MSAB BLOG

Green meets Blue: A brief RCS forensic excursion

- WHAT RCS is? (Rich Communication Services)
- What can it do?
- WHO uses RCS and, if so, HOW?
- And, above all, where can I find forensic evidence of this?

MARKUS HESS
MSAB DIGITAL FORENSIC SPECIALIST

MSAB
Trusted Partner in Digital Forensics

01 — THE BASICS

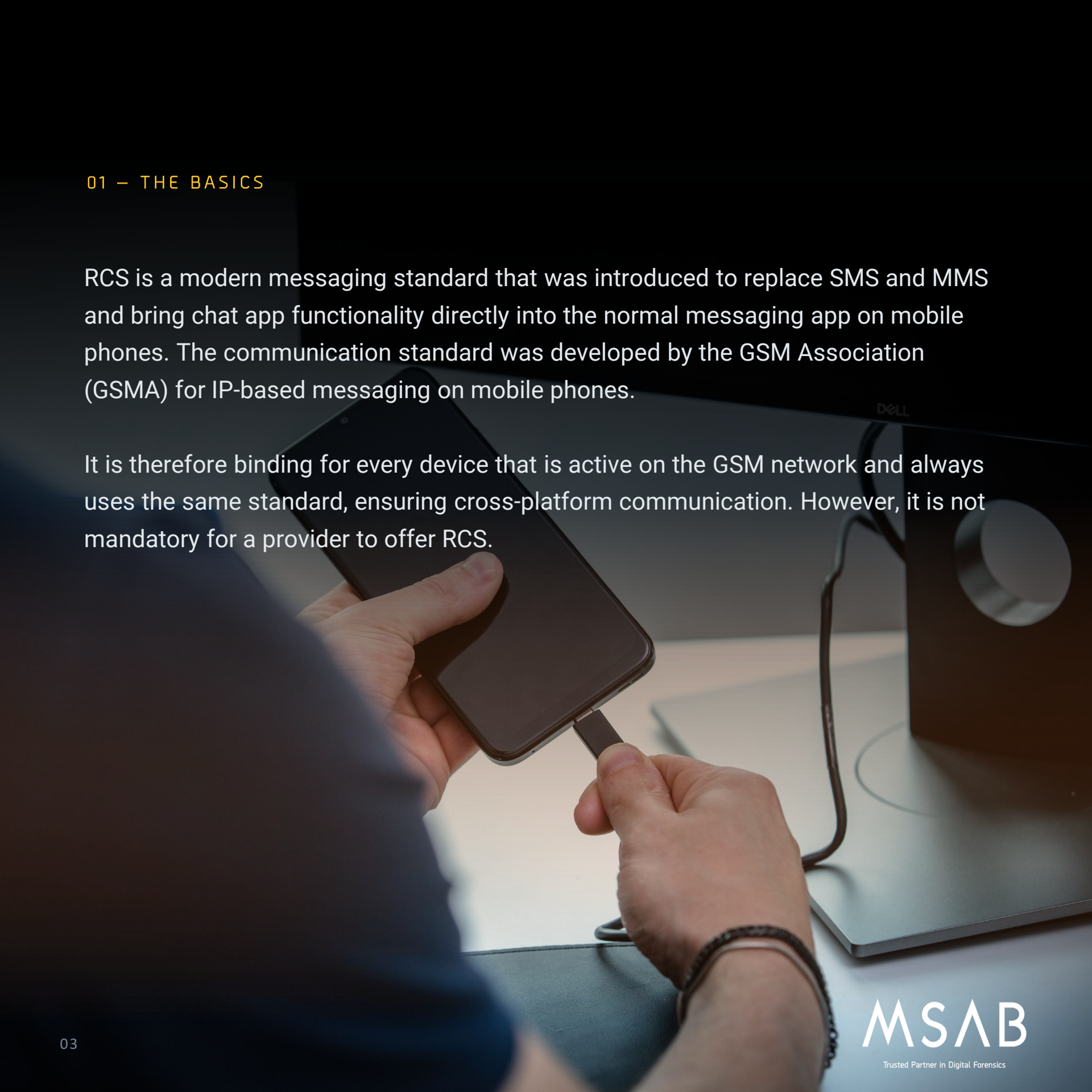
What is RCS anyway?

