IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Revue d'Intelligence Artificielle
Vol. 37, No. 2, April, 2023, pp. 475-481
Journal homepage: http://iieta.org/journals/ria

Design of an Efficient Smart Phone Data Extraction Tool Using Aho-Corasick Algorithm

Annies Mary Jeyaseeli, Shanthi Chandrabose*

Department of Computer Science, Vels Institute of Science, Technology, and Advanced Studies, Chennai 600117, India

Corresponding Author Email: shanc08071978@gmail.com

https://doi.org/10.18280/ria.370224

Received: 9 February 2023
Accepted: 10 March 2023

Keywords:
data extraction, Aho-Corasick algorithm, smart phones

ABSTRACT

Data recovery from Android mobile devices has become an increasingly important area of research in recent years. With the rise of mobile technology, accidental deletion or erasure of data is becoming a common problem among users. Many methods have been developed to recover data from such devices, but most of them are either time-consuming or require specialized technical skills. In this paper, we present an approach using the Aho-Corasick algorithm, which has been shown to be highly effective in locating strings in large datasets. Our proposed method aims to reduce the computational time required for data recovery, making it more accessible to a wider range of users. In this paper, we develop a Aho-Corasick algorithm is the most effective method for recovering data from Android mobile devices after it has been erased inadvertently. The Aho-Corasick algorithm is one approach that can be utilised in the process of locating strings. It is a piece of software that scans a given document in search of occurrences of strings that have been selected from a dictionary. It carries out a simultaneous match on each of the strings at the same time. The second part of our method involves determining, with the assistance of the Aho-Corasick algorithm, whether files have been removed from the system. To accomplish this, the file types are compared to those of files that are already established as being trustworthy and the proposed method achieves a reduced computational time than other tools.

1. INTRODUCTION

Mobile phones have become an indispensable part of modern society, providing not only communication but also a wide range of services and functions, including internet browsing, email access, social media, and more. In recent years, the development of more complex telecommunications infrastructures has further increased the importance of mobile phones, particularly those with their own operating systems for intelligence. Android, one such operating system, has gained a significant market share, with millions of users worldwide. As such, the need for effective methods of recovering data from Android mobile devices, particularly after inadvertent erasure, has become increasingly critical. Mobile phones, particularly those with their own operating systems for intelligence, have become increasingly significant to modern society as a whole as a result of the development of more complex telecommunications infrastructures. This is especially true for mobile phones with advanced intelligence [1]. The creation of the term smart phone, which is used to describe a type of mobile phone like a personal computer, with independence from the operating system, that can install software, such as games from third-party service providers, by the users, that can expand mobile phone functions through such programmes constantly, and that can realise the wireless network through the mobile communication network [2].

The steady increase in the number of people using smart phones can be attributed, at least in part, to the numerous benefits offered by smart mobile phones as well as the marketing efforts of companies that specialise in the development of such devices. This is why the number of people using smart phones is expected to continue to rise soon. The ownership of a smart phone is associated with a variety of benefits; on the other hand, it is also associated with a variety of drawbacks [3]. When a traditional cell phone is analysed, the results are typically the same well-known collection of data; however, when a smart phone is analysed, the results are typically a wealth of information since each app stores application-related data [4], such as call history, text messages, contacts, and images.

There has been an increase in criminal activity that makes use of mobile phones and various networks, and this trend is expected to continue. Because of the significant damage that may be caused to society by mobile phones and the crimes that can be carried out using them, the police have focused a lot of their attention on both of these aspects of the problem. Mobile phone evidence has become more significant during treatment for many instances. This is since mobile phones often preserve essential information that can provide clues and a basis to clarify the facts of the case [5]. Mobile phones are getting more and more commonplace in today culture.

At the present time, the criminal use of a mobile phone can be separated into three distinct categories: first, mobile phones are used as communication tools during the commission of criminal acts; second, mobile phones serve as a kind of storage medium for evidence of criminal acts; and third, mobile phones are used as an implementation tool for criminal activities such as SMS fraud, SMS harassment, and the communication of viruses. During an inquiry, analysts may find the information gathered from smart phones to be highly helpful [6].

