IJCSE International Journal of Computer Sciences and Engineering Open Access

Research Paper Vol.-7, Issue-1, Jan 2019 E-ISSN: 2347-2693

Identification of Tampered SMS Messages in iPhone – A Case Study

Eswara Sai Prasad Chunduru[1*], Krishna Mangarai[2], Subrahmani Babu[3], Manohar B Pathak[4]

[1, 2, 4]Computer Forensic Unit, Central Forensic Science Laboratory, Hyderabad
[3]Deloitte Shared Services India LLP, Bengaluru

Available online at: www.ijcseonline.org

Accepted: 22/Jan/2019, Published: 31/Jan/2019

Abstract: Mobile Phones, now-a-days have become the primary source of evidence in any type of investigation for providing initial leads for framing the future investigation as well as primary evidence. The retrieval and analysis of SMS messages, including deleted ones from the memory of smart mobiles is one of the main process of mobile phone forensics. With the evolution of technology, criminals can manipulate the SMS database and tamper the messages to prove their virtue and mis-guide the investigation. In the current paper we have discussed the method adopted for proving the authenticity of SMS messages retrieved during the analysis of an iPhone.

Keywords: SMS Messages, iPhone, Tampering, Authenticity, iOS

I. INTRODUCTION

The use of mobile communication devices has increased rapidly over the years. These wireless communication devices were started as devices to store individual's personal information and transformed into hand held computers, providing many services to the users. Short message service (SMS) is one such popular wireless service throughout the globe facilitating a user to be in touch with his counterparts. This service is playing a major role in the current day to day walks of life starting from sharing information to bonding persons to performing transactions related to financial, educational, governmental via m-commerce, m-banking, m-governance etc. As we know that the technology is double edged weapon, with these conveniences there bear amenities through which fallacious benefits are gained by way of manipulation of these services. Proving the authenticity and veracity of these services sometimes may become the thrust of investigation and analysis.

The forensic analysis of Mobile phones primarily involves retrieving the Phone basic information i.e., Phone Book Entries, Call Logs and SMS Text messages. Current day Mobiles Phones store these entries basically in SQLite Database formats. These SQLite databases are prone to manipulation. Unwanted mileage can be gained by inserting one or more SMS Text messages fraudulently as part of the database of the Mobile Phones. Forensic analysis of the Mobile Phones simply retrieves the SMS Text Messages from the SQLite Database file, but unable to establish the authenticity of the SMS database. In the present paper we have tried to discuss a methodology to analysis the SQLite database file i.e., ***sms.db*** file retrieved from an iPhone for determining its authenticity.

II. BACKGROUND

Short Message Service (SMS) is a text messaging service component of mobile communication systems. It uses standardized communications protocols to allow devices to exchange short text messages. SMS was the most widely used data application. This service, using standardized phone protocols has been adopted into the modern mobile communication devices from radio telegraphy in radio memo pagers. The protocols were defined in 1985 as part of the Global System for Mobile Communications (GSM). Initially, these protocols were for mobile communication devices working on GSM technology, allowing exchanging messages of up to 160 characters between GSM mobile handsets. Later support for the service has been expanded to include other mobile technologies, such as ANSI (American National Standards Institution) CDMA (Code Division Multiple Access) networks and Digital AMPS (Advanced Mobile Phone System), as well as satellite and landline networks. The SMS services began in the early 1980s. The first action plan of the Conference of European Post and Telecommunications (CEPT) Group GSM was approved in December 1982. This plan included the exchange of text messages either directly between mobile stations or transmitted via message handling systems in use at that time.

The SMS concept was developed in the Franco-German GSM cooperation in 1984 by Friedhelm Hillebrand and Bernard Ghillebaert. Since telephony was identified as the major application used those days, GSM was optimised for the same. SMS has been implemented by using this telephone-optimized system, to transport text messages when no telephony signals existed. Initially the length of messages has been limited to 128 bytes and later enhanced to 160 seven-bit characters so that the messages could fit into the existing signalling formats. Based on observations and on analysis of the typical lengths of postcard and Telex messages, 160 characters were enough to express most of the messages concisely. (Hillebrand).

The Global Service for Mobile communications (GSM), has several security vulnerabilities. In the GSM, only the airway traffic between the Mobile Station (MS) and the Base Transceiver Station (BTS) is optionally encrypted with a weak and broken stream cipher (A5/1 or A5/2). The validation is unilateral and susceptible. Such vulnerabilities are inherent to SMS. In addition to these, SMS messaging has some extra security vulnerabilities due to its store-and-forward feature, and the problem of fake SMS can be possible by exploring these vulnerabilities.

As the use of smart phones is on the rise, the same being used as a tool to commit crime or store data related to an act or manipulate the data stored to have a wrongful advantage. It is to be accepted, today whenever a crime is reported the Investigating agencies first piece of evidence gathering is starting at a mobile phone. The investigation starts by collecting the Call Logs/CDRs, Phone Book and SMS Messages from the service provider and from the phone itself. In many investigations the SMS messages retrieved from the phone play vital role in resolving the cases.

In old days phones, as the internal memory is limited the amount of space allocated for the storage of SMS messages is also limited and the file system to maintain these messages is also native to the Operating System of the phone. The arrangement of SMS messages is mainly on random indexing concept. The vulnerabilities associated with the SMS messages is primarily network based. Sometimes the SMS messages can be manipulated by framing the desired sender or receiver or text with the help of third-party tools. Authenticating the SMS messages in these cases is very difficult as the storage sequence or ordering of the SMS messages is random in nature.

Currently the SMS messages in the most modern mobile phones are maintained in the SQLite format. The storage of the SMS messages in the SQLite makes the arrangement of SMS messages well defined by the database principles of the SQLite. This gives the advantage of maintaining, migrating, indexing, logging of the activities within the database flexible across various mobile operating systems.

This forensic analysis of the SMS text messages in SQLite format can give an investigator to identify the offline manipulations if any performed with the Text messages. In the current case the forensic analysis of the ***sms.db*** retrieved from the backup of an iPhone reveals that the failure of the authenticity of the SMS text messages parsed.

III. METHODOLOGY

In the present case an iPhone 4S with Model No. A1387 running iOS version 7.0 was referred for verifying the SMS messages for their veracity. The backup of the iPhone was taken with the help of iTunes version 11.3 on a system running Microsoft Windows Vista Home Premium Operating System.

The backed-up data is stored in a preconfigured folder for each of the operating systems. In the current case as the operating system is Microsoft Windows Vista the location of the backup file is as follows:

\Users\(username)\AppData\Roaming\AppleComputer\MobileSync\Backup\

The UDID (Unique device ID – 40 hexadecimal characters long) of the backup folder is as follows:

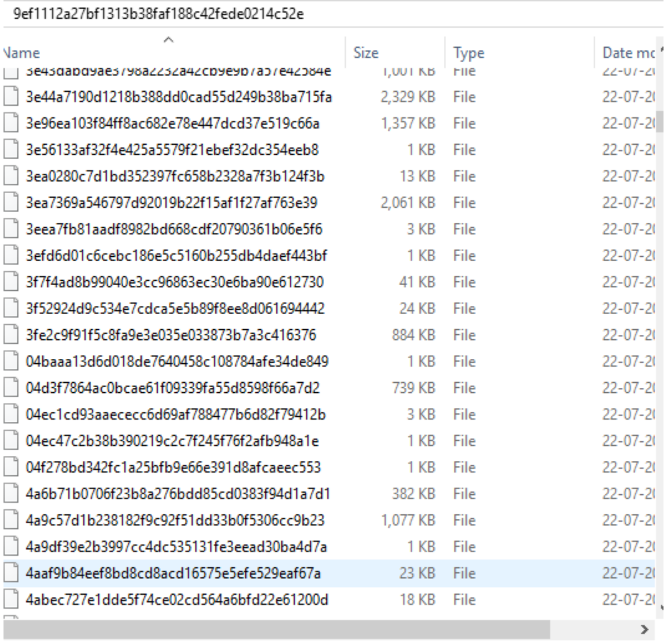
9ef1112a27bf1313b38faf188c42fede0214c52e

The above iTunes backup folder contains hundreds of files which are not in readable format and are uniquely named with a 40-digit alphanumeric hex value without any file extension.

