Online Communication Traces in Android Memory

Jørgen Ellingsen, Espen Kjellstadli Lund, Ingrid Johanne Kjesbu Larsen, Jan Petter Berg Nilsen, Emil Volckmar Ry, Anders Sefjord Torbjørnsen, Magnus Omland Torgersen
{jorgeell, espenklu, ingridjl, jpnilsen, emivr, andetorb, magnuoto}@stud.ntnu.no

ABSTRACT

Write a short abstract here.

Keywords

Android; Memory; Forensics; Practical approach

1. INTRODUCTION

Today the battery life of smartphones has reached the point that even with all the usage we do on our phones today, it will in most cases last the entire day. As a result of this, a smartphone can stay on for weeks, or even months at a time and old information can still be present in the memory. As most of our communication today are done with mobile applications like Facebook, GMail and Snapchat this project tries to analyze what information is stored in memory, and how long it is recoverable for the different applications.

For data to be used by the processor in a smart phone it has to be present in memory while being presented or used. This project aims to examine what happens when the applcation is closed, and the data is no longer in use. For the traces to be gone the memory must not only be freed up for other applications to use, but overwritten by new data. If not, the data will survive until the device is rebooted or new applications allocate and overwrite the now free space. Additionally, some applications claim to have features like anonymity and auto-delete functionality, and applications like this should take extra care to make sure there is no fragments of the communication left in memory after the application is closed.

2. BACKGROUND

Many of todays smart phones are running Android as their operating system, and data claims it dominates the market with an 87.6% share in the second quarter of 2016[1]. Making it essential for future mobile forensic work, for gathering information in an investigation.

When an investigation occurs, there are several approaches to data acquisition: (1) Manual acquisition, (2) Logical acquisition, (3) Physical acquisition, (4) Brute force acquisition. The field of interest for this paper is physical acquisition of the primary storage also known as random access memory(RAM). In essence it consists of creating a bit-by-bit copy of the physical storage(memory dump). Due to RAM being volatile, it is preferred to store information about illegal activities there, to reduce the amount of traces being left behind. If data was stored in a hard drive, it would still be resident until OS tries to overwrite the same physical area. The point being that data residing on a secondary storage device, will be living longer and probably be logged more extensively.

The Android OS is in short words a Linux-based OS, where most OS tasks are performed by open source C libraries, and Java is used for the development of Android applications. These applications are compiled to bytecode for the Java virtual machine (JVM), which is then translated for a second virtual machine which executes them. Depending on which version of Android a smart phone is running, different virtual machines are used for execution. For Android versions 4.4 and prior, Dalvik was used. It was replaced by ART in 4.5, and is still the de-facto standard.

Why does this matter? The virtual machines that are running Java applications, have gained several features of the Java programming language. One of these features are the memory management module, which has a built-in garbage collector(GC). It lets the user create objects without worrying about memory allocation and deallocation. Reducing the need for boilerplate code, and problems with memory leaks and such, which often are languages like C and C++ are subject to.

The GC attempts to reclaim garbage, or make space for new objects. Depending on how it is implemented, it might remove dead data from applications. Same can be said about the trigger mechanism, however it is often based upon size limits and collision detection.

Both ART and Dalvik use by default a method called "concurrent mark and sweep" for their GC[2, 3]. It works by traversing the heap for objects that are "reachable" or used by applications, those who are not will be regarded as free space again. Making space for potential new data to be stored within it. A figure of the process, can be seen at figure 1. Generally the heap is a region of the memory that is regarded as free memory for any process to use, however in Java it is used to store all objects created. Therefore, the GC may potentially remove one of the better sources for information, the objects. The good news are that once data is freed, no overwriting is enforced; in other words, data is not overwritten until another object takes its place[3]. Furthermore, each application is running with its own GC and private heap, potentially delaying other applications from running their GCs.[4] Creating room for better life expectancy.

Because all objects created in the heap, there may be a po-

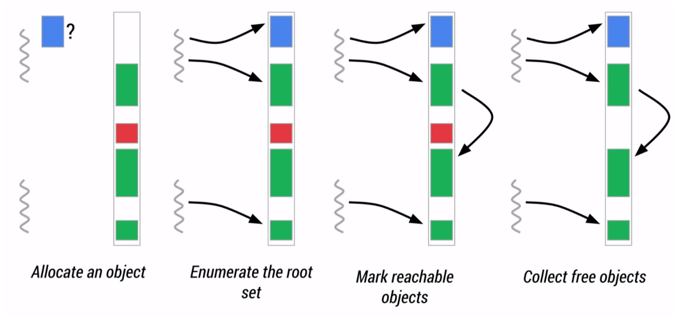


Figure 1: Concurrent Mark and Sweep[2]

tential trace of information of an application. Through the use of several popular online communication applications, the amount of data that can be collected through a memory dump will be explored; despite knowing that the GC might start, or extra security measures might prevent us to do so.