As a direct consequence of this, the primary emphasis of study in the field of mobile phone forensics has shifted to include evidence of criminal activity. The Android system, which was developed by Google, is the first mobile software for the mobile terminal that is entirely open source. It is based on the Linux kernel, which serves as the foundation for the open-source operating system for mobile phones. Since it was originally presented in 2008, the new smart mobile phone system has shown a continuous pace of improvement ever since it was initially released. For this reason, the analysis of Android forensics will take place very soon [7, 8].

The ability to retrieve lost data is a vital component in mobile forensics since it permits us to look through erased data in search of potential evidence. The Yaffs2 file format that is stored in memory may be analysed, which provides us with the capability to search for and restore data that had been destroyed in the past. We need to ensure that the original file is secure before we can proceed with making a copy of the content that is saved in the memory of the mobile phone. The rest of the work is organized is as follows: the related work is discussed in section 2. Before explaining the details of our proposed method, we will provide a background on data recovery techniques and the Aho-Corasick algorithm in section 3. We will then describe the implementation and evaluation of our proposed method in section 4. Finally, we will conclude the paper with a discussion of the results and future directions for research in this area.

2. RELATED WORKS

The process of forensic data recovery on the cloud presents its own unique set of problems that must be overcome. As a result, the use of trusted platform modules (TPM) in hypervisors, the implementation of multi-factor authentication as potential solutions to the difficulties that are associated with cloud forensics. Some of these problems include having restricted access of validating its integrity, presenting evidence, or making decisions on data maintained in separate locations [9].

According to the findings of an evaluation that was carried out as part of a cloud-related forensic investigation using both of these products, volatile and non-volatile data can be successfully retrieved from the cloud by using Guidance Encase and the Access Data forensic toolkit. This information was gleaned from the cloud. This prepares the way, from a forensics point of view, for the development of next-generation data gathering strategies that are based in the cloud and are both dependable and safe [10]. These strategies will be able to collect data in a more efficient and reliable manner. It has been proposed that a methodical approach to the collection of evidence data be utilised in cloud-based forensic investigations to guarantee the investigation credibility [11].

The purpose of this study is to give a comprehensive reference for future research by analysing the work that has been done over the course of seven years on the forensic examination of many smartphone platforms, data gathering tactics, and information recovery procedures. Before the development of forensic technology, the traditional approach to memory gathering placed a significant amount of emphasis on the collection of physical evidence [12]. This occurred in the past, before the time when forensic technology was readily available. To complete this job successfully, you will need to take the memory chip out of its socket on the motherboard usually. During the cleaning process, it is possible that crucial pieces of evidence will be lost or thrown away due to the operation of these methods [13].

To carry out a forensic investigation that can be relied upon, it is essential to take measures to protect and retrieve the volatile data that is stored within the memory of the mobile device. These measures must be taken to conduct an investigation that can be relied upon. Because of this, an approach to backup and acquisition has been suggested that is suitable for use with iPhones in addition to Android Mobile phones and Windows Mobile phones. Investigation was done into a few potential methods for regaining access to and interpreting data that has been removed from a smart phone [14].

The findings indicate that there is no one method that can obtain from the device all the data that is necessary for the forensic inquiry to proceed. However, even though a variety of smartphone forensic tools have been developed because of ongoing studies into mobile device forensics, these studies do not verify the integrity of the data, which is essential for digital forensic investigation. In other words, even though these studies have resulted in the development of a variety of smartphone forensic tools, the research has not been successful [15].

Researchers analysed the factors that affect data integrity during acquisition that are related to Android device recovery mode variables by looking at Android device acquisition while the devices were in recovery mode. This allowed the researchers to analyse the factors that affect data integrity during acquisition. An Android data gathering tool was built as a direct consequence of this to assure the veracity of the information that was acquired. Everyone was made aware of the fact that there are now no predetermined processes in place for the acquisition of evidence from smartphones [16].

When it comes to the collection of digital evidence, forensic investigators make use of methods that have been shown to be reliable via previous research and testing. In addition to this, it was suggested that various versions of the software that is installed on cellphones may make it possible for investigators to access variable volumes of information that is relevant to the case. On the other hand, there was no attempt made to retrieve evidence from a formatted Android smartphone, from which it was reported that all the data and programmes had been removed. It was claimed that this piece of evidence could not be reclaimed in any way [17].

