Forensic analysis of Email on Android volatile Memory

Long Chen
Institute of Computer Forensics
Chongqing University of Posts and Telecommunications
Chongqing, China
chenlong@cqupt.edu.cn

Yue Mao
Institute of Computer Forensics
Chongqing University of Posts and Telecommunications
Chongqing, China
maoyue512@163.com

***Abstract*—With the popularity of smart phones and the emergence of the mobile office mode, the traditional email forensics that works for computer has been already unable to satisfy the demands of reality, so forensic work needs to be expanded to a range of mobile devices, such as mobile phone, tablet, etc. In this paper, we will focus on examining if we can discover email-related information in the volatile memory of the mobile phone. Specifically, we choose Android mobile as a research focus, and two Chinese mainstream Android email applications—MailMaster and QQMail are as email client to the experimental test. Finally, we not only sort out the email-related information stored in the volatile memory, but also identify the patterns of the information saved in the memory. Moreover, based on these patterns, we also develop a tool named EmailFinder that can automatically extract the email-related information from memory dump. It can be utilized as a forensic tool on Android phones to assist forensic investigators retrieve email-related evidence from memory dump.**

Keywords—email forensics; volatile memory; email client; patterns; email-related information

I. INTRODUCTION

Email forensics mainly includes three parts: evidence collection, evidence analysis and evidence submission, the most critical steps are extracting and analyzing the email-related data which is stored on a variety of devices. However, we are in a world that smart phones and mobile office mode are popular, which means the traditional email forensics that applied to computer has been already unable to meet practical needs, so the forensic work needs to be expanded to numerous mobile devices, such as mobile phone, tablet, etc. The statistics report of Radicati Group [1] shows that by the end of 2018, the total number of global mobile email users is expected to exceed 2.2 billion. This also indicates that the mobile email forensics will gradually become one of the research focuses of email forensics.

However, compared with the large volume of computer hard drives, mobile phone built-in storage capacity is limited. What's more, email-related data is usually not stored in the non-volatile memory (e.g., NAND flash memory, SD card, etc.) of mobile phones. Not only that, but as an increasing number of mobile phones support disk encryption function, the contents of the disk dump acquired from the phone faces the risk of being encrypted, which leads to the failure of data extraction. It is worth noting that, in addition to the non-volatile memory, the volatile memory(RAM) of computer or mobile devices also stores a large amount of data information. Joseph [2] extracted the email address, email password, and other email-related information from the physical memory dump file of the computer. Similarly, we also consider whether we can discover such information in the volatile memory of the mobile phone.

At present, majority of researches and analyses of mobile phone forensics are concentrated on the non-volatile memory (e.g., the Subscriber Identity Module, memory cards, the internal flash memory, etc.) [3]. On the contrary, Android volatile memory related forensic research is still in its initial stage. In particular, to the best of our knowledge, so far there is no research on analysis and extraction of email-related information from the volatile memory of the mobile phone. So this paper will focus on examining if we can discover email-related information in the volatile memory of the mobile phone. If possible, we will analyze the patterns of data storage to realize the automatic analysis and extraction of the information.

In this paper, we choose Android mobile as a research focus, because it occupies most of the mobile phone market, and two Chinese mainstream Android email applications—MailMaster and QQMail are as email client to the experimental test. The main contribution of this paper includes:

1) Examine each investigated email client application to know whether we can discover email-related information in the volatile memory of the mobile phone.

2) Determining the patterns of how the email-related information are stored in the memory dump.

3) According to the patterns, we develop a tool named EmailFinder that can automatically extract email-related information from the memory dump of the mobile phone.

In the rest of the paper, Section II gives the background information of the research. Section III introduces the related work. Section IV describes the procedure for the acquisition of the volatile memory of the Android mobile phone and the details of experiments. Section V presents the results of experiments. Finally, the conclusion and future work are given in the Section VI.

2324-9013/16 $31.00 © 2016 IEEE
DOI 10.1109/TrustCom.2016.158

II. BACKGROUND

A. Email grammatical structure

Email includes two parts, the message header and the message body. The message header contains the sender, recipient, subject, time, MIME version, the type of mail content and other important information. Each piece of information is referred to as a field, which is composed of field name and field value. The common field name of message header and its meaning are shown in Table I, in which "***Received***" and "***Massage-ID***" are two important fields. "***Massage-ID***" is the identification number assigned by the sender side mail server to the mail. This number follows the mail from beginning to end, and it is unique, that is to say two different mails will not have the same Massage-ID. "***Received***" generally includes, the sending time of the mail, the IP address of mail sender, as well as the IP address of the sender side mail server.