Example:

0b11910eaa43489d869902c3081301cde166472c

The screen shot showing the portion of the backup folder is shown at ***Fig. 01***.



9ef1112a27bf1313b38faf188c42fede0214c52e

Name	Size	Type	Date mo
3e43dabd9ae3798a2232a42cb9e9b7a57e42584e	1,001 KB	File	22-07-20
3e44a7190d1218b388dd0cad55d249b38ba715fa	2,329 KB	File	22-07-20
3e96ea103f84ff8ac682e78e447dcd37e519c66a	1,357 KB	File	22-07-20
3e56133af32f4e425a5579f21ebef32dc354eeb8	1 KB	File	22-07-20
3ea0280c7d1bd352397fc658b2328a7f3b124f3b	13 KB	File	22-07-20
3ea7369a546797d92019b22f15af1f27af763e39	2,061 KB	File	22-07-20
3eea7fb81aadf8982bd668cdf20790361b06e5f6	3 KB	File	22-07-20
3efd6d01c6cebc186e5c5160b255db4daef443bf	1 KB	File	22-07-20
3f7f4ad8b99040e3cc96863ec30e6ba90e612730	41 KB	File	22-07-20
3f52924d9c534e7cdca5e5b89f8ee8d061694442	24 KB	File	22-07-20
3fe2c9f91f5c8fa9e3e035e033873b7a3c416376	884 KB	File	22-07-20
04baaa13d6d018de7640458c108784afe34de849	1 KB	File	22-07-20
04d3f7864ac0bcae61f09339fa55d8598f66a7d2	739 KB	File	22-07-20
04ec1cd93aaececc6d69af788477b6d82f79412b	3 KB	File	22-07-20
04ec47c2b38b390219c2c7f245f76f2afb948a1e	1 KB	File	22-07-20
04f278bd342fc1a25bfb9e66e391d8afcaeec553	1 KB	File	22-07-20
4a6b71b0706f23b8a276bdd85cd0383f94d1a7d1	382 KB	File	22-07-20
4a9c57d1b238182f9c92f51dd33b0f5306cc9b23	1,077 KB	File	22-07-20
4a9df39e2b3997cc4dc535131fe3eead30ba4d7a	1 KB	File	22-07-20
4aaf9b84eef8bd8cd8acd16575e5efe529eaf67a	23 KB	File	22-07-20
4abec727e1dde5f74ce02cd564a6bfd22e61200d	18 KB	File	22-07-20

Fig. 01: Contents of the iTunes Backup folder

These files contain copy of everything on the iPhone under analysis including contacts, SMS Text Messages, photos, calendar, music, call logs, configuration files, database files, keychain, network settings, offline web application cache, safari bookmarks, cookies and application data, etc. It also contains the device details like serial number, UDID, SIM hardware number and the phone number.

The files found in the above backup directory can be classified into five categories:

- SQLite3 database files
- Plain text plist files
- Binary plist files
- Multimedia and text files
- Non-standard data files

The SQLite3 database files store a single database each in SQLite3 format. Each database can contain an arbitrary number of tables.

The plain text plist files are Extensible Markup Language (XML)-like text files.

Their binary counterparts are XML-like files stored in binary format which can be easily converted back to a plain text format with the Mac OS X plutil utility.

In addition to the files described above, the backup directory contains five more standard files, called meta files with a fixed name

- Info.plist,
- Manifest.plist,
- Status.plist,
- Manifest.mbdb and

A. Info.plist:

This is property list file containing the device details like device name, build version, IMEI, phone number, last backup date, product version, product type, serial number, sync settings and a list of application names that were installed on the device, etc.

B. Manifest.plist:

This is property list file containing the third-party application bundle details, Backup Keybag, a flag to identify the passcode protected devices (WasPasscodeSet) and a flag to identify the encrypted backup (IsEncrypted), etc.

C. Status.plist:

This is property list file containing the details about the backup. It includes backup state, a flag to identify the full backup (IsFullBackup), date and version, etc.

D. Manifest.mbdb:

This binary file contains information about all other files in the backup along with the file sizes and file system structure data.

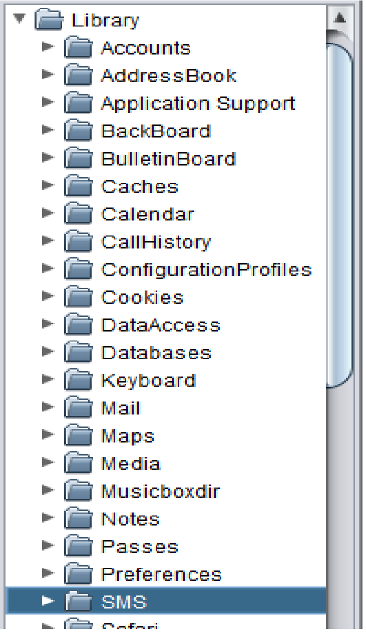
In older version of iTunes, the backup file structure is managed by Manifest.mbdb and Manifest.mbdx. The Manifest.mbdx file acts as an index file for the backup and indexes the elements that will be found in Manifest.mbdb. From iTunes 10, index file (mbdx) is eliminated and the backup is managed by a single mbdb file.

The iPhone stores a lot of user data in the backup files. Each of the 40 digit hex file name shown in ***Fig. 01*** is the SHA1 hash value of the file path appended to the respective domain name with a '-' symbol. So, the hash of DomainName-file path will match to the correct file in the backup. In the backup folder of iTunes applications and inside data are classified into 12 domains (11 system domains and one application domain). iTunes stores/reads the domain names and path names from Meta files i.e., Info.plist, Manifest.plist, Status.plist and Manifest.mbdb. The categorization of backup files is described by their domain. The domain for each file is written in its corresponding record in the Manifest.mbdb file. Each file has a domain name chosen from the following list:

- Application domain.
- Home domain.
- Keychain domain.
- Managed Preferences domain.
- Media domain.
- Mobile Device domain.
- Root domain.
- System Preferences domain.
- Wireless domain.

The application domain will contain number of sub domains each related to the specific application installed in the iPhone. The sub domain is further divided in to directories with standard and similar structure across various applications.

In file with 40 digit hex file name ***3d0d7e5fb2ce288813306e4d4636395e047a3d28 (It may be noted that the file with this 40 digit hex name in any iTunes backup folder always contains data of SMS messages)*** in the backup folder with UDID (Unique device ID) ***9ef1112a27bf1313b38faf188c42fede0214c52e*** has the data related to SMS messages, *to be analysed for their veracity in the current case*, in one of the ***directories*** of the sub domain ***library*** of ***home domain*** as shown in ***Fig. 02***.

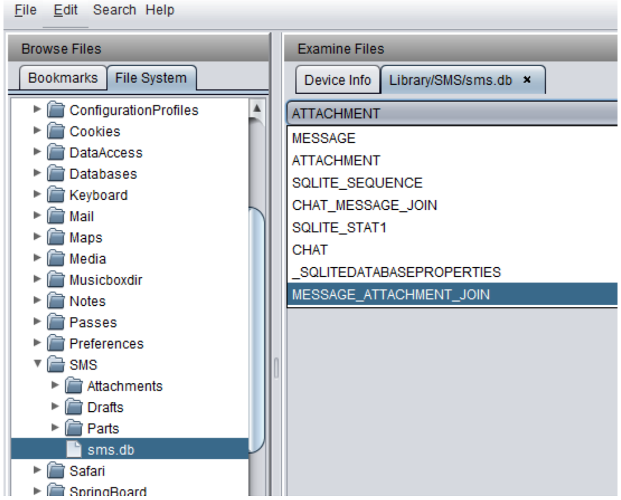


Library
Accounts
AddressBook
Application Support
BackBoard
BulletinBoard
Caches
Calendar
CallHistory
ConfigurationProfiles
Cookies
DataAccess
Databases
Keyboard
Mail
Maps
Media
Musicboxdir
Notes
Passes
Preferences
SMS
Safari

Fig. 02: Directories under the sub domain library

The SMS database ***sms.db*** contains a series of tables that store different pieces of data. These tables have relationships through which the SMS messages or conversations are made up.