This study looked at five different forensic scenarios and offered a method to obtain data from smartphone, regardless of the physical design of the device. The research was carried out by the authors [18]. The purpose of this investigation was to discover an answer to the problems that were discussed earlier. Investigations into mobile devices based on the operating systems Symbian and Windows substantiated these architecture-related challenges, which suggests that forensic investigation methods for smartphone mobile devices will perpetually encounter challenges due to the ever-evolving nature of technology. Both issues occurred even though these files had not been deleted. However, the results of the inquiry showed that not all instruments are made in the same way and are therefore not comparable.

The evidence that was gathered and analysed from a Nexus 4 phone revealed a vulnerability that gave hackers full access to the phone even after the bootloader had been unlocked, which in most cases deleted all user data. The vulnerability was discovered through the discovery of a vulnerability that gave hackers full access to the phone. Because of this vulnerability, the hackers were able to access the phone even after the bootloader had been successfully unlocked. It does not yet have a mechanism for forensic examination, the difficulties of smartphone forensics are continuing to get more difficult [19, 20].

3. THE PROPOSED METHOD

In this section, we create our technique for retrieving information from a database that has been deleted in the past. Before making any attempt to retrieve data from a mobile device, it is strongly recommended that a backup of all the information that has been saved on the device be produced. The original data does not need to be altered in any way for the data recovery procedure to be successful because it can be performed entirely on the image copy. The journal and the inodes together form the backbone of the system, which is supported by the file system, which is where all the system folders and files are stored. The file system is supported by the journal.

Read all the entries that are in the file system journal and inodes, extract all the metadata, and then save it (file type, file name, size, data, file addresses, created date, last modified date, flag whether the file is deleted, etc.).

Following the processing of each file, a CRC-32 checksum is subsequently produced for that file. To certify that the recovered file is in usable condition, this checksum is compared with the checksum that is computed after the damaged files have been rebuilt.

To determine which files and directories were removed, read the properties of those that were deleted from the journal and inodes of the file system. This will allow you to identify the files and directories that were removed.

Give the user your best guess as to what happened to the files after they were deleted and explain why you reached that conclusion. It possible that a deleted file fingerprint and data type won't be able to be recovered at all. When this occurrence takes place, our system immediately moves on to the subsequent phase. During this phase, it initiates the process of parsing the second copy of the metadata to recover the information that was omitted before.

This method provides a user-friendly interface for entering search terms, determines whether the entered terms match the metadata for the deleted files, the Aho-Corasick technique is then used to decide whether or not the entered terms match the metadata for the deleted files. This occurs after our method determines whether the entered terms match the metadata for the deleted files. Following the conclusion of this step, the user will be shown with a list of the files that have recently been removed so that they may confirm that the procedure was successful.

Rebuilding and recovering the files that correspond to the user input and the metadata will result from this action. In order to accomplish this goal, first create new files that have the same size as the ones that were destroyed, and then copy the contents of the deleted files into the new files.

At this stage, we do a comparison between the CRC-32 value of the restored file and the value of the original file to determine whether the two values are same. If the two numbers match up, then the file in question does not have any tainted information and can be viewed in the usual manner as in Figure 1.

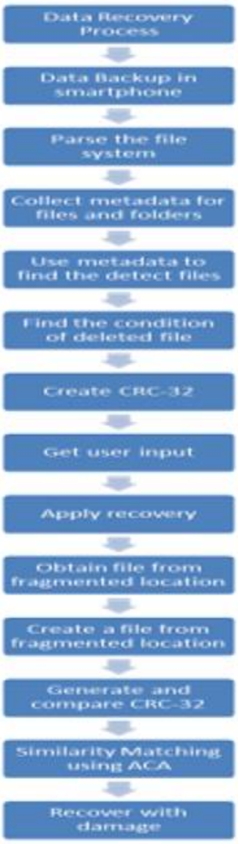

Data Recovery Process
Data Backup in smartphone
Parse the file system
Collect metadata for files and folders
Use metadata to find the detect files
Find the condition of deleted file
Create CRC-32
Get user input
Apply recovery
Obtain file from fragmented location
Create a file from fragmented location
Generate and compare CRC-32
Similarity Matching using ACA
Recover with damage

Figure 1. Data recover process

3.1 Aho-Corasick algorithm

The fact that the computation time for the AC algorithm is linear with the length of the input stream and that it does not depend on the signature strings that lends credence to the idea that it could be a viable contender for finding a solution to the string-matching problem.