TABLE I. COMMON MESSAGE HEADER FIELDS AND ITS MEANING

Field name	***Field Description***
Sender/From	The email address of the sender.
Return-Path/Reply-To	Address that should be used to reply to the email.
Delivered-To/To	The email address of the recipient.
Cc	Generally same as To Field. Generally a To field specifies primary recipient who is expected to take some action and CC addresses receive a copy as a courtesy.
Bcc	Address of recipient whose participation is not disclosed to recipients specified in To and CC addresses.
MIME-Version	Indicates that the email is MIME-formatted.
Status	Identify the status of the email (whether the new mail, read, etc.).
Received	Contains trace information that includes originating host, Mediators, relays, and MSA(Mail Submission Agent) host domain names and/or IP addresses.
Date	Record the creation date and time of the email.
Subject	It describes the subject or topic of the email.
Message-ID	Globally unique email identification string generated when it is sent.
Content-Type	Indicates the media type of the email content.
Content-Transfer-Encoding	Indicate the type of transformation that has been used in order to represent the body in an acceptable manner for transport.
X-*	Email extension fields, non-standard fields created by the developer.

Message headers are the important part for investigating email messages. Al-Zarouni [4] and Banday [5] analyze the specific meaning of each field in the message header in detail, dig out useful information that found the sender of the email, and apply it to the tracing investigation on the source of spoof emails. The latest research, Nurse [6] suggests that the message header of outgoing emails could leak the sensitive information of corporation and personality.

The message body contains the contents of the message, and its type is pointed out by the "***Content-Type***" field of the header. Common types includes text/plain (plain text), text/html(hypertext), application/octet-stream (binary stream data), multipart/alternative (the copy of hypertext), multipart/mixed (attachment), multipart/related (embedded resources). The email body can be divided into a plurality of segments, each segment also includes two parts, the segment header and the segment body.

B. Android operating system

Android is an open-source mobile phone operating system based on the Linux kernel, currently developed by Google. It has gradually utilized in terminals such as tablet PC, TV, digital cameras, game consoles and other mobile terminal devices. The memory of the Android phone can be roughly divided into two types, each of which serves different purposes.

1) The volatile memory (RAM), just like RAM of computer, the stored data will disappear while you restart your device. Its main function is to store the dynamic data of the systems and the applications at running time, which contains a lot of important information, such as username, password, encryption keys, the activity information of network and application data, etc.

2) The non-volatile memory, refers to the internal storage (NAND flash memory) and equipment external extensible storage device such as an SD card. The data stored is still there, even after shutdown or restart. This type of memory is mainly used to store the static data of system files and user's data file.

Previous studies show that some mobile applications data presents itself in the volatile data only (e.g., financial applications) and no trace of evidence could be found in the non-volatile memory. For privacy and security considerations, most of the applications data stored in the non-volatile memory may have been encrypted. Moreover, after version 4.0 of Android, Android smart phones begin to support disk encryption feature to transparently encrypt user partitions [7]. On the contrary, the content of the volatile memory present in the form of plaintext.

III. RELATED WORK

At present, most of the researches and analyses of the mobile phone forensics are focused on the non-volatile memory. On the contrary, the research of Android volatile memory forensics is still in its initial stage, it mainly includes memory acquisition and memory analysis.

A. Android volatile memory acquistion

Researches on acquiring the volatile memory of Android mobile phone is very limited. Thing [8] first mentioned the important role of the volatile memory of the mobile phone in the forensic investigation process, and developed a specific process memory acquisition tool memgrab for Android, which can gain access to specific process's address space through performing the Process Trace (ptrace) system call. However, this tool cannot acquire the complete volatile memory from Android phone. Leppert [9] described the way of acquiring a heap-dump for Android 2.3 till Android 4.0 by using the DDMS(Dalvik Debug Monitor Server) [10] tool, but this method can only employ in the Android emulator environment rather than real mobile devices. In the latest research, Sylve [11] developed a Loadable Kernel Module for dumping the volatile memory of Android phone, named Droid

Memory Dumpstr (DMD) or known as Linux Extraction Memory (LiME), and so far it is a unique tool that allows complete memory acquisition from Android phone. LiME supports the feature that dumps the memory directly to SD card or over the network. Moreover, LiME also minimizes interaction between the user land and the kernel land processes during acquisition which allows it to produce memory captures, and it is more forensically sound than other methods [3]. Although the LiME still has some limitations (e.g., require root privileges), it has become the most commonly used method to acquire the volatile memory of Android system [12].