The screenshot of various tables in the ***sms.db*** file are shown in ***Fig. 03***.



ATTACHMENT

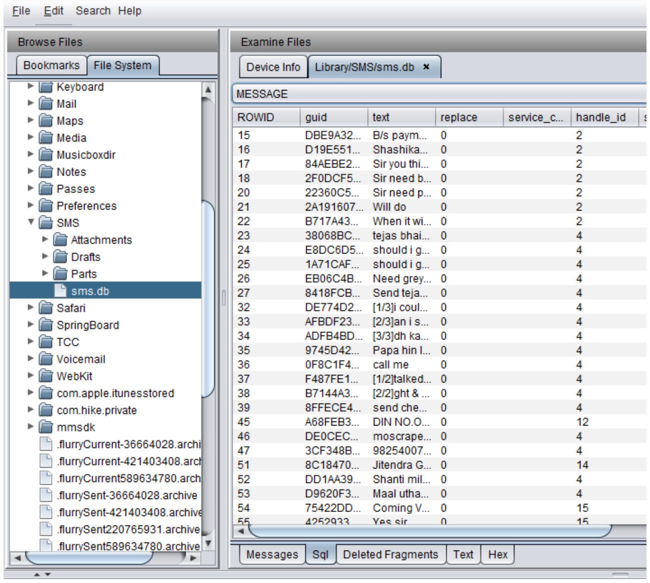
MESSAGE
ATTACHMENT
SQLITE_SEQUENCE
CHAT_MESSAGE_JOIN
SQLITE_STAT1
CHAT
_SQLITEDATABASEPROPERTIES
MESSAGE_ATTACHMENT_JOIN

Fig. 03: List of tables in sms.db

Each table shown in the ***Fig. 03*** have specific columns, and the tables are interconnected to each other in a specific structural relationship to form the complete database. The description of these tables as to what and how the different pieces of data related to SMS messages in an iPhone is arranged is described below:

E. Message Table:

This is the main table where all the messages are stored. The message table contains the content of each SMS message as well as details of the sender or recipient, the message thread that the message belongs to, and the status of the message. The portion of the screen shot of the message table is shown in ***Fig. 04***.



MESSAGE

ROWID	guid	text	replace	service_c...	handle_id
15	DBE9A32...	B/s paym...	0		2
16	D19E551...	Shashika...	0		2
17	84AEBE2...	Sir you thi...	0		2
18	2F0DCF5...	Sir need b...	0		2
20	22360C5...	Sir need p...	0		2
21	2A191607...	Will do	0		2
22	B717A43...	When it wi...	0		2
23	38068BC...	tejas bhai...	0		4
24	E8DC6D5...	should i g...	0		4
25	1A71CAF...	should i g...	0		4
26	EB06C4B...	Need grey...	0		4
27	8418FCB...	Send teja...	0		4
32	DE774D2...	[1/3]i coul...	0		4
33	AFBDF23...	[2/3]an i s...	0		4
34	ADFB4BD...	[3/3]dh ka...	0		4
35	9745D42...	Papa hin l...	0		4
36	0F8C1F4...	call me	0		4
37	F487FE1...	[1/2]talked...	0		4
38	B7144A3...	[2/2]ght & ...	0		4
39	8FFECE4...	send che...	0		4
45	A68FEB3...	DIN NO.O...	0		12
46	DE0CEC...	moscrape...	0		4
47	3CF348B...	98254007...	0		4
51	8C18470...	Jitendra G...	0		14
52	DD1AA39...	Shanti mil...	0		4
53	D9620F3...	Maal utha...	0		4
54	75422DD...	Coming V...	0		15
55	4252933	Yes sir	0		15

Fig. 04: Portion of the table MESSAGE

The details of the various fields in the Message Table and their details are given in ***Table 01***.

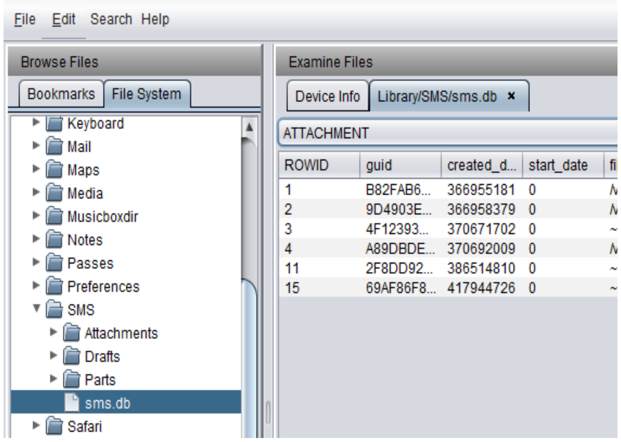
Table 01: Fields in Message Table

Field Name	Type	Value / description
ROWID	Integer Primary Key Autoincrement	Primary key
address	Text	Null or a name or a phone number (with or without spaces) of the other person (sent to or received from)
date	Integer	Message date
text	Text	Message content
flags	Integer	Unknown, possible values: 0, 2, 3, 5, 35, 16387. Probably a bit-set. The value 35 was set in a SMS that couldn't get sent out and is still marked with a red exclamation mark letting you send it again.
replace	Integer	Unknown, possible values: 0, 1, 2
svc_center	Text	Null
group_id	Integer	0 or foreign key to msg_group.rowid
association_id	Integer	Often 0, but sometimes a copy of the date field
height	Integer	Always 0
UIFlags	Integer	Unknown, possible values: 0, 4, 5, 6, 7
version	Integer	Always 0
subject	Text	The subject of an imessage/mms-message, or null if it's a sms or if subject is not used on a imessage/mms-message
country	Text	Null or an iso country code (eg: 'in' for india)
headers	Blob	Always NULL
recipients	Blob	Normally null, one entry had an xml value in it
read	Integer	0 or 1 (assume 0 = unread and 1 is read though madrid messages are always 0 so it probably doesn't apply to them)
madrid_attributedBody	Blob	Blob, content unknown. The only strings in it are "jfif" and "exif", so this is probably meta-data.
madrid_handle	Text	Null if not an imessage or a phone number of the other person (sender or receiver)
madrid_version	Integer	Null if pre-ios 5.0 or 0
madrid_guid	Text	Guid (unique to the message) or null if not an imessage
madrid_type	Integer	0 or null if pre-ios 5.0
madrid_roomname	Text	Null

madrid_service	Text	'Madrid' or null if not an imessage
madrid_account	Text	Null if not an imessage or 'p:' & own phone number or 'e:' & email registered for imessage
madrid_flags	Integer	Message type. Known values: null (if not an imessage), 12289 (received), 32773 (send error), 36869 (sent), 45061 (sent), 77825 (received message containing parsed data eg: phone, email, website), 102405 (sent message containing parsed data eg: phone, email, website)
madrid_attachmentInfo	Blob	Null or blob. The blob contains these strings: streamtyped, nsmutablearray, nsarray, nsobject, nsmutablestring, nsstring, and a guid. Format unknown.
madrid_url	Text	Always an empty string
madrid_error	Integer	Empty string if message is pre-ios 5.0 or 0 if after
is_madrid	Integer	Specifies if it's an imessage or not, value 0 or 1 (0=sms/mms, 1=imessage)
madrid_date_read	Integer	Null if message is pre-ios 5.0, 0 if sms or a sent imessage, integer value representing the date read
madrid_date_delivered	Integer	Null if message is pre-ios 5.0, 0 if sms or a received imessage, integer value representing the date sent
madrid_account_guid	Text	Guid of account used (multiple entries may be found representing either the phone or email registered with imessage) or empty if not an imessage

F. *Attachment Table:*

This table contains the details of the message attachment, if the message is an MMS (Multi Media Service) and provides details of type of content i.e., image, audio, video, where the images are backed up to, MIME type, file size, etc., and whether is an incoming or outgoing etc., The portion of the screen shot of the table is given at ***Fig. 05***.