The ability of the AC algorithm to find a match for a string in a time that is proportional to the stream length is one of the most appealing features of the approach. This ability makes the method one of the most desirable candidates for use. This is done to maintain the same level of consistency in the processing time (DFA). The fact that the AC technique needs a sizeable amount of memory to hold the NFA transition rules is the most important drawback associated with using this approach.

Take into consideration a signature set in which there are n total strings, L is the average length of the strings, and L is the total number of strings. On the AC state graph, the variable n L, which represents the maximum number of states that can be achieved, indicates the maximum number of states that can be reached. There are 256 different routes that can be used to get from one state to another across the country.

It is possible for the state graph to contain a maximum of 25.6 million transition edges if the value of n is set to 5,000 and L is set to 20. Because of the little size of embedded devices, it is safe to infer that it would be impractical to create such a comprehensive transition rule table for these kinds of devices.

A tree-like form has been created from the NFA state graph, which has been compacted. When NFA is employed, there is a risk that the automaton will go through a significant number of state changes for each character that is entered. This is something that need to be avoided wherever possible. to properly process each character that is entered into a memory based NFA implementation, a variety of different table look-up operations will need to be carried out.

We see a particularly difficult aspect of the NFA in terms of the string-matching problem. Comparable to the various pattern substrings that comprise the signature set, the nodes of the state graph are what make up the graph itself.

This is the case even if there are multiple states in the state graph. It makes no difference whether the state graph is dynamic; this is always the case. We illustrate how this trait may be exploited to boost processing performance in a pipelined design while simultaneously reducing the size of the state graph. This can be accomplished by leveraging the property inherent pipelined nature.

In theory, the pipeline will cycle through a single state for each character that is present in the input stream. This is the case even if in practise it will cycle through multiple states. If we are going to process one character each cycle, then each table look-up operation needs to be finished within of only one cycle. The process of hashing data is the procedure that is used most frequently while building the framework for the execution of huge LUTs.

When dynamic signature sets are utilised, collisions are an unavoidable and unavoidable outcome. When collisions occur, the number of memory visits that need to be carried out to complete a database lookup can skyrocket, depending on the severity of the collision. The throughput of the system will be negatively affected because this is the case. The underlying AC approach uses a DFA model to represent the system in its analysis. If we refer to the AC automaton for a signature set as

$M()=(Q, q0, \delta, F)$,

where,

q0-initial state,

Q-states,

δ-transition function, and

$F\subseteq Q$-output state set represents the AC automaton for this signature set.

When we discuss the transition function, we are talking to any mapping that moves from one $Q^{\times}\rightarrow Q$. The transition function is synonymous with Q mapping. One way to express the automaton M is by writing it down as the graph.

$G=(Q, E)$,

where,

Q-nodes and

E-edges.

$E=\{(u, x, v)|u\in Q\wedge x\in \wedge v=\delta(u, x)\}$.

where,

x-input symbol,

u-current state,

v-future state,

E-expression

The position 0 is where the system operation is initiated for the first time. It performs an analysis of the currently active state in conjunction with the character that was input during each cycle and then modifies the state in accordance with the findings. When considered in this setting, the terms edge and transition, as well as state and node, are interchangeable with one another.

A representation of the AC state graph for the string combination apple, past is presented in Figure 1. The label for each node u in the state graph is a string value that begins with the prefix U, which is short for some string Y. This prefix identifies the string Y. The empty string stands in for the starting state, which is indicated by the q0 value. This value is denoted by the string. To make the process simpler and clearer to comprehend, reverse transitions have had their input symbols masked, and their values have been set to q0.

The character that appears at the end of the string is the one that is utilised as the input symbol for a transition, and the node that the transition is bound to is responsible for representing this character. By analysing the length of the text that is associated with a certain node in the state graph, it is possible to ascertain the level of a node in the state graph.