B. Android volatile memory analysis

Though the analysis on volatile memory has been researched for a few years, most of them are based on x86 architecture of Windows and Linux operating system. In order to analyze the volatile memory of Linux, a popular forensic investigation framework called Volatility [13] is usually employed. Since Android at the kernel level shows no significant difference from a Linux system, the Linux volatile memory analysis method can be used to analyze Android memory dump. Volatility enables the extraction of digital evidence from a memory dump, but it supports a limited set of analysis capabilities, such as extracting running processes, open network sockets, memory maps for each process, or kernel modules, etc. For those applications such as MailMaster, QQMail, we are unable to get the data structures. In the last few years, there are some of the volatile memory analysis researches targeting Android phone. Ntantogian [14] has proved that it is possible to recover users' authentication credentials (e.g., username and password) of mobile applications (i.e., mobile banking, financial applications, password managers, etc.) from the volatile memory of rooted Android mobile phone. In order to circumvent the problem that the chats stored in non-volatile memory are encrypted or unrecoverable after deleting, Zhou [15] recovered the WeChat chat record information from the volatile memory dump of Android mobile phone. Moreover, Andersen [16] retrieved the encryption keys of LUKS from the volatile memory of Android mobile phone. In particular, to the best of our knowledge, so far there is no research on analysis and extraction of email-related information from the volatile memory of mobile phone.

Other related research work, Müller and Spreitzenbarth [17] proved that it was practical to perform cold boot attacks against smart phones, and developed a data recovery tool named FROST(Forensic Recovery of Scrambled Telephones). Forensic investigators with physical access to an encrypted Android device that is running but locked can perform cold boot attacks to reconstruct personal information including personal messages, photos, passwords and the encryption key from RAM with FROST. The defect of this method is that it needs to unlock the bootloader first in order to boot FROST, and the unlocking procedure would wipe all personal data of user partition. Moreover, this method needs to reboot the Android smart phone by unplugging the battery briefly, however, lately a lot of smart phones are becoming unibody, meaning that we cannot open it up, or remove the battery.

IV. EXPERIMENTS

A. Volatile memory acquisition method

From the prior art, we decided to use an open-source and free forensic tool called Linux Memory Extractor(LiME) to capture the volatile memory dump of a rooted Android mobile phone. LiME (formerly known as DMD) is a loadable kernel module, up to now, it is a unique tool that allows full memory capture from Android mobile phone [18]. Loading the kernel module into the OS kernel requires root permissions, so the mobile phone must be rooted before executing LiME. The root privileges in Android mobile phone is not granted to users by default due to a security mechanism. However, according to the latest survey, a growing number of Android smart phone users root their mobile phones in China for a different purpose. In particular, 80% Chinese Android smart phone user respondents rooted their smart phones in 2014 [19]. In this paper, we assume that the Android mobile phone used in every experiment has been rooted，and we get physical access to it. And other cases are not discussed.

Specifically, in the process of using LiME to obtain Android mobile volatile memory, it firstly needs to download the kernel source code of the mobile phone used in experiments. Secondly, according to the operating instructions document [20, 21] on LiME, cross-compile the source code of LiME to generate the lime.ko file, but it failed. Through research and analysis of the kernel source code, we get a resolution, that is in order to ensure successful cross-compiling and generate lime.ko, it needs to modify the Makefile file of the kernel source code so that the value of the variables (e.g., 'VERSION', 'PATCHLEVEL', 'SUBLEVEL' and 'EXTRAVERSION') are consistent with the specification of the mobile phone used in experiments. Finally, we copy the lime.ko module to the phone's SD card using Android Debug Bridge(ADB) [22], and execute the *insmod* command to install it. After a period of waiting, the acquisition process is completed. In order to perform the analysis, it is required to copy the dump to the host device by the Android Debug Bridge(ADB).