ROWID	guid	created_d...	start_date	fi
1	B82FAB6...	366955181	0	M
2	9D4903E...	366958379	0	M
3	4F12393...	370671702	0	~
4	A89DBDE...	370692009	0	M
11	2F8DD92...	386514810	0	~
15	69AF86F8...	417944726	0	~

Fig. 05: Portion of the table ATTACHMENT

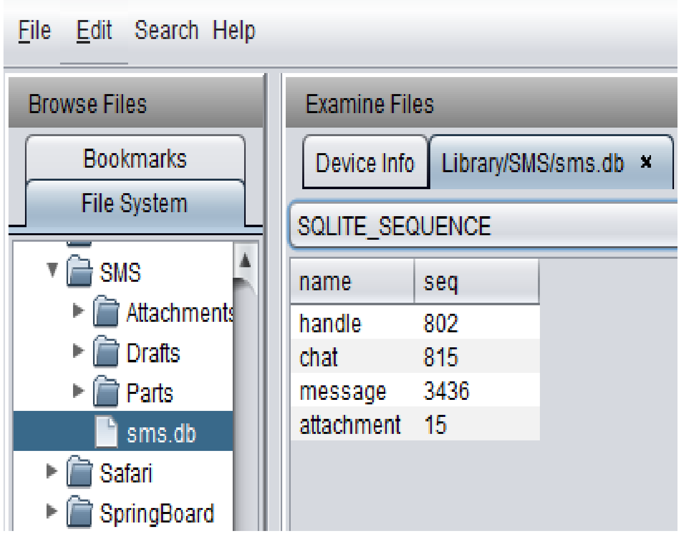
The details of the various fields in the Attachment Table and their details are given in ***Table 02***.

Table 02: Fields in Attachment Table

Field Name	Type	Value / Description
RowId	Integer primary key autoincrement	Primary key
Attachment_guid	Text	Guid - this matches the subfolder name in the folder attachments
Created_date	Integer	Unsigned integer value with the creation date
Start_date	Integer	0
Filename	Text	Complete filename (with path)
Uti_type	Text	'Public.jpeg' or 'public.vcard'
Mime_type	Text	'Image/jpeg' or 'text/vcard'
Transfer_state	Integer	5
Is_incoming	Integer	0
Message_id	Integer	-1

G. *SQLITE_SEQUENCE Table:*

This table contains the details of increment in various fields and gets autoincremented as and when any new entry is added to the message table. The screen shot of the table is shown in ***Fig. 06***.

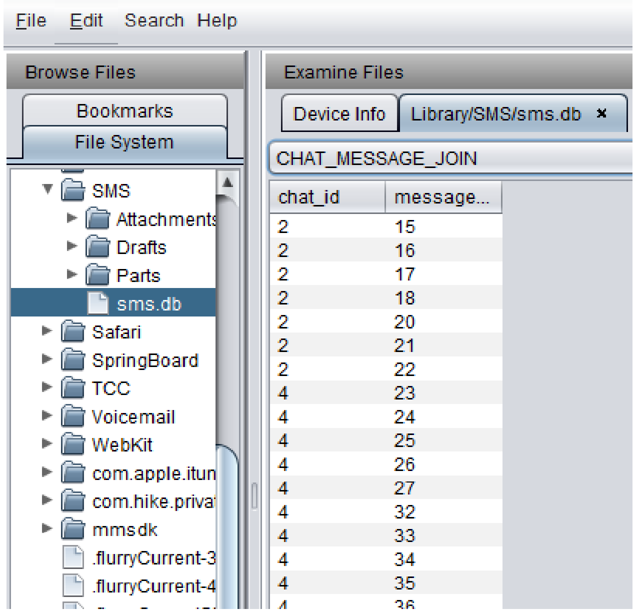


name	seq
handle	802
chat	815
message	3436
attachment	15

Fig. 06: Portion of the table SQLITE_SEQUENCE

H. *CHAT_MESSAGE_JOIN Table:*

This table links the ROW IDs of chats and the messages inside them. The portion of the screen shot of the table is given at ***Fig. 07***.



CHAT_MESSAGE_JOIN

chat_id	message...
2	15
2	16
2	17
2	18
2	20
2	21
2	22
4	23
4	24
4	25
4	26
4	27
4	32
4	33
4	34
4	35
4	36

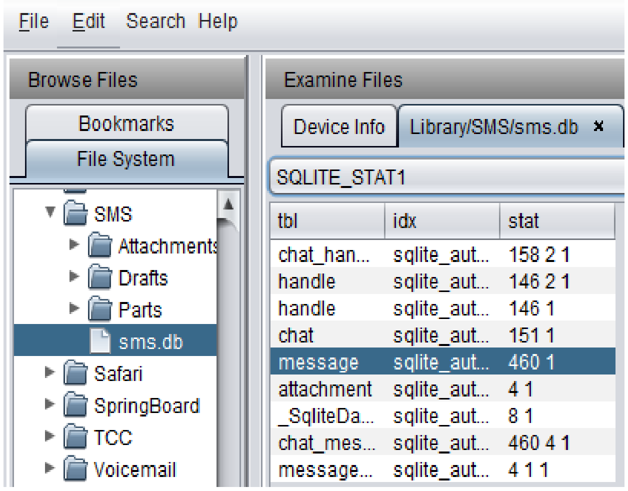
Fig. 07: Portion of the table CHAT_MESSAGE_JOIN

The description of the fields in the above table is as below:

- chat_id the id of the chat
- message_id the id of the message that is part of the chat

I. SQLITE_STAT1 Table:

This table is used to store statistical information about the tables and indexes analysed. The portion of the screen shot of the table is given at ***Fig. 08***.



SQLITE_STAT1

tbl	idx	stat
chat_han...	sqlite_aut...	158 2 1
handle	sqlite_aut...	146 2 1
handle	sqlite_aut...	146 1
chat	sqlite_aut...	151 1
message	sqlite_aut...	460 1
attachment	sqlite_aut...	4 1
_SqliteDa...	sqlite_aut...	8 1
chat_mes...	sqlite_aut...	460 4 1
message...	sqlite_aut...	4 1 1

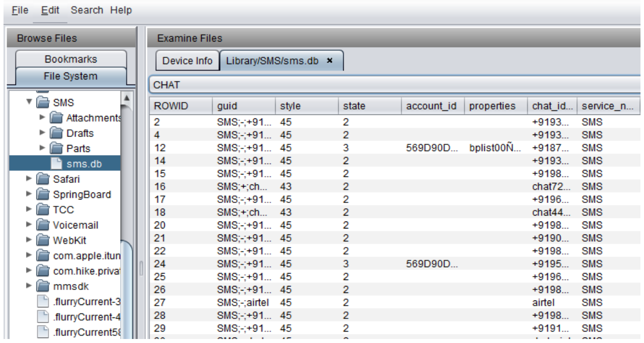
Fig. 08: Portion of the table SQLITE_STAT1

The description of the fields in the above table is as below:

Column Name	Description
tbl	The table name that was analysed.
idx	The name of the index that was analysed.
stat	Information about the table and indexes analysed that will be later used by the query optimizer.

