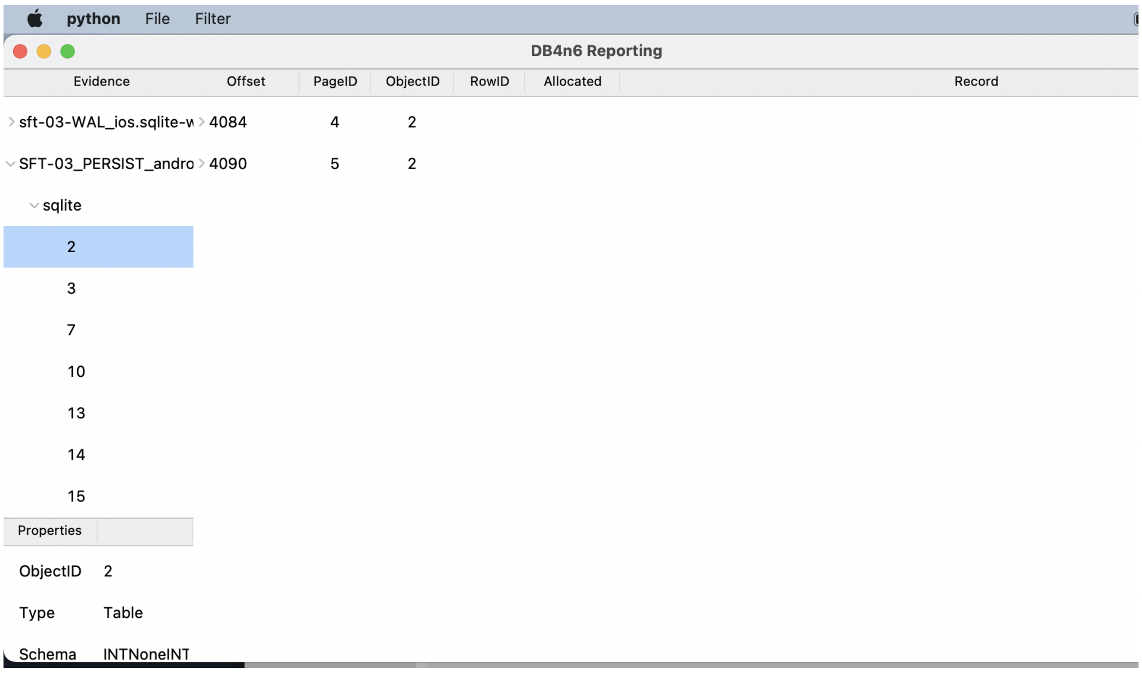
Case I If we have .db or .sqlite, we can follow the same procedure for both of the file format. In this file formats, we can restore the deleted data, missing data also. Firstly, we have to extract the header data, from that we get page size, write version, read version, un used space at end of the every page, page count, payload, number of freelist trunk pages, etc.,. Then we can use this information to extract schema of the page, offset and we can get the info which page ism belong to which table in database. Then entering deep into



/Users/amar/Desktop/SFT030/SFT-03_PERSIST_android.sqlite
/Users/amar/Desktop/SFT030/sft-03-WAL_ios.sqlite-wal
/Users/amar/Desktop/SFT030/SFT-03_PERSIST_ios.sqlite-journal

succussefully carved...

