Email Conversations Reconstruction Based on Messages Threading for Multi-person

Xia Wang, Ming Xu, Ning Zheng, Mo Chen
Institute of Computer Application Technology,
Hangzhou Dianzi University, Hangzhou, ZheJiang, 310018, China
annewang@stu.hdu.edu.cn, hz_xuming@126.com , nzheng@hdu.edu.cn, mchen@stu.hdu.edu.cn

Abstract

The email conversations reconstruction and analysis is a key task of the digital forensic, however, most forensic tools only parse email data without analysis. This paper presents a novel thread-based conversations reconstruction mechanism that provides an effective analysis and statistics of the email flows for multi-person. The mechanism includes a data extraction rule for email headers extraction and redundant emails filtering, a messages mapping algorithm to keep the messages without Message-ID in correct Parent/Child relationship, and a topic-based heuristic to merge or decompose threads to conversations. The experiment results show that our mechanism exhibits high performance on conversation detection, tracking and Parent/Child relationship keeping, which suggest that the mechanism is a feasible strategy for email conversations reconstruction in digital forensic.

1. Introduction

Due to the convenience and cost effectiveness, email is being utilized widely for communication, either legitimately or illegitimately, and becoming an important medium of digital evidence. As a major task of computer forensic, email forensic include two steps after collecting--parsing and analysis, and the latter is more neglected than the former in most forensic tools. To analysis, two perspectives are focused on: 1) the text mining and 2) the link analysis for the social network analysis. Both of them need take emails as inputs, and a clustered email corpus by conversations will be a great benefit to the analysis, especially to the statistics of the reply rate and response time of a user.

An email thread is considered as a tree, where nodes of the tree represent emails and a directed arc going from one node to another represent the reply or forwarding relation of them. A conversation is based on an email thread but clustered by topic. Due to the Reference header of a message being omitted by some email clients or users' special operation, a conversation may span several threads. And a thread may be distributed over several conversations, because some people use the "Reply To" button instead of "New Mail" button to start a new conversation. Therefore, the threads have to be merged or decomposed to construct the conversations.

Many email clients group emails to conversations for users. Zawinski [1] detailed a message threading algorithm applied to the email clients, however, as a forensic way to construct conversations for multi-person, his algorithm still has some problems to resolve. In order to achieve high quality email mining, it is necessary to reconstruct email conversations at first. This is exactly the problem addressed in this paper.

In this paper, we present a novel thread-based conversations construction mechanism for an effective analysis and statistics of the email flows for multi-person. The main contributions of our mechanism are:

1) A data extraction rule for email headers extraction and redundant emails filtering for multi-person,
2) A messages mapping algorithm to help the messages without Message-ID find the correct Parent/Child relationship,
3) A topic-based heuristic to merge or decompose threads to conversations.

The mechanism is tested by a corpus collected from four volunteers of a research team. Compared with Outlook Express and Zawinski's algorithm, the results indicate that it performs better and is feasible and valuable in digital forensic analysis process.

The rest of this paper proceeds as follows. Section 2 details the related work. Section 3 shows the extraction of email data. Section 4 describes how to reconstruct conversations. Section 5 discusses experimental setup and results. Section 6 concludes this paper.

2. Related work

Past attempts [2, 3, 4] which work on email threads have largely focused on visualization of thread information. Deepak P et al [5] detailed the analysis of Enron email threads and quantification of employee responsiveness. Its threading approach is different from ours, since we not only use the subject lines but also the References and other headers of email to construct the conversations. And the EMT [6, 7] ranks the email users in social network by their responsiveness mentioned in [5]. BuzzTrack [8] presented a topic-based TDT algorithm to cluster emails, however, the Parent/Child relationship is not focused on as ours.

Most similar to our work is the message threading algorithm proposed by Jamie Zawinski [1]. It is used in Netscape Mail and News 2.0 and 3.0. The Java implementation of this algorithm is available in the Grendel source. This algorithm is also described in the imapext-thread Internet Draft by Mark Crispin and Kenneth Murchison [9]. In the algorithm the Parent/Child relationships are built using two methods: reconstructing a message's ancestry by the References header contained within it; and checking the original (not base) subject of a message to see if it is a reply to (or forward of) another message. Our approach is applied to construct conversations with multi-person's email data. In order to keep the correct Parent/Child relationships in the conversations, we adjust the order of Message-IDs in the wrong References headers, and improve his algorithm by adding the processes of message mapping, threads decomposition and merging.

3. Extraction of Relational Data

There are three fields in email headers contributing to threading: 1) Message-ID, a unique identifier for the message; 2) In-Reply-To, Message-ID of the message to which this one is a direct reply; and 3) References, Message-IDs of the message's ancestry. In practice, many email clients generate and use the References header instead of In-Reply-To, so sometimes there is only one of them in the mail. Because the Message-ID is appended when the mail passing SMTP servers, a mail in a sender's outbox has no Message-ID before sending. But the corresponding mail received by a recipient has been appended the Message-ID header. In fact, the two mails are considered as the same one.