Since q0 level number is zero and its state value is (a string with no characters), we can deduce that its level number is 0. Let call the collection of nodes that are situated on the i-th level of the state graph the Ni group for now. It is hypothesised that the edge formed by the coordinates (u, x, v) is forward if u and v are both positive integers, and if x does not have a negative value.

The remaining edges, which are called cross-edges and are indicated by dashed lines, are not connected to any of the other edges in the diagram. There are two distinct types of cross-edges: those that are unsuccessful and those that are successful. Both types are possible. In the scenario in which v=q0, the cross edge given by e=(u, x, v) will be successful; however, in any other scenario, e will fail. Take into mind the remark that comes next: According to this definition, the set of forward edges is referred to as Ef, the set of nonfailure cross edges is referred to as Ecn, and the set of failure edges is referred to as Ecf. E is proportional to Ef+Ecn+Ecf.

After going over the first two categories of edges that can be found, this article moves on to describe the third category of edges that can be found. Let imagine that u and v are two different states in Q and that u and v are the strings that correspond to those states. This will make it easier to understand what is going on. The following is an itemised list of characteristics of the leading edge, which may be modelled using the equation e=(u, x, v).

Both U and V are frequently seen functioning as prefixes for particular string.

The definitions and explanations of the three categories of edges that were described earlier can now be found in this section. Imagine that u and v are two separate states in Q, and that U and V are the strings that correspond to u and v, respectively. This would mean that u and v are both represented by the U. Edge e=(u, x, v) ∈Ef, where each of the requirements outlined below is met, then Ef can be regarded as a forward edge.

(i) U, and V are all examples of prefixes that can be tacked onto the start of a string (or several strings) in Γ.

(ii) v∈Ni+1 and u∈Ni for i≥0.

(iii) U·x=V.

The E_{cn} can be considered a nonfailing cross edge: Edge e=(u, x, v) ECN denotes a cross edge.

(i) v∈Nj and u∈Ni for i≥j>0.

(ii) According to the findings presented in, there is no such thing as a single forward edge with the equation e=(u, x, w) E∈f that exists for some w∈Q.

(iii) There exists a string y, which can be an empty string, in such a way that y is a suffix of U and y·x=V. This is possible because y is a string that can be empty and there is a string at this location.

(iv) There is no other suffix z of U that has a length that is longer than y and whose suffix z is shorter than x for each and every node v of Q. This is because there is no other suffix z’∈U.

The alphabet set is divisible into three distinct subsets, each of which can be differentiated from the others based on a single dividing point.

f (u)={x|x∈∧ ∃e=(u, x, v) ∈Ef},
cn(u)={x|x∈∧ ∃e=(u, x, v) ∈Ecn}, and
c f (u)=-f (u)-cn(u).

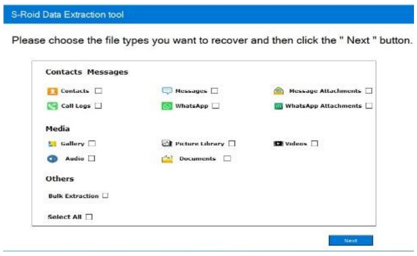
The overall amount of characters in the set, as well as the number of strings that are included in the set and their combined length. The value that is returned by the *M*() function can be somewhere between 0.7 and 0.8, and the number of edges can be anywhere from 256×β×n×L. It is likely that the number of edges in the AC state graph could approach 20 million if N is set to 5,000 and L is set to 20.

4. RESULTS AND DISCUSSIONS

During our tests, we took use of a Samsung Galaxy S2 i9100, which is a model that is currently quite popular. Because our investigation required a substantial quantity of storage on the device internal components, we chose not to use a memory card of the appropriate size. After that, we used the phone to take a few images before resetting it to the settings it had when it was first manufactured (see Figure 2). Resetting the phone to its factory settings had two purposes: the first was to remove any data that had been pre-installed by the manufacturer, and the second was to determine how successfully forensics tools could recover anything that had been removed from the phone.

The Enron dataset, which was already on the phone when the studies began and was relatively comparable to data acquired for the goal of detecting fraud, was not available to researchers at that time. The data collection in question is one that may be utilised effectively for the research at hand.