All delivered in what I hope is a concise and accessible way, while still bringing in the right level of expertise where it matters. Think of it as too detailed for a TikTok, but not quite long enough for a podcast.

01 — THE BASICS

RCS is a modern messaging standard that was introduced to replace SMS and MMS and bring chat app functionality directly into the normal messaging app on mobile phones. The communication standard was developed by the GSM Association (GSMA) for IP-based messaging on mobile phones.

It is therefore binding for every device that is active on the GSM network and always uses the same standard, ensuring cross-platform communication. However, it is not mandatory for a provider to offer RCS.



Important functions

- Sending high-resolution photos, videos, files and, in some cases, GIFs directly in the messaging app.

- Typing indicator (“someone is typing…”), read/receipt confirmations, longer text lengths and improved group chats with renaming and leaving groups.

- Use of mobile data or Wi-Fi instead of the traditional SMS signaling network.

Security and encryption

- RCS is fundamentally IP-based; many implementations encrypt messages at least during transmission.

- Provided that both participants are using compatible RCS implementations that support end-to-end encryption, 1:1 chats can be end-to-end encrypted.

Understanding the differences: SMS, iMessage and RCS

Before we address the forensic challenges, it is important to distinguish differences between these messaging protocols:

SMS — This traditional text messaging standard is limited to 160-character text messages, offers no encryption and only supports basic metadata.

iMessage — Apple's own messaging service uses end-to-end encryption and synchronizes messages across devices. Messages are stored in iCloud backups.

RCS — Unlike SMS, RCS works online and supports modern communication features like iMessage and other chat applications.

02 — ADOPTION

What if my conversation partner does not support RCS, or who supports RCS anyway?

Two factors are important here: the mobile operating system and the network operator.

While the former is relatively easy and straightforward to answer, all modern Android smartphones (from Android 6 with Google Messages onwards) and newer iPhones with iOS 18 or higher currently support RCS.

The second factor, network operator support, is somewhat more complicated. Although coverage is improving, it has not yet quite caught up with SMS. Many of the major providers worldwide, such as Telekom, Vodafone, Orange, AT&T, T-Mobile, Verizon and others, support RCS as standard at no extra charge.

Whether Android or Apple iOS, RCS messages are always stored and sent in the normal standard app (Messages, for example). No extra RCS app is required.

02 — ADOPTION

You can tell whether you are communicating with someone using the new RCS protocol by the fact that this is usually displayed right at the top of the chat view as can be seen in Figure 1.0. If one of the participants switches to RCS during a conversation (e.g. due to a mobile phone change/update or network operator updates/changes), this message may also appear in the middle of two messages.

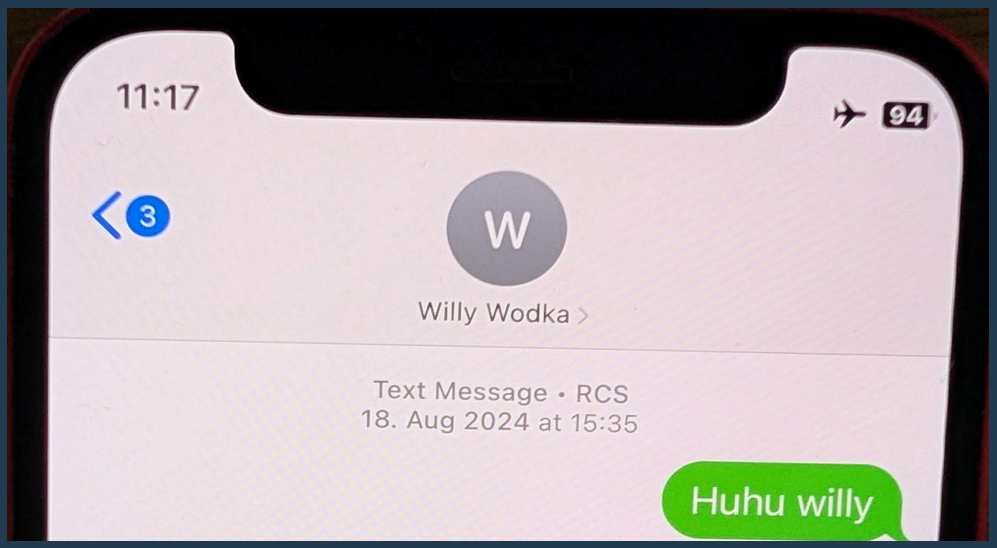


Figure 1.0 (Screenshot of iOS device using RCS messaging)