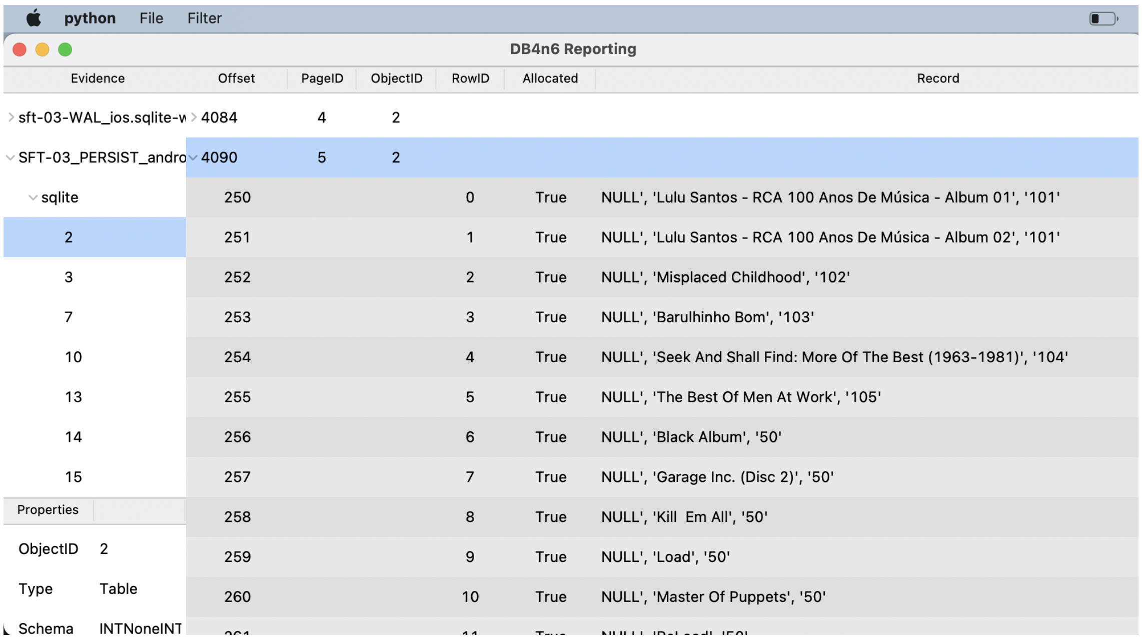
Figure 3.5: path of carved files in DB4n6



Evidence	Offset	PageID	ObjectID	RowID	Allocated	Record
sft-03-WAL_ios.sqlite-w	4084	4	2			
SFT-03_PERSIST_andro	4090	5	2			
sqlite						
2						
3						
7						
10						
13						
14						
15						

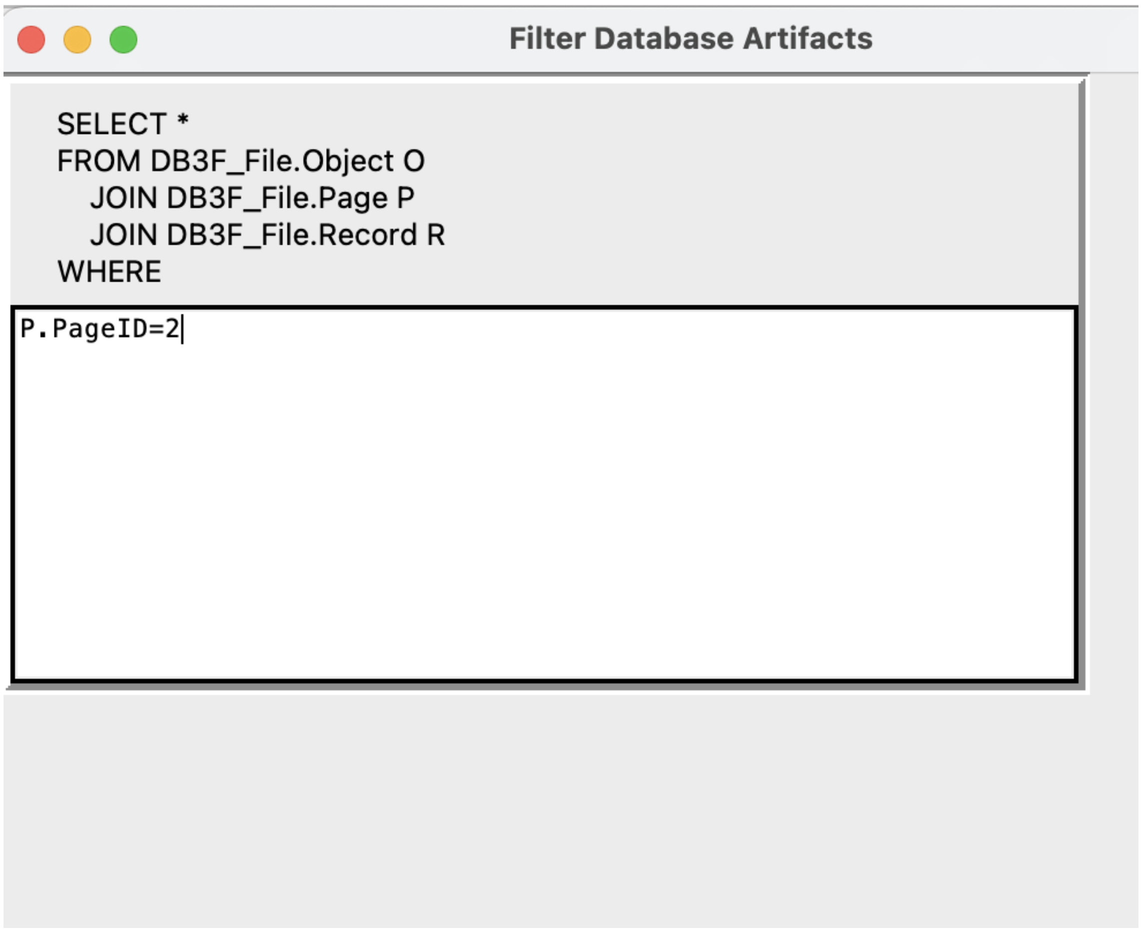
Properties	
ObjectID	2
Type	Table
Schema	INTNoneINT