J. CHAT Table:

This table is used to store the details of message conversations. The portion of the screen shot of the table is given at ***Fig. 09***.



CHAT

ROWID	guid	style	state	account_id	properties	chat_id...	service_n...
2	SMS;-;+91...	45	2			+9193...	SMS
4	SMS;-;+91...	45	2			+9193...	SMS
12	SMS;-;+91...	45	3	569D90D...	bplist00Ñ...	+9187...	SMS
14	SMS;-;+91...	45	2			+9193...	SMS
15	SMS;-;+91...	45	2			+9198...	SMS
16	SMS;+;ch...	43	2			chat72...	SMS
17	SMS;-;+91...	45	2			+9196...	SMS
18	SMS;+;ch...	43	2			chat44...	SMS
20	SMS;-;+91...	45	2			+9198...	SMS
21	SMS;-;+91...	45	2			+9190...	SMS
22	SMS;-;+91...	45	2			+9198...	SMS
24	SMS;-;+91...	45	3	569D90D...		+9195...	SMS
25	SMS;-;+91...	45	2			+9196...	SMS
26	SMS;-;+91...	45	2			+9198...	SMS
27	SMS;-;airtel	45	2			airtel	SMS
28	SMS;-;+91...	45	2			+9198...	SMS
29	SMS;-;+91...	45	2			+9191...	SMS

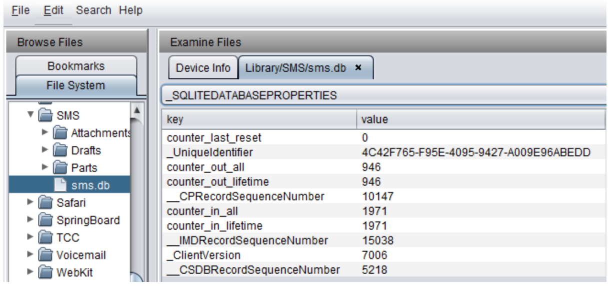
Fig. 09: Portion of the table CHAT

The description of the various fields in this table are as below:

Field Name	Type	Value / Description
Rowid	Integer Primary Key Autoincrement	Primary key
Style	Integer	Only known value is 45
State	Integer	Only known value is 3
Account_Id	Text	Guid Of the Imessage Account You Used
Properties	Blob	Null for the second row or a bplist (see below)
Chat_Identifier	Text	Phone number of the other person in international format, no spaces
Service_Name	Text	'Madrid'
Guid	Text	Same as chat_identifier, but with a '-' in front of it
Room_Name	Text	Null
Account_Login	Text	'P:' & own phone number or 'e:' & email registered for imessage
Participants	Blob	Bplist

K. _SQLITEDATABASEPROPERTIES Table:

A list of nine properties of the database such as the client version and GUID. This table is used to store the configuration details. The portion of the screen shot of the table is given at ***Fig. 10.***



_SQLITEDATABASEPROPERTIES

key	value
counter_last_reset	0
_UniqueIdentifier	4C42F765-F95E-4095-9427-A009E96ABEDD
counter_out_all	946
counter_out_lifetime	946
__CPRecordSequenceNumber	10147
counter_in_all	1971
counter_in_lifetime	1971
__IMDRecordSequenceNumber	15038
_ClientVersion	7006
__CSDBRecordSequenceNumber	5218

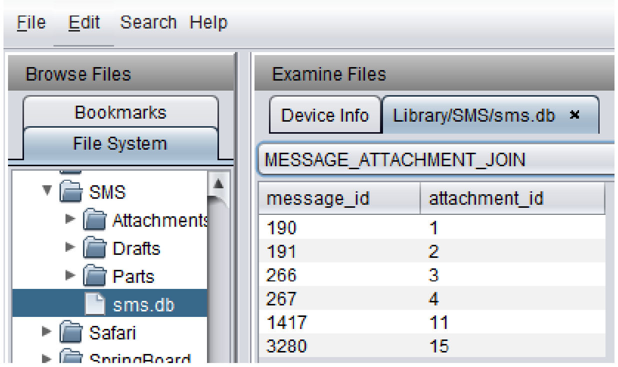
Fig. 10: Portion of the table_SQLITEDATABASEPROPERTIES

The description of some of the fields in the above table are as follows:

- counter_out_all: 946 - counts the number of outgoing messages (since last counter reset)
- counter_out_lifetime: 946 - counts the number of outgoing messages (since forever, isn't affected by a counter reset)
- counter_in_all: 1971 - counts the number of incoming messages (since last counter reset)
- counter_in_lifetime: 1971 - counts the number of incoming messages (since forever, isn't affected by a counter reset)

L. MESSAGE_ATTACHMENT_JOIN Table:

This table contains two columns, the ***message_id*** column and the ***attachment_id*** column, which correspond to the ***ROWID*** column of the message table. This will join the messages to their attachments if any via ***ROWID*** relationship. The screenshot of the table is given at ***Fig. 11.***



message_id	attachment_id
190	1
191	2
266	3
267	4
1417	11
3280	15

Fig. 11: Portion of the table MESSAGE_ATTACHMENT_JOIN

All the above tables in the **sms.db** file are related as shown in the ***Fig. 12.***

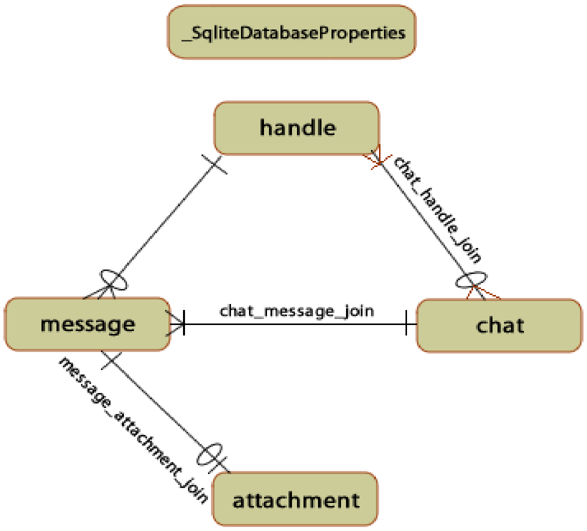


Fig. 12: Relational diagram of tables in sms.db

IV. TEST METHODOLOGY

The above iTunes backup folder has been analysed with the help of the Mobile Phone Forensic Tool Oxygen Forensic Suit 2014 Version 6.2.1.103. Also, iPhone Analyser Version 2.1.70 downloaded from crypticbit.com, DB Browser for SQLite and SQL Parse GUI have been used in the test analysis process.

For testing the integrity of the SMS messages with in the ***sms.db*** file retrieved we have focused our analysis on the content of the message table.

The control that could be providing the evidence to identify a manipulation of SMS database is the message sequencing. When a new message is sent or received, it is inserted at the end of the message table as a new record. In doing so the message receives a new ROW ID value in sequential order. In doing so the date and time values as well as the ROW ID values increment as one moves through the messages table from the earliest to the newest messages.