However, communication between two Apple iOS devices will continue to be handled by iMessage by default. You can change this and other RCS settings in the “Messages” settings. Since we already know that RCS is sent with the standard messaging apps, it makes sense to look here. A good forensic program should therefore also be able to decode RCS messages. But where do the files originate? How do I know which protocol is active in a conversation? To find out, let's take a closer look at the SQL data.

RCS on Android

Most forensic experts know that they need to search the mmssms.db file on almost every device for SMS/MMS data which can be found at the following path:

```
/data/data/com.android.
providers.telephony/
databases/mmssms.db
```

mmssms.db

Mmssms.db is a SQLite database and is supported by most forensic tools or can be read with any SQLite viewer. There are several interesting tables, but the most important information is mainly found in the message content, the message timestamp and the sender/recipient.

_id	thread_id	address	date	date_sent	body	creator
46	30	Apple	1727091577619	1727091576000	Dein Apple Accoun...	com.google.android.apps.messaging
63	16	Amazon	1728620376710	1728620375000	Amazon: Dein Code...	com.google.android.apps.messaging
87	40	+4915234687812	1729918199593	1729898619000	Hallo Papa, das ist ...	com.google.android.apps.messaging

Figure 2.0 (A snippet from mmssms.db)

icing_mmssms.db

```
/data/data/com.google.
android.gms/databases/
icing_mmssms.db
```

This is an additional database created by Google Play Services. It often contains the same content as mmssms.db, as it is used for indexing this service, among other things, but you may also find entries here that have already been deleted in mmssms.db. Of course, RCS content can also be found here, and the structure of the tables is largely the same.

Unlike mmssms.db, however, icing_mmssms.db is only available with an FFS/physical backup. Mmssms.db is backed up during a normal agent backup or logical extraction.

_id	conversation...	msisdn_re...	sent_timesta...	sender_send_d...	received_tim...	message_pro...
Search column...	Search column...	Search column...	Search column...	Search column...	Search column...	Search column...
64	25	+491636319247	1726750093576	+18339913448	1726750093576	3
168	43	+491636319247	1731494512137	+447423319017	1731494515931	3
537	86	+491636319247	1749722335138	apollo-optik_pj8ag4q...	1749722337534	3

Figure 3.0 (A snippet from icing_mmssms.db)

bugle_db

```
/data/data/com.google.
android.apps.messaging/
databases/bugle_db
```

Although the database structure differs from mmssms.db, its investigative value is similar. At a minimum, you should be able to identify key information such as the sender, recipient, message content, and associated timestamps.

It is also important to determine the message type, distinguishing between SMS, MMS, and RCS communications.

The Messages, Participants, and Parts tables are particularly useful starting points when examining the database during an investigation.

bugle_db is particularly valuable when examining RCS communications, as it typically contains more information and artefacts associated with this message type. However, when searching for deleted artefacts, it is important to examine all available messaging databases. An artefact may have been deleted from one database while remaining, either fully or partially, in another.

The bugle_db also contains SMS and MMS entries that overlap with those found in mmssms.db, making comparison between the databases particularly useful during an investigation.

_id	conversation...	sender_id	sender_send_...	msisdn_recei...	message_pro...	sent_timesta...
Search column...	Search column...	Search column...	Search column...	Search column...	Search column...	Search column...
27	14	18	NULL	NULL	0	1725340619000
28	14	7	NULL	NULL	5	1725340620790
63	25	7	NULL	NULL	0	1726750093412
64	25	27	+18339913448	+491636319247	3	1726750093576

Figure 4.0 (A snippet from icing_mmssms.db)

Columns of interest across all three databases

All three databases have almost identical column names. The following entries may be of interest in an RCS investigation:

Message Date: Date/time of receipt or sending of the message.