The smartphone has a random-access memory of 1 GB and a storage capacity of 16 GB, and it is powered by the Android OS version 2.3.4 Gingerbread (Android OS, Ice Cream Sandwich version 4.0.3). the interface used to extract the files is given in Figure 3



S-Roid Data Extraction tool

Please choose the file types you want to recover and then click the " Next " button.

Contacts Messages

Contacts □ Messages □ Message Attachments □
Call Logs □ WhatsApp □ WhatsApp Attachments □

Media

Gallery □ Picture Library □ Videos □
Audio □ Documents □

Others

Bulk Extraction □

Select All □

Figure 2. Interface of proposed s-roid extraction tool

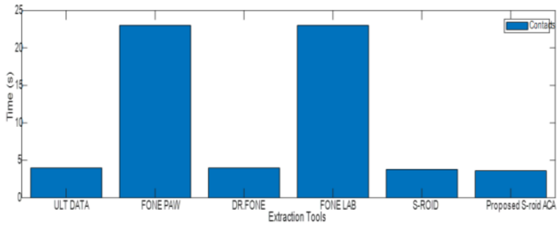


Figure 3. Computation time required to extract contacts with a data volume of 210.63

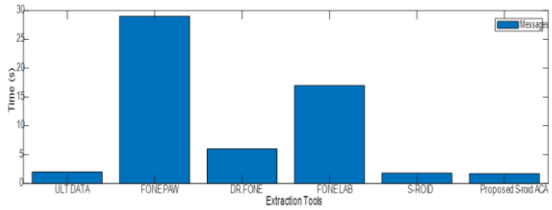


Figure 4. Computation time required to extract messages with a data volume of 138.59

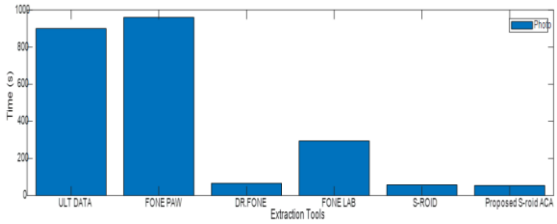


Figure 5. Computation time required to extract photos with a data volume of 719490

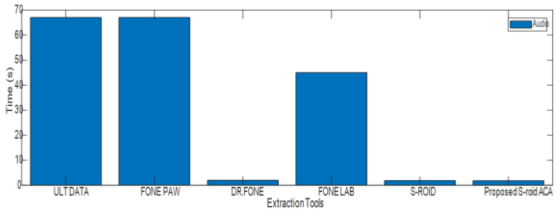


Figure 6. Computation time required to extract audio with a data volume of 105350

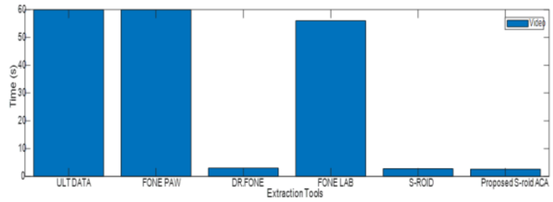


Figure 7. Computation time required to extract video with a data volume of 183430

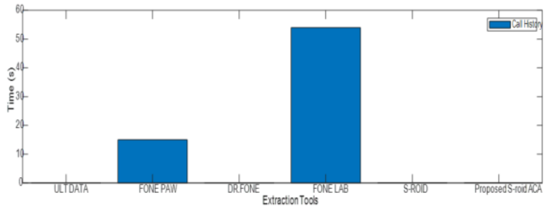


Figure 8. Computation time required to extract Call History with a data volume of 1620

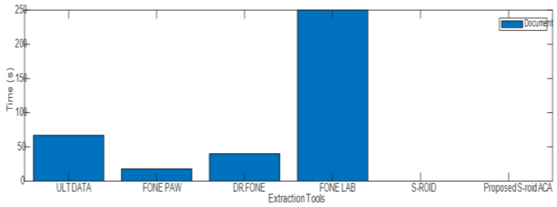


Figure 9. Computation time required to extract documents with a data volume of 37190

Except for graphics and other thumbnails, for which the two programmes have recovered approximately the same number of files, the research shows that dd File has fewer entries that are equivalent to one another. This leads one to believe that image files, such as thumbnails and graphics, can be restored more easily by using DD files. The portion of the file that is referred to as slack or free space is where the FTK File and the DD File differed greatly from one another. Both files had values that were similar to one another under the headings for the File Status and the File Category sections of the files.