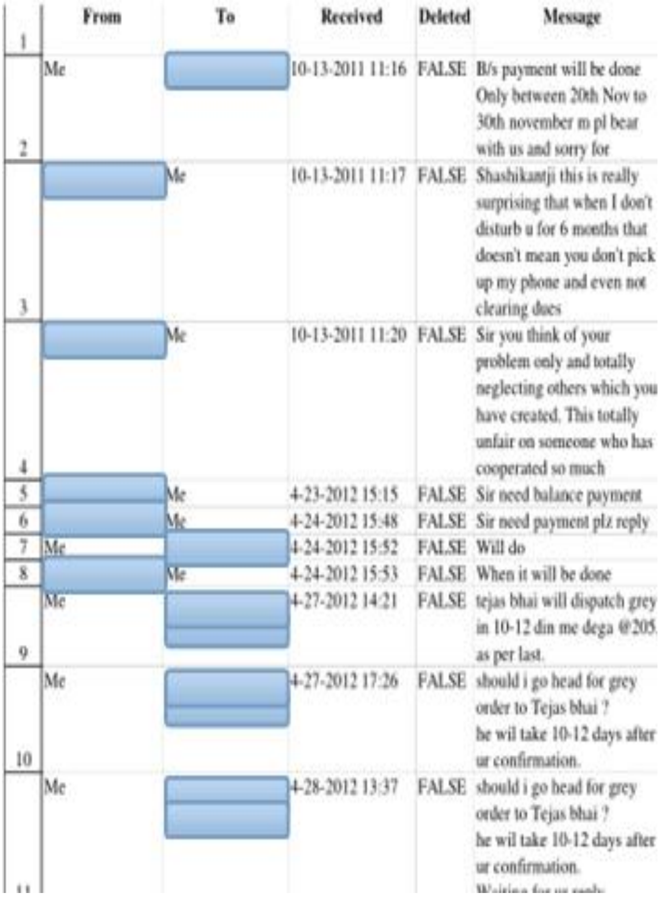
In the event a message or messages are inserted manually by using third party utilities, it is possible that the chronological sequence of the message dates and time values with that of the ROW IDs will be out of sequence.

The other controls that could provide evidence of manipulation are verifying the trigger for message count. If the trigger is active and updates the value in the newest_message field to reflect the message id for the inserted message. It should have the highest message id value for each group listed in the msg_group table. If there is any mismatch is an indication that a message or messages are being inserted. But in the current scenario the msg_group table is not present indicating no existence of group conversation in the current case.

As such in the current case we have relied on analysing the SMS messages in the message table by arranging them in the sequence of ROW ID and study if any anomaly in the date and time, which indicate the tampering in the database.

V. RESULTS

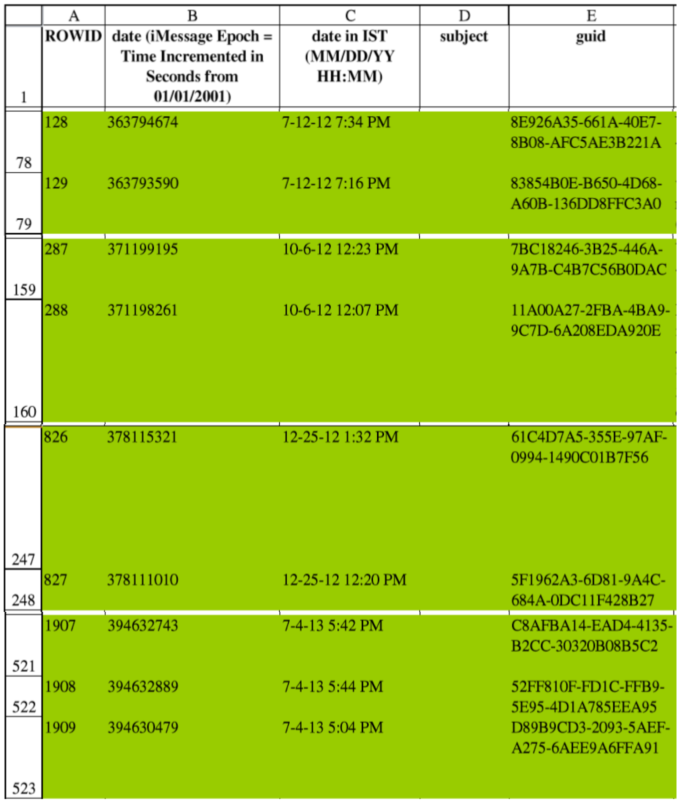
Fig. 13 shows the portion of the SMS messages retrieved by using the Mobile Phone Forensic Tool Oxygen Forensic Suit 2014 Version 6.2.1.103. From the figure, it is evident that the parsed SMS messages does not indicate specifically any information related to their authenticity.



1	From	To	Received	Deleted	Message
2	Me	[redacted]	10-13-2011 11:16	FALSE	B/s payment will be done Only between 20th Nov to 30th november m pl bear with us and sorry for
3	[redacted]	Me	10-13-2011 11:17	FALSE	Shashikantji this is really surprising that when I don't disturb u for 6 months that doesn't mean you don't pick up my phone and even not clearing dues
4	[redacted]	Me	10-13-2011 11:20	FALSE	Sir you think of your problem only and totally neglecting others which you have created. This totally unfair on someone who has cooperated so much
5	[redacted]	Me	4-23-2012 15:15	FALSE	Sir need balance payment
6	[redacted]	Me	4-24-2012 15:48	FALSE	Sir need payment plz reply
7	Me	[redacted]	4-24-2012 15:52	FALSE	Will do
8	[redacted]	Me	4-24-2012 15:53	FALSE	When it will be done
9	Me	[redacted]	4-27-2012 14:21	FALSE	tejas bhai will dispatch grey in 10-12 din me dega @205. as per last.
10	Me	[redacted]	4-27-2012 17:26	FALSE	should i go head for grey order to Tejas bhai ? he wil take 10-12 days after ur confirmation.
11	Me	[redacted]	4-28-2012 13:37	FALSE	should i go head for grey order to Tejas bhai ? he wil take 10-12 days after ur confirmation. Waiting for ur reply

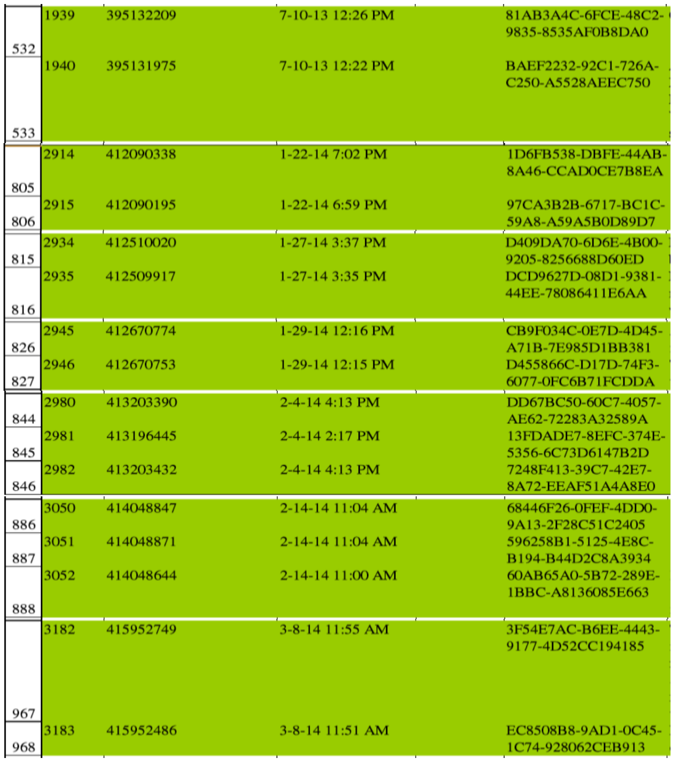
Fig. 13: Portion of SMS Messages in the retrieved messages table of sms.db file

As such, the message table of the ***sms.db*** has been exported as ***.csv*** file. By keeping the values in the ***ROW ID*** column arranged sequentially in the increasing order, the date ***(Prior to iOS 5, all text messages in sms.db were stored with a Unix Epoch time stamp, i.e., number of seconds since January 1, 1970. Since the release of iOS 5, standard SMS messages are still stored with a Unix Epoch time stamp; however, iMessages are stored with a different timestamp: Mac Epoch, or the number of seconds since January 1, 2001 and from iOS 6 onwards irrespective of whether the service is SMS or iMessage the date values are stored with Mac Epoch time stamp, or the number of seconds since January 1, 2001)*** **converted to IST (Indian Standard Time**), by applying conversion factor of seconds since January 1, 2001, field has been studied for any oddity. It has been observed incongruities at various places, that the date of receipt/sending is not in the sequentially ascending order with ROW ID as shown in the ***Fig. 14*** and ***Fig. 15***.