MessageID: The database row ID of the message.

Thread ID: The thread ID of the message.

Number: The number of the message partner.

Name: The name of the message partner (stored by the user).

Message type: The direction of the message (incoming or outgoing).

Message body: The text of the message.

Columns of interest, continued

Sent Date: The date the message was sent.

Last Modified Date: The date of the last change.

Geo Coded Location: The geographical location where this message was initiated.

Is read: Indicates whether this message has been read or not.

Is seen: Indicates whether this message has been seen or not.

Is locked: Indicates whether this message has been locked or not.

Error code: The error code if an error occurred during the sending process.

Most entries can be found in the “message” (see the two screenshots above) or “part” table:

While the timestamp and, above all, the protocol (SMS, RCS, etc.) can be easily found in the “message” table, there is not much information in plain text about the actual content. A “0” in the “message_protocol” column indicates an SMS. A “3” indicates an RCS as can be seen in Figure 5.0.

	_id	conversation_id	sender_id	sender_send_destination	msisdn_recei...	sent_timesta...	message_pro...	queue_insert...
	Search column...	Search column...	Search column...	Search column...	Search column...	Search column...	Search column...	Search column...
1	879	105	101	+4915110322594	+491636319247	1768470681153	3	1768470680881
1	27	14	18	NULL	NULL	1725340619000	0	1725340621095
2	28	14	7	NULL	NULL	1725340620790	5	1725340621219

Figure 5.0 (A snippet showing message_protocol)

This is where the “part” table comes in. If you match the conversationID from both tables, you will get all possible information about a sent message. Here you can also find information about the media (in the screenshot, a sent location, video, Figure 6.0, as an example).

			text	content_type	timestamp	media_modif...	longitude	latitude
	Sear	Sea	Search column...	Search column...	Search column...	Search column...	Search column...	Search column...
I	874	105	NULL	image/jpeg	1768470534714	1768470533850	0	0
II	875	105	NULL	audio/mp4	1768470561923	-1	0	0
III	876	105	https://www.googl...	application/vnd.gs...	1768470579677	-1	9.9106642	51.5238128

Figure 6.0 (A snippet showing information about the media)

The storage location of the media is also listed there (“local_cache_path” column).

By the way, if you want to know which app is set as the default messaging app on your device, take a look at android.xml. The entry for the default messaging app can be found under <string name="default_sms_package">.

RCS on iOS

In an Apple extraction, RCS messages are not stored in a separate “RCS database” but within the existing Messages structure (SMS/iMessage), mainly in the familiar sms.db (or sms-wal and sms-sha) and the associated attachment folders.

```
/private/var/mobile/Library/SMS
```

(Also extracted in a logical extraction)

If you look at these in a SQL viewer, you will see many columns in plain text, i.e. unencrypted. As SMS/MMS messages are also stored here, as mentioned above, the structure and designation are the same and usually already known. Nevertheless, there are some fields that are only used by RCS or are of particular interest to RCS. The most interesting ones are certainly:

"guid", "text" and
"account_guid", "account"

Many people are probably already familiar with these four. The GUID stores a unique ID for each message/action. The Account_GUID does the same thing, but for each user with whom you have exchanged messages, so it can occur more frequently. The telephone number/address is then entered in “account” accordingly. And “text” naturally contains the message text. If this line is empty, something other than normal text was sent here, for example a reaction, voice message, etc.

guid	account_guid	account	text
Search column…	Search column…	Search column…	Search colu
93CCD27F-17A9-40...	B2F941DE-34F8-4D2B-B392-0F...	P:+491759246...	„I hope w...
707D9DBA-0415-4...	B2F941DE-34F8-4D2B-B392-0F...	P:+491759246...	„Nope. A...

Figure 7.0 (Snippet from sms.db)

"service", "date_read" and "date_delivered"

This is where it gets interesting for RCS. In the “service” column, we can see which protocol was used to send a message (RCS, SMS, iMessage) as can be seen in Figure 8.0.