2.1 Forensics Process

A digital forensics process is used in order to conduct the most thorough analysis of the data of a device, and prevent investigators from making mistakes. It also helps ensure that every step of the process is documented. Årnes et al. describes a process with five stages: Identification, collection, examination, analysis and presentation[5]. The process in itself is an iterative process, and specifically the phases of collection, examination and analysis are done over and over again in some cases. The digital forensics process presented here is a generic process, which means it can be used on every digital device there is. By using a digital forensics process like this, the investigators are able to keep the integrity of the data and also document chain of custody which is important when a case is going to court.

Preparation is important in the phase of identification. When looking for a mobile device, it is essential to bring equipment to charge the device in case it is turned on (live), because there may be data on the device that will disappear if is turned off, for instance data in the RAM. Devices that are turned off (dead) should not be turned on, because it can destroy data. Mobile devices that are live at the crime scene should be put in a box that isolates it from networks and blocks signals. The box will prevent the mobile device from for example remote deletion or tampering of the data on it[5].

The collection phase consists of retrieving data from the device (or devices) that are brought in by the investigators[5]. Brezinski and Killalea[6] suggests evidence collection to be done in an order of volatility. In the case of a mobile device, the RAM should therefore be collected before the primary storage. This is done by copying data over to another system from the mobile device. While a regular copy may cause the system to overwrite existing data, write-blockers are used in order to deny the system anything other than read access, which makes a copy doable without interfering with the existing content on the device. A hash of both the copied device and the copy itself is then created, and compared so that it is no doubt that the content of the copy matches the original device[5].This is done in order to keep the integrity of the actual system, while investigators can work on the copy of it.

Examination is the process of finding data aqcuired in the collection phase which might be of interest to the specific case. Forensics tools can for instance be used to categorize the data found on the mobile device, and at the same time track what the investigator is doing. Some tools also offers the capability of automatically removing known files used by the operating system, which makes the process of filtering out gigabytes of date easier for the investigator of the device.

After the phase of examination is done, analysis is done in order to see if the files contain information that might be useful in the case. This can be everything from photos, text messages or GPS localization data of the device. Time lines are often used as a help to determine what happened in what order. Log files or metadata like time stamps from files can help create a time line, which gives a better overview of what has happened. The time line created can then be expanded to contain real-life activities, which helps the investigator further understand the data. Note that when this process is done, i.e. one device has gone through the collection phase, examination phase and analysis phase, the same process starts again for a new device that might be relevant to the case. After all the devices has gone through these stages of the process, the presentation phase is started.

3. PRACTICAL APPLICATION

We wanted to find remaining traces in memory from communication data and application metadata. For this we used the phone Sony Xperia M2 with Android 5.1.1.

3.1 Laboratory Environment

The phone was factory reset to ensure a reproducible result. At this point the phone was rooted using the KingRoot application version 4.9.5. This process did not require restart of the phone, potentially leaving traces of memory from before rooting and making it ideal for memory forensics provided the phone can be unlocked regardless of root status. After testing the root, the phone was restarted to ensure a consistent state. As part of this process ADB USB debugging was enabled. A memory cleaning app 'Clean Master'(version 5.14.4) was installed which claimed to remove temporary files and free memory.

We used this tool because we wanted to check if data could survive a cleaning tool. Since we couldn't test over longer periods of time, and since the phone had very low memory**TTL memory, JP**, we used a cleaningtool to provide a more realistic environment with lot of use.

Several applications was installed for testing. The applications installed were:

Facebook

Facebook Messenger

Snapchat

WhatsApp

Jodle

Google Apps
Preinstalled on the phone, including GMail, Maps and Chrome

Since this was performed after rooting of the phone, testing of remains in memory when rooting was impossible.

Github was accessed using the Chrome browser.

Service	User Name	Password
Facebook	mis2016forensics@gmail.com	hackmyphone01
Snapchat	canuhackmyphone	hackmyphone07
WhatsApp	[Phone number]	-
Google	mis2016forensics@gmail.com	hackmyphone
Github	canuHackmyphone	hackmyphone06

Table 1: Credentials used

Approach for each application

3.2 Process

This section outlines the steps used for preparing the services and applications for memory forensics.

3.2.1 *Creation of credentials*

We created user accounts corresponding to individual applications. A full list is below.

The credentials for whatsapp has been removed from the list, however a normal phone number was used, together with a password.

After creation of the accounts in the applications, the account was used to authenticate with the applications.

3.2.2 *Creating searchable data*

In each of the applications, a normal session was fabricated with easily searchable data. The types of data used for each application is specified below:

Facebook
Information about user and friends.

Facebook messenger
Unique random strings and images from camera taken in application.

Snapchat
Images from camera taken in application.