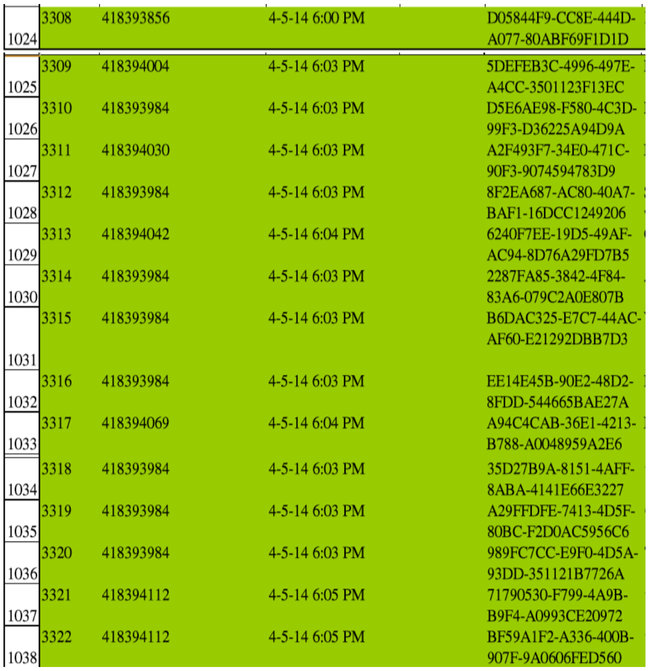
	A	B	C	D	E
1	ROWID	date (iMessage Epoch = Time Incremented in Seconds from 01/01/2001)	date in IST (MM/DD/YY HH:MM)	subject	guid
78	128	363794674	7-12-12 7:34 PM		8E926A35-661A-40E7-8B08-AFC5AE3B221A
79	129	363793590	7-12-12 7:16 PM		83854B0E-B650-4D68-A60B-136DD8FFC3A0
159	287	371199195	10-6-12 12:23 PM		7BC18246-3B25-446A-9A7B-C4B7C56B0DAC
160	288	371198261	10-6-12 12:07 PM		11A00A27-2FBA-4BA9-9C7D-6A208EDA920E
247	826	378115321	12-25-12 1:32 PM		61C4D7A5-355E-97AF-0994-1490C01B7F56
248	827	378111010	12-25-12 12:20 PM		5F1962A3-6D81-9A4C-684A-0DC11F428B27
521	1907	394632743	7-4-13 5:42 PM		C8AFBA14-EAD4-4135-B2CC-30320B08B5C2
522	1908	394632889	7-4-13 5:44 PM		52FF810F-FD1C-FFB9-5E95-4D1A785EEA95
523	1909	394630479	7-4-13 5:04 PM		D89B9CD3-2093-5AEF-A275-6AEE9A6FFA91

Fig. 14: Screen shot showing the oddity in the SMS messages



532	1939	395132209	7-10-13 12:26 PM		81AB3A4C-6FCE-48C2-9835-8535AF0B8DA0
533	1940	395131975	7-10-13 12:22 PM		BAEF2232-92C1-726A-C250-A5528AEEC750
805	2914	412090338	1-22-14 7:02 PM		1D6FB538-DBFE-44AB-8A46-CCAD0CE7B8EA
806	2915	412090195	1-22-14 6:59 PM		97CA3B2B-6717-BC1C-59A8-A59A5B0D89D7
815	2934	412510020	1-27-14 3:37 PM		D409DA70-6D6E-4B00-9205-8256688D60ED
816	2935	412509917	1-27-14 3:35 PM		DCD9627D-08D1-9381-44EE-78086411E6AA
826	2945	412670774	1-29-14 12:16 PM		CB9F034C-0E7D-4D45-A71B-7E985D1BB381
827	2946	412670753	1-29-14 12:15 PM		D455866C-D17D-74F3-6077-0FC6B71FCDDA
844	2980	413203390	2-4-14 4:13 PM		DD67BC50-60C7-4057-AE62-72283A32589A
845	2981	413196445	2-4-14 2:17 PM		13FDADE7-8EFC-374E-5356-6C73D6147B2D
846	2982	413203432	2-4-14 4:13 PM		7248F413-39C7-42E7-8A72-EEAF51A4A8E0
886	3050	414048847	2-14-14 11:04 AM		68446F26-0FEF-4DD0-9A13-2F28C51C2405
887	3051	414048871	2-14-14 11:04 AM		596258B1-5125-4E8C-B194-B44D2C8A3934
888	3052	414048644	2-14-14 11:00 AM		60AB65A0-5B72-289E-1BBC-A8136085E663
967	3182	415952749	3-8-14 11:55 AM		3F54E7AC-B6EE-4443-9177-4D52CC194185
968	3183	415952486	3-8-14 11:51 AM		EC8508B8-9AD1-0C45-1C74-928062CEB913

Fig. 15: Screen shot showing the oddity in the SMS messages



1024	3308	418393856	4-5-14 6:00 PM		D05844F9-CC8E-444D-A077-80ABF69F1D1D
1025	3309	418394004	4-5-14 6:03 PM		5DEFEB3C-4996-497E-A4CC-3501123F13EC
1026	3310	418393984	4-5-14 6:03 PM		D5E6AE98-F580-4C3D-99F3-D36225A94D9A
1027	3311	418394030	4-5-14 6:03 PM		A2F493F7-34E0-471C-90F3-9074594783D9
1028	3312	418393984	4-5-14 6:03 PM		8F2EA687-AC80-40A7-BAF1-16DCC1249206
1029	3313	418394042	4-5-14 6:04 PM		6240F7EE-19D5-49AF-AC94-8D76A29FD7B5
1030	3314	418393984	4-5-14 6:03 PM		2287FA85-3842-4F84-83A6-079C2A0E807B
1031	3315	418393984	4-5-14 6:03 PM		B6DAC325-E7C7-44AC-AF60-E21292DBB7D3
1032	3316	418393984	4-5-14 6:03 PM		EE14E45B-90E2-48D2-8FDD-544665BAE27A
1033	3317	418394069	4-5-14 6:04 PM		A94C4CAB-36E1-4213-B788-A0048959A2E6
1034	3318	418393984	4-5-14 6:03 PM		35D27B9A-8151-4AFF-8ABA-4141E66E3227
1035	3319	418393984	4-5-14 6:03 PM		A29FFDFE-7413-4D5F-80BC-F2D0AC5956C6
1036	3320	418393984	4-5-14 6:03 PM		989FC7CC-E9F0-4D5A-93DD-351121B7726A
1037	3321	418394112	4-5-14 6:05 PM		71790530-F799-4A9B-B9F4-A0993CE20972
1038	3322	418394112	4-5-14 6:05 PM		BF59A1F2-A336-400B-907F-9A0606FED560

It is to be noted that for manipulating the SMS messages in an iPhone directly in the phone itself requires a process called ***jailbreaking***, which facilitates access to various internal memory resources of the iPhone with out any restrictions. Alternatively, it can be achieved by way of backup of iPhone data through iTunes application software and performing required manipulations to the SMS messages and restoring the modified backup to the iPhone. Further, while manipulated SMS database is reinserted into the iTunes backup, various control vales with in the iTunes backup needs to be adjusted to reflect that the modified SMS database has been inserted. Failure to do so will restrict the restoration of the backup to iPhone. The adjustment of the control values with in the iTunes backup for the manipulated SMS database is a tedious and tricky process and manually doing the same is a jargon. But the same can be tackled in a lucid manner with the help of third-party applications. In the present case it has been observed that the iPhone is not ***jailbroken***. As such it can be concluded that for manipulating the SMS messages the alternative method discussed above might has been adopted.