Figure 3.6: View of page level data in DB4n6



Evidence	Offset	PageID	ObjectID	RowID	Allocated	Record
sft-03-WAL_ios.sqlite-w	4084	4	2			
SFT-03_PERSIST_andro	4090	5	2			
sqlite	250			0	True	NULL', 'Lulu Santos - RCA 100 Anos De Música - Album 01', '101'
2	251			1	True	NULL', 'Lulu Santos - RCA 100 Anos De Música - Album 02', '101'
3	252			2	True	NULL', 'Misplaced Childhood', '102'
7	253			3	True	NULL', 'Barulhinho Bom', '103'
10	254			4	True	NULL', 'Seek And Shall Find: More Of The Best (1963-1981)', '104'
13	255			5	True	NULL', 'The Best Of Men At Work', '105'
14	256			6	True	NULL', 'Black Album', '50'
15	257			7	True	NULL', 'Garage Inc. (Disc 2)', '50'
Properties	258			8	True	NULL', 'Kill Em All', '50'
ObjectID 2	259			9	True	NULL', 'Load', '50'
Type Table	260			10	True	NULL', 'Master Of Puppets', '50'
Schema INTNoneINT						

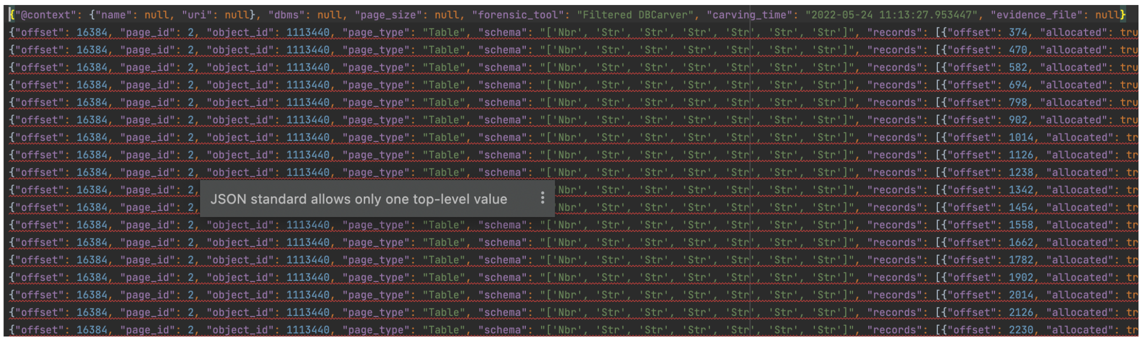
Figure 3.7: View of cell level data in DB4n6



```
SELECT *
FROM DB3F_File.Object O
   JOIN DB3F_File.Page P
   JOIN DB3F_File.Record R
WHERE
```

```
P.PageID=2
```