WhatsApp
Unique strings and images from camera taken in application.

GMail
Unique random strings and images from camera taken in application.

Google Chrome
Credentials to GitHub. Web history.

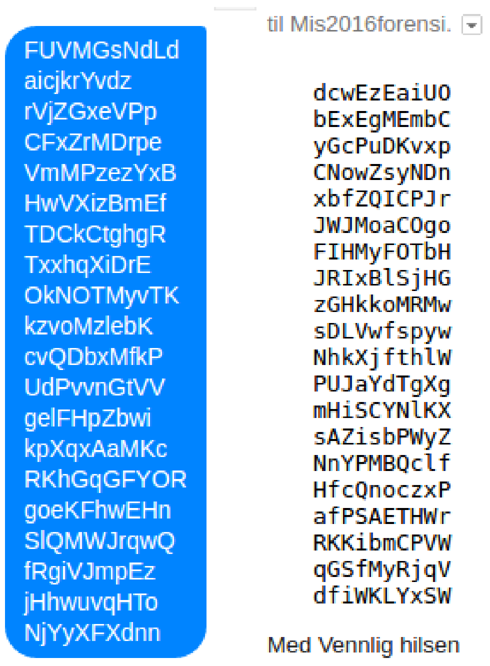
Google Maps
GPS data, locations and route between locations.

In addition to the data listed, credentials as stored by the application is present.

3.2.3 *Extracting memory*

To extract the memory AMExtract was chosen. It did not have a profile for the phone, so a profile for extraction method, size of and other properties was created and tested.

The tool was then compiled using the ndk-build tool. As part of the linux kernel headers has been modified, one of the types had to be redefined using an older header file.



FUVMGsNdLd
aicjkrYvdz
rVjZGxeVPp
CFxZrMDrpe
VmMPzezYxB
HwVXizBmEf
TDCkCtghgR
TxxhqXiDrE
OkNOTMyvTK
kzvoMzlebK
cvQDbxMfkP
UdPvvnGtVV
gelFHpZbwi
kpXqxAaMKc
RKhGqGFYOR
goeKFhwEHn
SlQMWJrqwQ
fRgiVJmpEz
jHhwuvqHTo
NjYyXFXdnn

(a) Messenger

til Mis2016forensi.

dcwEzEaiUO
bExEgMEmbC
yGcPuDKvxp
CNowZsyNDn
xbfZQICPJr
JWJMoaCOgo
FIHMyFOTbH
JRIxBlSjHG
zGHkkoMRMw
sDLVwfspyw
NhkXjfthlW
PUJaYdTgXg
mHiSCYNlKX
sAZisbPWyZ
NnYPMBQclf
HfcQnoczxP
afPSAETHWr
RKKibmCPVW
qGSfMyRjqV
dfiWKLYxSW

Med Vennlig hilsen

(b) GMail

Figure 2: Example of string used

3.2.4 *Searching memory*

The extracted memory was searched using the credentials from table1 and strings which was previously entered. For this purpose the 'strings' tool was used to extract continuous regions of ASCII text and 'grep' was used to search in this text. This has the disadvantage of only finding continuous buffers using simple storage mechanisms, but is quick to execute.

3.2.5 *Carving memory*

To find other types of resources a simple file carving was attempted using 'scalpel' with matches for the following file types:

- PNG
- JPG
- TIFF

3.3 What were we looking for

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

4. RESULTS

In this section we explore our findings, what we expect to find, what we did find and other information we found.

4.1 Results from generic search terms

When searching for generic information and structures, some structured information was found. This included JSON when searching for '{' and http requests when searching for 'http://'.

A search for coordinates matching the following RegEx '60[0-9]*,10[0-9]*' was performed, which corresponds to the area the phone was located. Matches for a location in Google Maps was returned with the street address. This was from a search performed in the Google Maps application during the preparation. Searching for part of the address string revealed the address at different stages of completion.

4.2 Results from specific key phrases

The passwords and usernames from table 1 was used to detect the presence of credentials. Results are in the following table **??**

For finding information related to Facebook, a case-insensitive search for the string Facebook was performed. This revealed information about friends of the logged in user, metadata of other users profile pictures was found from the Facebook application along with chat messages from Facebook Messenger. Information regarding whether the contact was muted, timestamp of last delivery and timestamp of last confirmed received message by the other participant. Searching for the name of the participants could identify the chat messages, since all the messages had metadata containing name of the other party.

For GMail and e-mail messages, the content and metadata was often found in the memory dump. As such the messages could be found by searching for the e-mail address.

4.3 Carved images

When carving several images were recovered. Most of these where part of stock Android OS or part of an applications assets, however when matching JPEG images, several of the pictures taken by the camera was discovered. These had no metadata. A overview of what was returned from each application can be found below:

Camera
: The full scale image was only available for a short duration after use before being stored on the device, however a small scale preview from the select image screen or camera was still in memory.