Firstly, the corpus collected from multi-person are parsed, and for each email, a record is inserted to a database with its sender, send-time, recipient, subject, Message-ID and ReferenceIDs. The ReferenceIDs field is populated from the References and/or In-Reply-To headers. If a mail only has one of the two headers, set ReferenceIDs the same as the one. If a mail has none of them, set ReferenceIDs NIL. The Message-IDs in a References are displayed in order. The first one is the parent of the second, and the second is the parent of the third, etc. In fact, some clients append a Message-ID to References in the opposite order. So if a mail has both two headers, the In-Reply-To should be considered firstly. For each mail, if the Message-ID in the In-Reply-To isn't same as the last one but the first in the References, adjust the References by turning the first one to be the last and set ReferenceIDs the same as the adjusted References, otherwise set it the original one.

The redundant records have to be filtered: 1) For the records with Message-ID, we delete the duplicate records with the same Message-ID; 2) For the records without Message-ID, if its corresponding records with Message-ID can be found in the database, delete it, else retain it. The corresponding records can be matched by send-time, sender, recipient and other features. After filtering, the records are ordered by send-time. All the above work can be done by database SQL queries.

4. Conversation Reconstruction

Step 1 (Message Threading): In this step, a table is created for associating emails' Message IDs with their parents', called T_P/C. And a hash table, T_Mesg, is created, which associates emails' Message-IDs with their message objects about the mails' information. Then the two tables are populated by Zawinski's algorithm to build the Parent/Child relation links with each record. A dummy mail is created to be the parent of the mails with NIL ReferenceIDs. To each mail without Message-ID, a unique Message-ID is assigned for it. To each mail whose Message-ID only can be extracted from References, set its message object NIL.

Owing to the ordered records, T_P/C is populated in send-time order too, and it plays a key role in message threading. The construction of a thread tree is a recursion process starting from the "root" mail. After the recursion of each mail in T_P/C, the tree has been created, and then we delete T_P/C.

Step 2 (Nodes Mapping): Each node of the tree represents a mail identified by its Message-ID. There are three kinds of mails in the tree as we show in the Figure 1(a): 1) The really existent mails with Message-IDs in the database; 2) The dummy mails with the NIL message objects which don't exist in the database (deleted by the email users), such as M'. Their Message-IDs are traced from the References of other mails, and use their earliest descendant's send-time and subjects to instead of theirs; 3) The mails with the Message-ID assigned by us, like M. As we address in section 3, their corresponding mails have Message-IDs which will be quoted in References of the reply mails. So if there is a corresponding mails M of M' in the thread tree, replace M' with M and use the Message-ID of M' to instead of the one assigned by us. We define the following heuristic to define the mapping.

— Map M' to M where:

- They have the same parent mail;
- The earliest child of M' is sent after M;
- M' or the parent of M' has the same base subject with M;

- If M' has children, the senders of them must be in the recipients list of M.

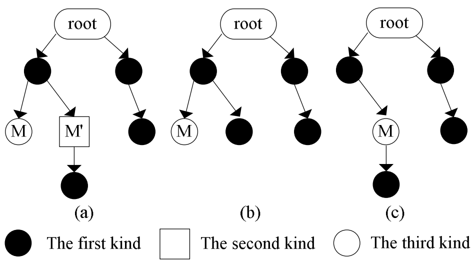

Figure 1 A simple case of a thread tree. (a) is a tree got by step 1. (b) is a tree got by Zawinski's approach. (c) is a tree after mapping

If more than one mail be found after above filtering, chose the sent-time nearest one. Figure 1(c) shows the mapping result of (a). (b) is the result through Zawinski's approach, which M' is deleted without mapping, and its child is promoted to its level. The Parent/Child relationship of the mails is broke.

Step 3 (Threads Decomposition): Each subtree under the root is a thread. A thread should be divided into several when the mails in it have the different base subject lines. Traverse each thread under the root, for each mail, if its parent isn't root and its base subject is different to its parents', break the links with its parent and promote it to be the last child of root as the first mail of a new thread. In order to reconstruct the conversation completely, most dummy mails are retained but the dummies in top-level under the root, for their disadvantage in threads merging. Traverse each child of the root, for each mail, if it is a dummy and only has one child, delete it and promote its child to its level, but if it has more than one child, retain it.

A difficulty is the topic of thread drift. Someone may change the subjects of email which really belong to one thread, and the threads Decomposition will mistakes such case. In terms of the robustness, the text mining can be better to this difficulty. Gladly, the average number of topics per thread in the development corpus is just 1.02 – this problem applies to only a small number of emails.