VI. CONCLUSION

Aforesaid manual interpretation procedure will greatly help the forensic examiners in understanding the **sms.db** file to find out whether any messages have been manually inserted or otherwise in iPhone devices by using third party utilities or by jailbreaking the iPhone devices in verifying and proving the veracity of the SMS messages, where most of the third-party commercial forensic toolkits will not be able to highlight these anomalies.

VII. Acknowledgments

The authors are very thankful to *Shri. Surya Prasad Yanamandra*, Head of Computer Forensic Unit, Central Forensic Science Laboratory, Hyderabad for his valuable guidance and also to *Shri. Venkata Ramana Potturi* for his support in this manuscript drafting, scrutiny and grammatical corrections.

REFERENCES

[1] Practical Investigations of Digital Forensics Tools for Mobile Devices by Maynard Yates II, M.S. Florida Agricultural and Mechanical University Department of Computer and Information Sciences Technical Building A, Room 211 Tallahassee, FL 32307-5100 Maynard1.yates@famu.edu retrieved December 2018.

[2] Barrios, Rita M. and Lehrfeld, Michael R., "Ios Mobile Device Forensics: Initial Analysis" (2011). Annual ADFSL Conference on Digital Forensics, Security and Law. 4. http://commons.erau.edu/adfsl/2011/friday/4 December 2018

[3] Hoene, Thomas & Creutzburg, Reiner. (2011). iPhone forensics: a practical overview with certain commercial software. 10.1117/12.884589. December 2018.

[4] Cheema, Ahmad & Iqbal, Waseem & Ali, Waqas. (2014). An Open Source Toolkit for iOS Filesystem Forensics. 10.1007/978-3-662-44952-3_15. December 2018.

[5] Forensic Analysis on iOS Devices Author: Tim Proffitt, tim@timproffitt.com https://www.sans.org/reading-room/whitepapers/forensics/forensic-analysis-ios-devices-34092 December 2018.

[6] iPhone and iOS Forensics: Investigation, Analysis and Mobile Security for Apple iPhone, iPad and iOS Devices by Andrew HoogKatie Strzempka.

[7] Bader, M., & Baggili, I. (2010). iPhone 3GS forensics: logical analysis using apple itunes backup utility. Small scale digital device forensics journal, 4(1), 1-15 January 2019.

[8] SQLite Wikipedia article. URL https://en.wikipedia.org/wiki/SQLite January 2019.

[9] How to back up your iPhone, iPad, and iPod touch URL https://support.apple.com/en-in/HT203977 January 2019.

[10] Parsing the iPhone SMS Database article. URL https://linuxsleuthing.blogspot.com/2011/02/parsing-iphone-sms-database.html January 2019.

[11] Wikipedia article. URL https://www.theiphonewiki.com/wiki/Messages#msg_group January 2019.

[12] https://smarterforensics.com/2014/09/sqlite-parser-theres-a-new-gui/ January 2019.

[13] https://smarterforensics.com/2017/09/time-is-not-on-our-side-when-it-comes-to-messages-in-ios-11/ visited on 19/01/2019.

[14] https://linuxsleuthing.blogspot.com/2012/10/whos-texting-ios6-smsdb.html visited on 19/01/2019.

[15] https://www.codeproject.com/Articles/833535/Accessing-Backed-Up-iPhone-SMS-Messages visited on 19/01/2019.

[16] https://blog.jverkamp.com/2015/01/27/ios-backups-in-racket-messages/ visited on 19/01/2019.

[17] http://mrdreigon.com/ios6-sms-database-investigation/ visited on 19/01/2019.

[18] https://www.vivekmchawla.com/2013/04/erd-crows-foot-relationship-symbols-quick-reference.html/ visited on 19/01/2019.

[19] Forensic investigations of Apple's iPhone by Mats Engman, 2013. https://www.diva-

portal.org/smash/get/diva2:651693/fulltext01.pdf retreived January 2019.

[20] Smartphone Forensics Analysis: A Case Study by Mubarak Al-Hadadi and Ali AlShidhani International Journal of Computer and Electrical Engineering, Vol. 5, No. 6, December 2013. January 2019.

Authors Profile

Mr. Eswara Sai Prasad Chunduru pursed Master of Science in Physical Chemistry from Andhra University, Vishakhapatanam, Andhra Pradesh, India in 1996 and Master of Science in Information Tehnology from Nagarjuna Univerity, Guntur, Andhra Pradesh, India in 2017. He has completed his Postgraduate Diploma in Chemical Analysis and Quality Managewment from University of Hyderabad. He is a Information Security Management System Lead Auditor from BSI Management Systems, AccessData Certified Computer Forensic Examiner, Certified ProDiscover Computer Forensic Examiner, Certified Cyber Forenisc Examiner from C-DAC, Trivandrum, Certified Computer Hacking Forensic Investigator Version 8 from EC-Council, Certified e-Suraksha Proffesional from C-DAC, Hyderabad. He is currently working as Scientist B in Computer Forensic Unit of Centra Forensic Science Laboraotry, Directorate of Forensic Science Services, Government of India, Hyderabad. During his tenure he has analysed more than 1500 cybercrime cases referred by various investigation agencies and provided reports. His main research work focuses on Windows Forensics, Mobile Phone Forensics and Registry Foreniscs. He has 20 years of rich experience in dealing with various types of cybercrime related issues. He has 03 years of teachning experience.

Mr. Krishna Mangarai pursed Master of Science in Chemistry from Osmania University in 1987. He has his M. Tech (IT) from Ahnadabad Agricultural University. He is a GCIH - GIAC Certified Incident Handler from SANS, USA. He is a Certified ProDiscover Computer Forensic Examiner, Certified Computer Hacking Forensic Investigator Version 8 from EC-Council, Certified e-Suraksha Proffesional from C-DAC, Hyderabad. He is currently working as Assistant Director/Scientist C in Computer Forensic Unit of Centra Forensic Science Laboraotry, Directorate of Forensic Science Services, Government of India, Hyderabad. During his tenure he has analysed more than 1750 cybercrime cases referred by various investigation agencies and provided reports. His main research work focuses on Windows Forensics, Mobile Phone Forensics and Registry Foreniscs. He has more than 20 years of rich experience in dealing with various types of cybercrime related issues.

Mr. Subrahmani Babu pursed Master of Information Technology M.Sc[IT] at Alagappa University, Karaikudi, Tamilnadu, India in 2002 and Master of Philosophy in Computer Science M.Phil [Computer Science] at Bharathidasan University, Trichy, Tamilnadu, India in the year 2004. He is a GCFA - GIAC Certified Forensic Analyst from SANS, USA. He is currently working with Deloitte Shared Services as Manager in cyber security and digital forensics investigation, Pentest, CASB, DL, eDiscovery and Vulnerability management systems since 2018. Prior to this he worked with Mphasis, Bangalore as Global Head in Digital Forensics Investigation and E-Discovery as part of Chief Risk Office and as Digital Forensic Scientist, Scientist C in Cyber Forensics Division, Indian Computer Emergency Response Team (CERT-In) from August 2008 to March 2016. During his tenure at these organisations he has analysed various cybercrime cases of national and international importance. His main research work focuses on Cryptography Algorithms, Network Security, Cloud Security and Privacy, Big Data Analytics, Data Mining, IoT. He has 14 years of rich experience in dealing with various types of cybercrime related issues.

Mr. Manohar Bhushan Pathak pursed his Master degree in Chemistry from MJPRU, Barelly, Uttar Pradesh, India. He got training in the field of Cyber Forensics from NICFS, C-DAC. He is currently working as Assistant Central Intelligence Officer Grade I with Central Forenasic Science Laboratory, Directorate of Forensic Science Services, Hyderabad science 2010. During his tenure he has involved in conducting forensic analysis of various impotant cases. He has 09 years of rich experience in dealing with various types of cybercrime related issues.