B. Experimental environment and conditions

To capture the memory dump with LiME, additional preparations are required. In this paper, experiments are based on a rooted Nexus Galaxy (I9250) which is the third smart phone in the Google Nexus series, and runs Android4.04 system. We can download the kernel source code from official website of Google. Moreover, through the investigation on the Chinese email application market, we choose MailMaster [23] and QQMail [24] that are two most mainstream email applications as email client to carry out experimental tests. Both applications support a variety of common protocol of mail services, such as 163, QQ, Sina, Sohu, Hotmail, Gmail mailbox, and apply to different systems (i.e., Android, iOS, etc.). It should be noted that the applications update frequently, so in this paper the application version selected for experiments is currently latest version. In each experiment, we mainly study and analysis the email-related information in the volatile memory dump of Android mobile under six various mobile usage scenarios (as shown in Table II). To

avoid the interference of experiments, after finishing the test of MailMaster-related scenarios, the battery will be unplugged immediately and plugged in a few days to erase most of data stored in memory, next, the test of QQmail-related scenarios will be implemented.

TABLE II. SUMMARY OF EXPERIMENTAL SCENARIOS

Scenarios	*Step description*
scenario# 1	Login MailMaster, use (including browse, edit, send and receive email, etc.) it, let the application run in the background, and then acquire the memory dump with LiME.
scenario# 2	Login MailMaster, use (including browse, edit, send and receive email, etc.) it, logout the application and then acquire the memory dump with LiME.
scenario# 3	Login MailMaster, use (including browse, edit, send and receive email, etc.) it, reboot the phone and then acquire the memory dump with LiME.
scenario# 4	Login QQMail, use (including browse, edit, send and receive email, etc.) it, let the application run in the background, and then acquire the memory dump with LiME.
scenario# 5	Login QQMail, use (including browse, edit, send and receive email, etc.) it, logout the application and then acquire the memory dump with LiME.
scenario# 6	Login QQMail, use (including browse, edit, send and receive email, etc.) it, reboot the phone and then acquire the memory dump with LiME.

V. EXPERIMENTAL RESULTS

A. Detailed volatile Memory analysis

Once the volatile memory, under six various mobile usage scenarios, is successfully acquired from the Android mobile phone, detailed analysis is needed to examine if we can discover email-related information in the memory dump. The detailed analysis results are as follows.

1) Username and password information

Through analyzing the memory dump (obtained in scenario 1), we find that it is possible to extract the email's usernames and passwords of the email client (MailMaster) from the memory dump. For example, as shown in Fig. 1, the username is found after a keyword "uid=" and password is found after "passwd=" keyword. However, through analyzing memory dump obtained in scenario 4 the email client (QQMail) will hash or erase the password, we cannot directly find the password information stored in plaintext in the memory dump. Password information is not discovered in other scenarios.

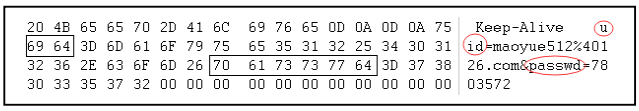

Fig. 1. Username and password of the email client(MailMaster).

2) Email client information

By analyzing memory dump acquired from scenarios 1 and 2, client information is gained. For example, Fig. 2. reveals a reasonable amount of information on email client (MailMaster) searched in the memory dump. Here, the email client (MailMaster) can be identified as well as its version number (4.7.2).

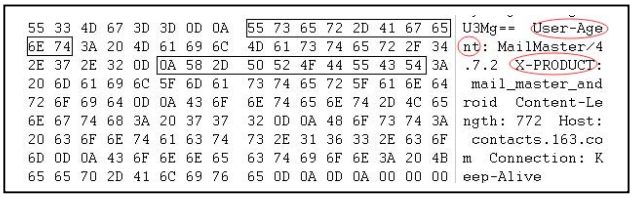

Fig. 2. Email client (MailMaster) information.

Through analyzing memory dump acquired from scenarios 4 and 5, client information is gained. For example, Fig. 3. illustrates a large amount of information on email client (QQMail) found in the memory dump. Here, the email client(QQMail) and its version number(5.0.1) can be identified. Moreover, the operating system (Android) of the device used by the sender along with its version number(4.0.4) can also be distinguished.