Figure 3.8: Filtering the data using query in DB4n6



```
{"@context": {"name": null, "uri": null}, "dbms": null, "page_size": null, "forensic_tool": "Filtered DBCarver", "carving_time": "2022-05-24 11:13:27.953447", "evidence_file": null}
{"offset": 16384, "page_id": 2, "object_id": 1113440, "page_type": "Table", "schema": "['Nbr', 'Str', 'Str', 'Str', 'Str', 'Str', 'Str']", "records": [{"offset": 374, "allocated": tru
{"offset": 16384, "page_id": 2, "object_id": 1113440, "page_type": "Table", "schema": "['Nbr', 'Str', 'Str', 'Str', 'Str', 'Str', 'Str']", "records": [{"offset": 470, "allocated": tru
{"offset": 16384, "page_id": 2, "object_id": 1113440, "page_type": "Table", "schema": "['Nbr', 'Str', 'Str', 'Str', 'Str', 'Str', 'Str']", "records": [{"offset": 582, "allocated": tru
{"offset": 16384, "page_id": 2, "object_id": 1113440, "page_type": "Table", "schema": "['Nbr', 'Str', 'Str', 'Str', 'Str', 'Str', 'Str']", "records": [{"offset": 694, "allocated": tru
{"offset": 16384, "page_id": 2, "object_id": 1113440, "page_type": "Table", "schema": "['Nbr', 'Str', 'Str', 'Str', 'Str', 'Str', 'Str']", "records": [{"offset": 798, "allocated": tru
{"offset": 16384, "page_id": 2, "object_id": 1113440, "page_type": "Table", "schema": "['Nbr', 'Str', 'Str', 'Str', 'Str', 'Str', 'Str']", "records": [{"offset": 902, "allocated": tru
{"offset": 16384, "page_id": 2, "object_id": 1113440, "page_type": "Table", "schema": "['Nbr', 'Str', 'Str', 'Str', 'Str', 'Str', 'Str']", "records": [{"offset": 1014, "allocated": tr
{"offset": 16384, "page_id": 2, "object_id": 1113440, "page_type": "Table", "schema": "['Nbr', 'Str', 'Str', 'Str', 'Str', 'Str', 'Str']", "records": [{"offset": 1126, "allocated": tr
{"offset": 16384, "page_id": 2, "object_id": 1113440, "page_type": "Table", "schema": "['Nbr', 'Str', 'Str', 'Str', 'Str', 'Str', 'Str']", "records": [{"offset": 1238, "allocated": tr
{"offset": 16384, "page_id": 2,  JSON standard allows only one top-level value  ⋮  Nbr', 'Str', 'Str', 'Str', 'Str', 'Str', 'Str']", "records": [{"offset": 1342, "allocated": tr
{"offset": 16384, "page_id": 2,                                                     Nbr', 'Str', 'Str', 'Str', 'Str', 'Str', 'Str']", "records": [{"offset": 1454, "allocated": tr
{"offset": 16384, "page_id": 2, "object_id": 1113440, "page_type": "Table", "schema": "['Nbr', 'Str', 'Str', 'Str', 'Str', 'Str', 'Str']", "records": [{"offset": 1558, "allocated": tr
{"offset": 16384, "page_id": 2, "object_id": 1113440, "page_type": "Table", "schema": "['Nbr', 'Str', 'Str', 'Str', 'Str', 'Str', 'Str']", "records": [{"offset": 1662, "allocated": tr
{"offset": 16384, "page_id": 2, "object_id": 1113440, "page_type": "Table", "schema": "['Nbr', 'Str', 'Str', 'Str', 'Str', 'Str', 'Str']", "records": [{"offset": 1782, "allocated": tr
{"offset": 16384, "page_id": 2, "object_id": 1113440, "page_type": "Table", "schema": "['Nbr', 'Str', 'Str', 'Str', 'Str', 'Str', 'Str']", "records": [{"offset": 1902, "allocated": tr
{"offset": 16384, "page_id": 2, "object_id": 1113440, "page_type": "Table", "schema": "['Nbr', 'Str', 'Str', 'Str', 'Str', 'Str', 'Str']", "records": [{"offset": 2014, "allocated": tr
{"offset": 16384, "page_id": 2, "object_id": 1113440, "page_type": "Table", "schema": "['Nbr', 'Str', 'Str', 'Str', 'Str', 'Str', 'Str']", "records": [{"offset": 2126, "allocated": tr
{"offset": 16384, "page_id": 2, "object_id": 1113440, "page_type": "Table", "schema": "['Nbr', 'Str', 'Str', 'Str', 'Str', 'Str', 'Str']", "records": [{"offset": 2230, "allocated": tr
```