The FTK file provides more recovery files than the DD file does in the case of evidence that may be recovered from slack areas. This is the case even though the DD file was the original. This conclusion is arrived at because of looking at the data. To put it another way, unused space may hold data that is redundant or no longer useful, as well as files that have been partially deleted or file fragments.

The problem is that it likely that recovering from it won't be worth anything if there isn't any evidence that worth rescuing from the gaps in the data. FTK File 8, which is where we find the Slack and Free Spaces, was not successful in recovering any of the files that were being searched for. The tests conducted by the CFTT Program discovered vulnerabilities in the FTK File hard disc file preparation technique that was implemented on the Windows XP operating system.

The fact that the files in Figures 4-9 are so large provides support to the theory that Foremost retrieved content that Access Data FTK did not discover because to slack or free space. This information may comprise files that have been erased, file pieces that have been deleted, and potentially even files that have been buried.

Figures 4-9 show that the leading forensic tool was able to recover more data from the Backtrack DD image than it was able to recover from the FTK image. This is seen by the comparison of the two images. This image also displays the file sizes together with the total number of files that were retrieved from the lost device. When it comes to analysing the data files that are created by Backtrack dd Image, this also implies that the best forensic tool, Backtrack dd Image, works better than FTK Image. According to the data that was gathered, the types of documents that were most frequently recovered by the Foremost forensic tool were zip, jpeg, mp4, and png files, followed by pdf files as the next most frequently recovered file type.

5. CONCLUSIONS

The rise in the use of smartphones for monetary transactions and other forms of social connection has been matched by an increase in the number of instances of cybercrime committed using mobile phones. The complexity of these devices, as well as the wide variety of software that they housed, presented forensic investigators with a new set of obstacles that they had not previously come across. They were effective in obtaining or intercepting data from passwords, screenshots recorded by programs, images, audio, videos, messages shared, and profile images. The AC algorithm needs a significant amount of memory in order to keep track of the transition rules of the deterministic finite automaton that serves as its foundation. This automaton is employed in the construction of the algorithm. One of the key contributions of this study is the application of the Aho-Corasick algorithm to mobile data recovery. While this algorithm has been previously used in other contexts, to our knowledge, this is the first study to apply it to mobile data recovery. Our results demonstrate that the algorithm is well-suited to this task and can significantly improve the efficiency and accuracy of data recovery.

Furthermore, this study highlights the importance of file type identification in the data recovery process. By comparing file types to a trusted set of files, we were able to accurately identify removed files and reduce the risk of false positive

REFERENCES

[1] Dorai, G., Houshmand, S., Aggarwal, S. (2020). Data extraction and forensic analysis for smartphone paired wearables and IoT devices. In HICSS, pp. 1-10. https://doi.org/10.24251/HICSS.2020.172

[2] Fukami, A., Stoykova, R., Geradts, Z. (2021). A new model for forensic data extraction from encrypted mobile devices. Forensic Science International: Digital Investigation, 38: 301169. https://doi.org/10.1016/j.fsidi.2021.301169

[3] Shankar, V.G., Devi, B., Srivastava, S. (2019). DataSpeak: Data extraction, aggregation, and classification using big data novel algorithm. In Computing, Communication and Signal Processing: Proceedings of ICCASP 2018, Springer Singapore, pp. 143-155. https://doi.org/10.1007/978-981-13-1513-8_16

[4] Aggarwal, S., Dorai, G., Karabiyik, U., Mukherjee, T., Guerra, N., Hernandez, M., Parsons, J., Rathi, K., Chi, H., Aderibigbe, T., Wilson, R. (2019). A targeted data extraction system for mobile devices. In Advances in Digital Forensics XV: 15th IFIP WG 11.9 International Conference, Orlando, FL, USA, pp. 73-100. Springer International Publishing. https://doi.org/10.1007/978-3-030-28752-8_5