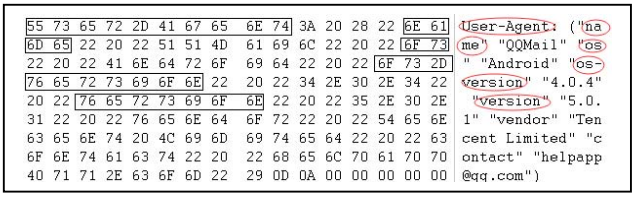

Fig. 3. Email client (QQMail) information.

3) Message header and email body information

By analyzing memory dump acquired from scenarios 1 and 2, message head information is gained, For example, Fig. 4. describes the message header information found in the memory dump after the MailMaster logout. It not only includes the "***Subject***", "***From***", "***To***", "***Date***", "***MIME-Version***", "***Message-ID***", "***Content-Type***" but also other message header fields, and all these data is stored in plaintext which is considerably valuable. Besides, through analyzing memory dump acquired from scenario 1, the whole email body information is also gained. As described in Fig. 5, the content between two red circles on the left is the email body information.

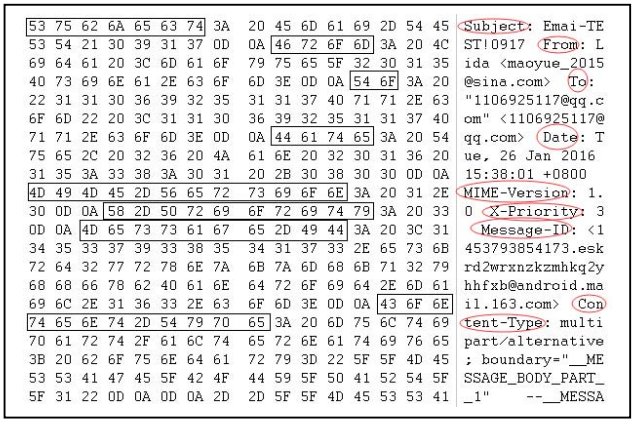

Fig. 4. Message header information of email client (MailMaster).

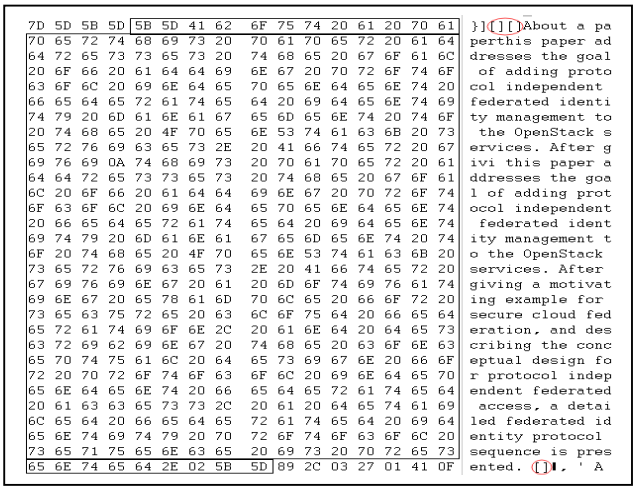

Fig. 5. Email body information (MailMaster).

Through analyzing memory dump acquired from scenarios 4 and 5, message header and abstract information is gained. For example, Fig. 6. represents the message information found in the memory dump after the QQMail logout. The keyword "subj" refers to the subject or topic of the email, "abs" implies the abstract of the email, which is partial content beginning with the email body. Different from results in scenarios 1 and 2, the whole email body information is not obtained in scenarios 4 and 5. "date" indicates the creation date and time of the email, "from" stands the email address and nickname of the sender, "toLst" conveys the email recipient information including email address and the nickname, "rly" expresses the response times of the email.

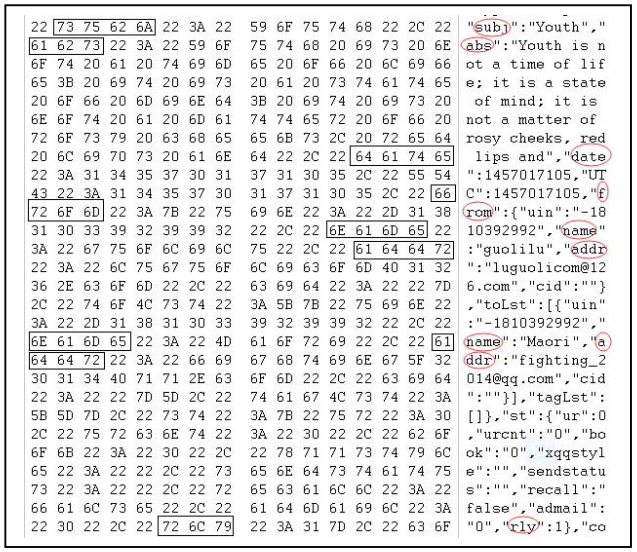

Fig. 6. Message header information of email client (QQMail).