WhatsApp
: No images could be recovered, however a reference to a file name of the image could be found in the chat conversation from memory suggesting it is stored on the device.

Snapchat
: The full scale image was only available for a short duration after use. After some time part of the memory region containing the image was overwritten by a sequence of 0 bits before finally being removed entirely, however a small scale preview from the select image screen or camera was still in memory.

5. DISCUSSION

Rooted without rebooting - In our experiment we tried two different rooting methods. One with help of a computer and a program called Odin. This method forced us to reboot the device and lose the memory. The second method was using an application called KingRoot. This application gave us root action over the phone shortly after installation. It did not need any external help to get root access.

Unallocated data, and partially overwritten

Limitations

Full access to the phone

Phone memory size

Timeframe âĂŞ The project had a stri

1GB memory, 800MB dump âĂŞ The mobile had 1 GB memory, but we were able to only dump 800 MB of it. This means that some of the memory are either locked or unavable for dumping due to the operation system or other locked memory information.

Writeblock âĂŞ Was used to prevent overwriting important data from the RAM. If there were any possibility for overwriting the data, that could compremise the essensial information in the case and could easly be dismissed in a court of law.

The collection phase consists of retrieving data from the device (or devices) that are brought in by the investigators[?]. Brezinski and Killalea[?] suggests evidence collection to be done in an order of volatility. In the case of a mobile device, the RAM should therefore be collected before the primary storage. This is done by copying data over to another system from the mobile device. While a regular copy may cause the system to overwrite existing data, write-blockers are used in order to deny the system anything other than read access, which makes a copy doable without interfering with the existing content on the device. A hash of both the copied device and the copy itself is then created, and compared so that it is no doubt that the content of the copy matches the original device[?].This is done in order to keep the integrity of the actual system, while investigators can work on the copy of it

Kingroot âĂŞ What does it do? - KingRoot is a tool used to gain root control on a mobilephone with a system exploit. KingRoot APK will not trip the Samsung KNOX and have the ability to close Sony_RIC perfectly. Rumours says that KingRoot may interfere with the android OS, but there were no changes to be found except root access.

No real test data âĂŞ Throughout the testing of the data retrieved from dumping, all the information that was gathered had been created by us.

Forensic Process

In this project the rooting of the phone and dumping the memory was done as an real forensic case would have done. First of all the phone was not turned off while it was rooted in case there was any

sensetive data in the memory as real case would have done.

6. CONCLUSION

Easy to extract memory âĂŞ To extract the memory from the phone a tool called AMExtract was used. It did not have a profile for the mobilephone that was used in this project. Therefore one was created and compiled using a ndk-build tool. As soon as the profile was created the memory dumping was a success.

Both of generic and specific searches - During our experimentation fase we used two different searches through the memory in order to gather information. One of the searches was a generic type of search. This means that a search-word with no other specification than HTTP was used. The second type of seacrh was a more specific one. The search word facebook was used and found conversations with both the message and the timestamps of the messages.

Cleaning tool and closing applications had little effect âĂŞ With the use of a cleaning tool and closing the different apps, it did not have any effect on the amount nor what type of information that was dumped. Regardless the result was the same.

Found data that was never saved (Snapchat) âĂŞ After going through the data that was dumped from the mobile-phone, a full scale image was found shortly after using the application Snapchat. Afterwards the memory containing the image was overwritten, nonetheless a small scale preview of the image was still in memory.

7. ADDITIONAL AUTHORS

8. REFERENCES

[1] IDC. Smartphone OS Market Share, 2016 Q2; 2016. http://www.idc.com/prodserv/smartphone-os-market-share.jsp.

[2] Carlstrom B, Ghuloum A, Rogers I. The ART runtime. Google I/O; 2014. .

[3] Anikeev M, Freiling FC, Götzfried J, Müller T. Secure garbage collection: Preventing malicious data harvesting from deallocated Java objects inside the Dalvik VM. Journal of Information Security and Applications. 2015;22:81–86.

[4] Google. Overview of Android Memory Management;. (Accessed on 4/11/2016.). https://developer.android.com/topic/performance/memory-overview.html.

[5] Årnes A, Flaglien AO, Sunde IM, Dilijonaite A, Hamm J, Sandvik JP, et al. Digital Forensics. 1st ed. John Wiley and Sons, Limited; 2017.

[6] Brezinski D, Killalea T. Guidelines for Evidence Collection and Archiving. RFC Editor; 2002. 3227. Available from: http://www.rfc-editor.org/rfc/rfc3227.txt.

[7] Androulidakis II. In: Mobile Phone Forensics. Cham: Springer International Publishing; 2016. p. 87–109. Available from: http://dx.doi.org/10.1007/978-3-319-29742-2_6.