Figure 3.9: Filtered data using query in DB4n6

the page the can find the page header and body. The page header contain the information about how many rows of data is stored in single page, row size, offset of page. By using header information, we can extract row data, row id, row offset, etc.,. Upto this point we get almost all data stored in the .sqlite or .db files. Now we have to extract data which is deleted or missing or saved in the unallocated space. We are using above mentioned 3 algorithms to extract all the deleted or missing data from freelist blocks and unallocated space. We can get all those data in standardised format DB3F. DB3F contains all data which are stored in the SQLite database in page level dictionaries stored in JSON file format.After we can construct SQLite database with the standard schema. Then we can view all the information. Then we can filter the page level database with SQLite query by doing so we can get all filtered data in JSON format which is used in further forensics to get info about hacker or what did he steal from the organisation.

Case II If we have .sqlite-wal, we are able apply forensic techniques on it. But It is impossible to recover all the data in the database. As we already know which type of data WAL file can store. We already it is just like a stack which collects the updates of data which are updated by queries. So, we can know data of every page which undergone though the update in SQLite database. So, we collect details by carving the frame which can contain the updated page data and frame size, etc.,. we can get the DB3F files which is in JSON. We can view Frame level data and also filter by using available parameters like pages , offsets, etc.,. we get filtered data in JSON file. Which very useful to find where our database is modified.

Case III If we have .sqlite-journal, we are able apply forensic techniques on it. But It is impossible to recover all the data in the database. As we already know which type of data Journal file contains. We already discussed about it in above sections. It stores the data of transactions which occurs while Queries are running on it. Transaction data are stored in pages. If that the case, as we already know to carve the data in database. So, we get the DB3F files which is in JSON. We can view page level data and also filter by using available parameters like pages , offsets, etc.,. we get filtered data in JSON file. Which very useful to find where our database is modified.

If we have all 4 file format we can analyse the filtered data the we can restart the database as well. And also we can find out what is the main object of hacker. Means what type of information he modified, steal, corrupted, personal data of the customers, clients, employees etc.,.

3.5.2 Case 2

Alan works as a database administrator for a government organisation that maintains track of residents' criminal histories. He was just convicted of fraud and wanted to take advantage of his powers by running DELETE FROM Record WHERE name 1/4 'Alan'. He is aware, however, that database operations are subjected to frequent audits to detect tampering with the agency's very sensitive data. Alan deactivates the audit log before performing the DELETE action to hide his tracks, then enables it again afterward. As a result, there is no record of his illicit database modification. However, until numerous storage artifacts generated by the deleted row are physically replaced, the database storage on disc will still hold evidence of the removed row. To detect such attacks and offer proof for how the database was modified, our technique identifies traces of deleted and outdated record versions and compares them to the audit metadata file or log files. The database is extracted from NIST n the file formats .db, .sqlite, .sqlite-journal, .sqlite-wal. Sqlite files are read and the deleted files that are unallocated are retrieved (by using carving). We would identify the deleted row using our method, and because it does not correlate to any action in the audit metadata files, we would flag it as possible evidence of manipulation. Certain filters can be configured to exactly know what are the files that are needed.

3.6 Comparison Table

Tools	**Input File Format**	**Output File Format**	**Visualization**	**Filter**
Bring2lite	.db, .sqlite, .sqlite-wal, .sqlite-journal	Log file(.log)	No	No
DBcarver	.db ,.sqlite	DB3F (.json)	No	No
Autopsy	.db, .sqlite	Excel (.xlsx)	Yes	Yes
Undark	.db, .sqlite	.csv	No	No
DB4n6	.db, .sqlite, .sqlite-wal, .sqlite-journal	DB3F (.json)	Yes	Yes

Conclusion and Future Work

4.1 Conclusion

The proposed model features some newer components wherein a certain scenario where an attack occurs, all the related data would be available in an apt formatted version after passing it through the developed tool. Hence, this model would not only be useful for the personnels with technical background, but to those who do not have a proper foundation in the forensics area. Moreover, the device owner, whether it be an organization or a customer, would be able to deduce certain conclusions after looking at the visualized data. This would make it more easy to use, understand and would in turn make the tool more reliable. The model works better with the free blocklists and unallocated data in a freespace, and in data modifications and retreivals. It shows better results than the tools that are already present for the same purpose, with an extra perk of visualization added.