In experiment scenarios 3 and 6, as the phone have no reset button, we unplug the battery and plug it instantly to reboot the phone and acquire memory dump. Analyzing memory dump obtained from scenario 3, we discover that although the password, email client and message header information are not gained like in scenarios 1 and 2, some mutual contact of email information and partial email body information are acquired. By analyzing trace in memory dump of tens of email, we find that mutual contact of email information is generally illustrated like in Fig. 7. Not only sending address and receiving address, but also most email body information is included in Fig. 7. Moreover, almost 20% email has kept complete email body information. However, analyzing memory dump in scenario 6 accordingly, trace of email-related information is not found, as shown in Fig. 8, only email address of sender and email body fragment information are obtained.

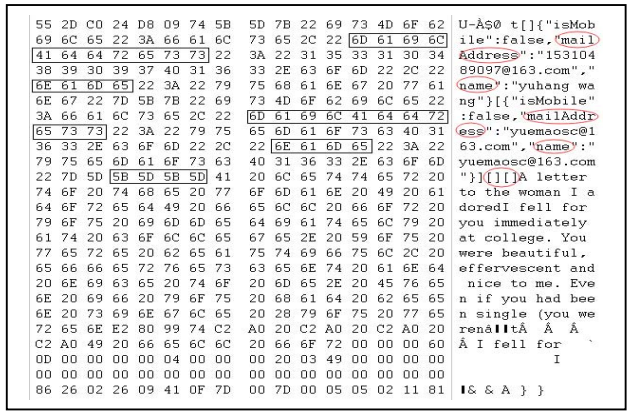

Fig. 7. Residual information after rebooting(MailMaster).

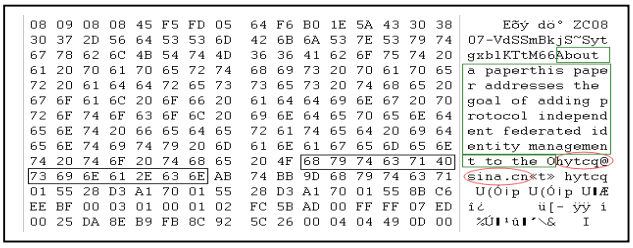

Fig. 8. Residual information after rebooting(QQMail).

4) Other email-related information

In addition to the email client(e.g., MailMaster and QQMail) related information, we also find the device(Nexus Galaxy) information and the network information of device from the memory dump of the mobile phone. The device information refers to the model of mobile phone, manufacturers, operating system, CPU model, and other information. The network information of device includes device network access mode (WiFi or mobile network)and the telecom operators. In our experiments, the mobile phone accesses the Internet via WiFi, and the experimental results show that we can discover the name of the WiFi hotspot and the corresponding IP address information from the memory dump.

Through the analysis of the memory dump obtained under the six experimental scenarios, the analysis results show that there is a large amount of email-related information in the memory dump of mobile phone, even the email password stored in clear text. Moreover, in the same experiment

condition, email-related information acquired in scenarios 1, 2 and 3 is much more and complete than in scenarios 4, 5 and 6. This shows that QQMail has a higher security level than MailMaster, yet forensic investigators cannot take that as advantage. Furthermore, besides six scenarios experiments mentioned above, in further experiments, we found that these information would preserve for a long period of time in memory, even restarting the phone after power off (remove its battery) for a few days. In the memory, there still exists some email-related information (mainly include email address, IP address and other information).

B. Extract Email information with EmailFinder

Although the results of the experiments confirmed that we can discover a great deal of the email-related information in the memory dump, manually looking into and analyzing the memory dump file of huge size is not practical and is very time consuming. Fortunately, we find that the email-related information being stored in some consistent patterns (model retains unchanged by long standing experiment test and various software version test)indicates where the information are located in the memory dump. Moreover, according to the patterns, we develop a tool named EmailFinder that can automatically extract email-related information from the volatile memory dump of the mobile phone.