[5] Laranjo, L., Ding, D., Heleno, B., Kocaballi, B., Quiroz, J.C., Tong, H.L., Chahwan, B., Neves, A.L., Gabarron, E., Dao, K.P., Rodrigues, D., Neves, G.C., Antunes, M.L., Coiera, E., Bates, D.W. (2021). Do smartphone applications and activity trackers increase physical activity in adults? Systematic review, meta-analysis and metaregression. British journal of sports medicine, 55(8): 422-432. https://doi.org/10.1136/bjsports-2020-102892

[6] Husnjak, S., Forenbacher, I., Peraković, D., Cvitić, I. (2022). UAV forensics: DJI mavic air noninvasive data extraction and analysis. In 5th EAI International Conference on Management of Manufacturing Systems, pp. 115-127. https://doi.org/10.1007/978-3-030-67241-6_10

[7] Thornton, G., Zadeh, P.B. (2022). An investigation into Unmanned Aerial System (UAS) forensics: Data extraction & analysis. Forensic Science International: Digital Investigation, 41: 301379. https://doi.org/10.1016/j.fsidi.2022.301379

[8] Liao, L., Li, J., Lu, C. (2022). Data extraction method for industrial data matrix codes based on local adjacent modules structure. Applied Sciences, 12(5): 2291. https://doi.org/10.3390/app12052291

[9] Scrivens, N., Lin, X. (2017). Android digital forensics: Data, extraction and analysis. In Proceedings of the ACM Turing 50th Celebration Conference-China, China, pp. 1-10. https://doi.org/10.1145/3063955.3063981

[10] Turow, J., Couldry, N. (2018). Media as data extraction: Towards a new map of a transformed communications

field. Journal of Communication, 68(2): 415-423. https://doi.org/10.1093/joc/jqx011

[11] Barreneche, C., Wilken, R. (2015). Platform specificity and the politics of location data extraction. European Journal of Cultural Studies, 18(4-5): 497-513. https://doi.org/10.1177/1367549415577386

[12] Gupta, A., Anand Shankar, S., Manjunath, C. (2017). A comparative study on data extraction and its processes. International Journal of Applied Engineering Research, 12(18): 7194-7201.

[13] Spolaor, R., Dal Santo, E., Conti, M. (2017). Delta: Data extraction and logging tool for android. IEEE Transactions on Mobile Computing, 17(6): 1289-1302. https://doi.org/10.1109/TMC.2017.2762692

[14] Kong, J. (2015). Data extraction on mtk-based android mobile phone forensics. Journal of Digital Forensics, Security and Law, 10(4): 3. https://doi.org/10.15394/jdfsl.2015.1209

[15] Ashawa, M., Ogwuche, I. (2017). Forensic data extraction and analysis of left artifacts on emulated android phones: a case study of instant messaging applications. Seizure, 19: 16. https://doi.org/10.22632/ccs-2017-252-67

[16] Aziz, N.A., Mokhti, F., Nozri, M.N.M. (2015). Mobile device forensics: extracting and analysing data from an android-based smartphone. In 2015 Fourth International Conference on Cyber Security, Cyber Warfare, and Digital Forensic (CyberSec), pp. 123-128. IEEE. https://doi.org/10.1109/CyberSec.2015.32

[17] Tajuddin, T.B., Abd Manaf, A. (2015). Forensic investigation and analysis on digital evidence discovery through physical acquisition on smartphone. In 2015 World Congress on Internet Security, pp. 132-138. IEEE. https://doi.org/10.1109/WorldCIS.2015.7359429

[18] Basyoni, Y., Talaat, H. (2015). A bilevel traffic data extraction procedure via cellular phone network for intercity travel. Journal of Intelligent Transportation Systems, 19(3): 289-303. https://doi.org/10.1080/15472450.2014.892380

[19] Cui, M., Wu, X., Mao, J., Wang, X., Nie, M. (2016). T2DM self-management via smartphone applications: A systematic review and meta-analysis. PloS One, 11(11): e0166718. https://doi.org/10.1371/journal.pone.0166718

[20] Firth, J., Torous, J., Nicholas, J., Carney, R., Pratap, A., Rosenbaum, S., Sarris, J. (2017). The efficacy of smartphone‐based mental health interventions for depressive symptoms: A meta‐analysis of randomized controlled trials. World Psychiatry, 16(3): 287-298. https://doi.org/10.1002/wps.20472