4.2 Future Work

We can improve the tool in different ways. We are mentioning some of our ideas,

- Improving Carving technique to increase the accuracy of finding or retrieving the deleted data or modified data.
- In this tool we mainly focused on only one type of database SQLite, But we can improve our tool by enabling different types of databases like MySQl, PostgreSQl, etc.,.
- We can do some research on the NOSQL type databases.
- MongoDB, CouchDB, CouchBase, Cassandra, HBase are some of NOSQL databases.
- we can consider timestamps which is also plays a major roles while analyzing the queries we run on database

Bibliography

[1] "Sqlite documentation," https://sqlite.org/index.html.

[2] J. Wagner, A. Rasin, K. Heart, R. Jacob, and J. Grier, "Db3f & df-toolkit: The database forensic file format and the database forensic toolkit," *Digital Investigation*, vol. 29, pp. S42–S50, 2019.

[3] J. Wagner, A. Rasin, T. Malik, K. Heart, H. Jehle, and J. Grier, "Database forensic analysis with dbcarver," in *CIDR 2017, 8th Biennial Conference on Innovative Data Systems Research*, 2017.

[4] C. Meng and H. Baier, "bring2lite: a structural concept and tool for forensic data analysis and recovery of deleted sqlite records," *Digital Investigation*, vol. 29, pp. S31–S41, 2019.

[5] 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.

[6] F. Ramisch and M. Rieger, "Recovery of sqlite data using expired indexes," in *2015 Ninth International Conference on IT Security Incident Management & IT Forensics*. IEEE, 2015, pp. 19–25.

[7] Y. Liu, M. Xu, J. Xu, N. Zheng, and X. Lin, "Sqlite forensic analysis based on wal," in *International Conference on Security and Privacy in Communication Systems*. Springer, 2016, pp. 557–574.

[8] R. Drinkwater, "Carving sqlite databases from unallocated clusters," *Internet: http://forensicsfromthesausagefactory. blogspot. se/2011/04/carving-sqlitedatabases-from. html*, vol. 14, p. 00, 1916.

[9] A. Case and G. G. Richard III, "Memory forensics: The path forward," *Digital Investigation*, vol. 20, pp. 23–33, 2017.

[10] J. A. Lapso, G. L. Peterson, and J. S. Okolica, "Whitelisting system state in windows forensic memory visualizations," *Digital Investigation*, vol. 20, pp. 2–15, 2017.

[11] A. Dewald and S. Seufert, "Afeic: Advanced forensic ext4 inode carving," *Digital Investigation*, vol. 20, pp. S83–S91, 2017.

[12] S. Nemetz, S. Schmitt, and F. Freiling, "A standardized corpus for sqlite database forensics," *Digital Investigation*, vol. 24, pp. S121–S130, 2018.

[13] D. M. Divakaran, K. W. Fok, I. Nevat, and V. L. Thing, "Evidence gathering for network security and forensics," *Digital Investigation*, vol. 20, pp. S56–S65, 2017.

[14] S. Quadri and W. A. Bhat, "A brief summary of file system forensic techniques," in *Proceedings of 5th national conference, INDIACom*. Citeseer, 2011, pp. 499–502.

[15] S. M. Hejazi, C. Talhi, and M. Debbabi, "Extraction of forensically sensitive information from windows physical memory," *digital investigation*, vol. 6, pp. S121–S131, 2009.

[16] C. Zoubek and K. Sack, "Selective deletion of non-relevant data," *Digital Investigation*, vol. 20, pp. S92–S98, 2017.

[17] M. R. Arshad, M. Hussain, H. Tahir, S. Qadir, F. I. A. Memon, and Y. Javed, "Forensic analysis of tor browser on windows 10 and android 10 operating systems," *IEEE Access*, vol. 9, pp. 141 273–141 294, 2021.

[18] R. Shree, A. K. Shukla, R. P. Pandey, V. Shukla, and D. Bajpai, “Memory forensic: Acquisition and analysis mechanism for operating systems,” *Materials Today: Proceedings*, vol. 51, pp. 254–260, 2022.

[19] C. Serhal and N.-A. Le-Khac, “Machine learning based approach to analyze file meta data for smart phone file triage,” *Forensic Science International: Digital Investigation*, vol. 37, p. 301194, 2021.