1）MailMaster- related information

Through searching pattern "{"bind":------}" in the memory dump, account information generated by logining MailMaster can be located, which mainly include email's username, device type and MailMaster's version. All these information above are shown in Fig. 9. Moreover, log information obtained by utilizing MailMaster to receive email successfully, which contain sending address, receiving address, subject, sending time, receiving time, size of the email and if it is open, can be found by searching pattern"[{ --{--"rcptSucceed":true,--}--}]". Fig. 10. illustrates the log information. And, searching pattern "[]{"isMobile":---}[{"isMobile":---}][][]--[]"can locate complete information of an email received by MailMaster.

```
==================== Account Info ====================
Account 1 :
        Account:        maoyue512@126.com
        sid:            rBanvWpnMaEkrASUXcnngKpNVqiblDQE
        language:       zh
        token:          79297ad4322e049e0003032cd6084159
        deviceType:     Android
        SubscribeType:  mailmaster
        App Version:    4.7.2
```

Fig. 9. Account information after logining MailMaster.

```
==================== Mail Recv Info ====================
Mail Recv 1 :
        From      :  "MaoRi" <maoyue512@126.com>
        To        :  "maoyue512@163.com" <maoyue512@163.com>
        Subject   :  happy new year!
        SendDate  :  2016-01-25 22:24:44
        RecvDate  :  2016-01-25 22:24:45
        Size      :  1347
        IsRead    :  True
```

Fig. 10. Log information of email received by MailMaster.

2）QQMail-related information

Searching pattern "User-Agent:(------)"can locate QQMail-related information covering version of QQMail, type and version of operating system etc. As described in Fig. 11. What's more, complete information of an email received by QQMail ranging from email sending address, sender's nickname, receiving address, receiver's nickname, to times of reply and forward, abstract of the email, subject and sending time of email etc can be found by searching pattern ""exname"---"from":{-----},"toLst":[{---}],"tagLst":[]},"st":{-----}". As implied in Fig. 12.

```
==================== User Agent ====================
User Agent 1 :
        name       :  QQMail
        os         :  Android
        os-version :  4.0.4
        version    :  5.0.1
        vendor     :  Tencent Limited
        contact    :  helpapp@qq.com
```

Fig. 11. Information of QQMail client.

```
==================== Mail Info ====================
Mail Info 1 :
        From name :  guolilu
        From addr :  luguolicom@126.com
        To name   :  Maori
        To addr   :  fighting_2014@qq.com
        IsReply   :  1
        IsForward :
        Subject   :  Youth
        Abstruct  :  Youth is not a time of life;
                     it is a state of mind;
                     it is not a matter of rosy cheeks,
                     red lips and
        Send_Date :  2016-03-03 22:58:25
```

Fig. 12. Email information received by QQMail.

VI. CONCLUSION

In this paper, we focus on examining if we can discover email-related information in the volatile memory of the Android mobile phone. Specifically, we choose the MailMaster and QQMail that are two most mainstream email applications as email client to carry out the experimental tests. The analysis of experimental results shows that there are a lot of email-related information in the volatile memory of the mobile phone, including email address, email password, message header information, message body information and the devices information (e.g., model, operating systems, etc.), network information (e.g., WiFi hotspots, telecom operators , etc.). Furthermore, in further experiments we found that these information would preserve for a long period of time in memory, even restarting the phone after the power off (remove its battery) for a few days. In the memory, there is still some email-related information (mainly include email address, IP address and other information). After analyzing the memory dump under the six categories of experiments, we not only sort out the email-related information stored in the volatile memory, but also identify the patterns of the information stored in the memory. Moreover, based on these

patterns, we also develop a tool named EmailFinder that can automatically extract the email-related information from the memory dump. The work of this paper can provide a strategy for forensic investigators, in circumstance like incapable of acquiring email-related information in Android non-volatile memory due to deletion or encryption, extracting email-related information from volatile memory can be considered. In specific examples, the EmailFinder mentioned in this paper can be used as a forensic tool on Android phones to assist forensic investigators retrieve email-related evidence from memory dump. Moreover, as mentioned above, volatile memory of phone is becoming a valuable evidence data resource. Further study on related field can adopt similar analysis method, that is to extract trace information in volatile memory of other applications in Android phone.