An important difference to SMS/MMS, especially from a forensic point of view, is the ability to check RCS messages against the recipient's delivery receipts to see whether messages have been delivered and/or read.

These can be tracked in the SQLite database in the same way as the delivery/read data from iMessage, in the columns “date_read” and “date_delivered”. The timestamps continue to use “Apple time*” (in nanoseconds).

service	account	account_...	date	date_read	date_delivered
sms	Search column...	Search column...	Search column...	Search column...	Search column...
iMessage	E:klara1korn@...	0517ADE3-2C2...	745360864668000000	745361282194657280	745360864974547584
RCS	P:+491759246...	B2F941DE-34F8...	745360870581358976	745652302664076032	745652302036316928
SMS	E:	BC8AE97E-D73...	745329544526881024	0	0

Figure 8.0 (Snippet from sms.db)

What else can we find here?

"is_audio_message" and "date_played"

For example, the “is_audio_message” column in connection with “date_played”.

As the name suggests, if there is a “1” in the “is_audio_message” row, this message is a voice/audio message. The “date_played” row then shows when the audio message was listened to. Very interesting for forensic investigators.

	attri...	account	account_guid	date	date_read	date_delivered	is_audio_mes...	date_played
rcs	Search colum	Search column...	Search column...	Search column...	Search column...	Search column...	Search column...	Search column...
RCS	362 Bytes ...	P:+491759246938	B2F941DE-34F8-4D...	745686701745546944	745686070327257216	0	1	745691014192093056

Figure 9.0 (Snippet from sms.db showing audio and date played information)

"associated_message_guid" and "associated_message_type"

These include links for replies, reactions ("tapbacks") and edits.

service	associated_message_guid	associated_message_type	text
Search column	Search column…	Search column…	Search column…
RCS	p:0/C054DBE2-22CE-4650-B7…	2000	„I hope we can decode it" added a heart
RCS	p:0/1E856F78-1123-414C-A08…	2003	„Nope. Android is the way to go…" added a laugh

Figure 10.0 (Snippet from sms.db showing replies etc)

Other columns and tables are already familiar to examiners from many years of SMS/MMS analysis and are mostly self-explanatory based on their names, such as "_deleted_" or "_message_".

And just as with SMS/MMS, if desired, you can map the attachment tables and folders to messages and use SQLite analysis to see if you can find/organize deleted or unparsed RCS fragments.

The sms.db is backed up both during a logical backup and during an FFS backup.

RCS messages are of course also stored in local or iCloud backups, if desired/set. The structure does not change.

A final note on validation

The artefacts, database structures, fields and file locations described in this blog were accurate and observed during testing at the time of writing. Mobile operating systems, applications and RCS implementations are constantly evolving, meaning that database schemas, storage locations and available artefacts may change between devices, software versions and extraction methods.

As with any digital forensic examination, never rely on a single artefact, database or forensic tool to support a conclusion. **Always** validate your findings against the underlying source data and, wherever possible, corroborate them using additional artefacts.

Tools parse the evidence.

The examiner validates it.

Our Vision:

Mobile technology changes the world. There are billions of mobile devices such as phones, tablets, GPS units and watches. Even where there are no computers, people use mobiles: in the cities, in the countryside, in deserts and mountains. Wherever we go, whatever we do, we bring our mobile devices.

This is also true for people who break the rules of society. They use mobile devices and that can be a good thing. Mobile technology leaves traces, like digital footprints. These can reveal a hidden world — a fragmented world at times, but a world that points to the truth. It is our task to help our customers search for that truth, in order to make the world a safer place.

Our customers are passionate about their jobs. They truly believe in justice and peace. However, when it comes to mobile technology they meet tough challenges. There is a constant flow of new devices, new operating systems and new apps. Staying ahead of these challenges is absolutely critical for our customers.

MSAB takes responsibility for supporting our customers with the best possible solutions for mobile forensics. This means that we also need to stay ahead of the game; with our products and services, with our vision and energy, with our people and how we do things. We are the pioneers of our industry. Our vision is simple: MSAB – Exceeding the limits of technology to find the truth.

MSAB
Trusted Partner in Digital Forensics