Although the volatile memory of mobile phone is of great value for the forensic investigation, the fragmentation is a serious problem in Android device. The manufacturers tend to customize the own Android OS to stand out in the Android marketplace, so that there is no general solution or approach for every type of Android smart phones. Encryption techniques are gradually applied in Android phone, the data forensic investigators acquired can possibly be encrypted, which brings a big challenge to forensic work. In the future work, we prepare to study on the method for extracting encryption keys from the memory dump of the Android mobile phone.

ACKNOWLEDGMENT

The authors would like to thank the reviewers for their detailed reviews and constructive comments. This work was supported by the National Social Science Committee of China (No. 14BFX156) and partially sponsored by New Direction Cultivation Program of Chongqing University of Posts and Telecommunications (No. A2015-45).

REFERENCES

[1] Radicati Group, Inc. “Email Statistics Report, 2014-2018,” 14-April-2016; http://www.radicati.com/wp/wp-content/uploads/2014/01/Email-Statistics-Report-2014-2018-Executive-Summary.pdf.

[2] Joseph, Neethu, et al. "Volatile Internet evidence extraction from Windows systems." Computational Intelligence and Computing Research (ICCIC), 2014 IEEE International Conference on. IEEE, 2014.

[3] Heriyanto, Andri P. "Procedures and tools for acquisition and analysis of volatile memory on android smartphones." (2013)

[4] Al-Zarouni, Marwan. "Tracing E-mail Headers." Australian Computer, Network & Information Forensics Conference. 2004.

[5] Banday, M. Tariq. "Analysing e-mail headers for forensic investigation." The Journal of Digital Forensics, Security and Law: JDFSL 6.2 (2011): 49.

[6] Nurse, Jason RC, et al. "Investigating the leakage of sensitive personal and organisational information in email headers." Journal of Internet Services and Information Security (JISIS) 5.1 (2015): 70-84.

[7] Google. “Ice Cream Sandwich,” 14-April-2016; http://developer.android.com/about/versions/android-4.0-highlights.html#UserFeatures.

[8] Thing, Vrizlynn LL, Kian-Yong Ng, and Ee-Chien Chang. "Live memory forensics of mobile phones." digital investigation 7 (2010): S74-S82.

[9] Leppert, Simon. "Android memory dump analysis." Student Research Paper, Chair of Computer Science 1 (2012).

[10] Google. “Using DDMS,” 14-April-2016; http://developer.android.com/tools/debugging/ddms.html.

[11] Sylve, Joe, et al. "Acquisition and analysis of volatile memory from android devices." Digital Investigation 8.3 (2012): 175-184.

[12] Wachter, Philipp, and Michael Gruhn. "Practicability study of android volatile memory forensic research." Information Forensics and Security (WIFS), 2015 IEEE International Workshop on. IEEE, 2015.

[13] Andrew Case. “Volatility Foundation” 14-April-2016; https://github.com/volatilityfoundation.

[14] Ntantogian, Christoforos, et al. "Evaluating the privacy of Android mobile applications under forensic analysis." Computers & Security 42 (2014): 66-76.

[15] Zhou, Fan, et al. "Dump and analysis of Android volatile memory on Wechat." Communications (ICC), 2015 IEEE International Conference on. IEEE, 2015.

[16] David R. Andersen. “Cracking LUKS on Android Phones,” 14-April-2016; https://dx.eng.uiowa.edu/dave/luks.php.

[17] Müller T, Spreitzenbarth M. Frost[C]//International Conference on Applied Cryptography and Network Security. Springer Berlin Heidelberg, 2013: 373-388.

[18] Joe Sylve. “504ensicsLabs/LiME,” 14-April-2016; https://github.com/504ensicsLabs/LiME.

[19] China Internet Watch, “80% China’s Mobile Users Rooted Smartphones in 2014,” 14-April-2016; http://www.chinainternetwatch.com/12926/80-china-smartphone-users-rooted/.

[20] Google, “volatility - AndroidMemoryForensics.wiki,” 14-April-2016; https://code.google.com/archive/p/volatility/wikis/AndroidMemoryForensics.wiki.

[21] The Lulz Kittens. “Pulling Memory off an Android Device,” 14-April-2016; http://thelulzkittens.blogspot.jp/search?q=Lime.

[22] Google, “Android Debug Bridge,” 14-April-2016; http://developer.android.com/tools/help/adb.html.

[23] NetEase, Inc. “MailMaster,” 14-April-2016; http://mail.163.com/dashi/.

[24] Tencent. “QQMail,” 14-April-2016; http://app.mail.qq.com/.