Step 4 (Threads Merging): The threads with the same base subject should be merged. When merging, the primary and the secondary relation of the threads and the conjunctional points should be considered. Firstly, the threads are sorted by their first mail's send-time. Then traverse each first mail of the threads. For each mail, if there is a sibling mail with the same base subject sent before it, merge the current mail S with the sibling mail P. There are three cases about the merging.

a) Both S and P are dummies: Make S a child of P.
b) S is a dummy but P isn't: Create a new dummy and make both P and S children of it.
c) S is not a dummy:

- Traverse each descendant of P to find a mail that meets the follow rulers to be the parent of S: 1) It's sent before S; 2) The sender of S is in its recipient lists; 3) Its send-time is nearest with S.
- If none can be found, use case a or b to merge according to whether P is a dummy.

Such merging mechanism makes the early one of the two threads primary, and the secondary one may be a subtree or a sibling-tree of the primary. If the first mail of the secondary is dummies, the parent mail of S can't be found. So a dummy is made to be both two threads' parent, take the case a or b as an example. If the first mail of the secondary is not dummies, the parent mail of S may be found in primary through the feathers mentioned in case c. When the parent mail still can't be found, a dummy parent should be made as case a or b.

Considering the merging mechanism, if someone uses the same subject in different threads which not belong to one conversation, the two threads should not to be merged. Thus the two threads' time interval and mail accounts should be considered. If time interval greater than 3 days or the sender of the first mail of the secondary is not in the list of mail accounts included in primary, never do the merging.

After above steps, the Conversation Reconstruction is over. Each conversation is clustered by topics, and keeps the Parent/Child relationship which will contribute to email statistics.

5. Evaluation

5.1. Corpus

We collect four Master students' emails to be the corpus. Enron corpus [10] is rejected to use for two reasons: First, it is cleared by other researchers, hence only contains the mails about the business. Second, some features in email header ("References" and "In-Reply-To") contributing to threading are cleared too.

The corpus is manually divided into conversations by the subject. It covers one academic semester from Dec 1, 2007 to Jun 1, 2008. Table 1 shows the resulting split. The number of mails of the Total Emails is not the sum of the four students', since the duplicate mails are filtered by the way addressed in Section 3.

Table 1 Detail of the corpus

Email Data Set	Number of Mails	Number of Conversations
Student 1	471	294
Student 2	320	186
Student 3	274	156
Student 4	132	81
Total Emails	1067	613

5.2. Evaluation Methodology

Though there are many forensic tools integrated with email parsing and analyzing, none of them groups email by conversations. Fortunately, many email clients provide such service. The performance of our approach is compared with OE's and Zawinski's using the Total Emails data set.

The performance is measured from four aspects: P_{ncd}, the precision of new conversation detection, F_{ncd}, the false alarm of new conversation detection, P_{ct}, the precision of conversation tracking, P_{rk}, the precision of Parent/Child relationship keeping. D_c means the new detected conversations that belong to the conversation set split by us. T means the conversations that just contain all the correct children mails. R means the conversations that keep Parent/Child relationship right. Four specific measures are displayed as follows:

$$P_{ncd} = \frac{|D_c|}{|Totle\ conversations|} * 100\% \quad (1)$$

$$F_{ncd} = \frac{|\overline{D_c}|}{|Totle\ conversations|} * 100\% \quad (2)$$

$$P_{ct} = \frac{|T|}{|Totle\ conversations|} * 100\% \quad (3)$$

$$P_{rk} = \frac{|T \cup R|}{|Totle\ conversations|} * 100\% \quad (4)$$

Besides these, the benefit of our approach is showed from another perspective: ART, the average response time of mails in a conversation. RT is defined as the difference between the send-time of a mail and its parents'. ART is the average of the sum of RT in a conversation. Each relative distance between a real conversation's ART and the ART calculated by each approach are compared to show the influence of each approach on the accuracy of ART.

5.3. Evaluation Result

The results are showed in the Figure 2, 3, 4. In the Figure 2, the performance of our approach is obviously greater than the others. It detects all the conversations, and keeps high precision (more than 92.02%) both in P_{ct} and P_{rk}. But its F_{ncd} (5.55%) is higher than the others (0%). It is caused by the threads decomposition. When parsing the mails to get subjects, some Chinese words are decoded to confused characters which affect the accuracy of threads decomposition. Since the absence of threads decomposition, the other two approaches have 0% false alarm, and both of them have a similar performance in each aspect. But OE is a little weak in the P_{ncd}, because it misses two conversations which will introduce a loop in the Parent/Child relationship.

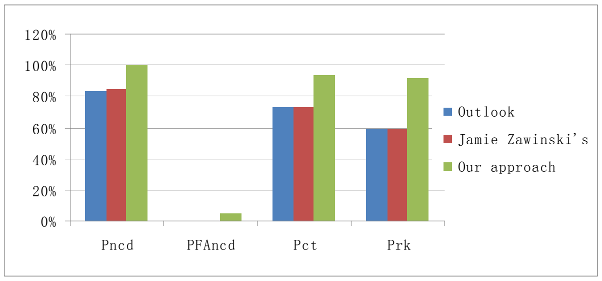

Figure 2 The performance compared by the three approaches

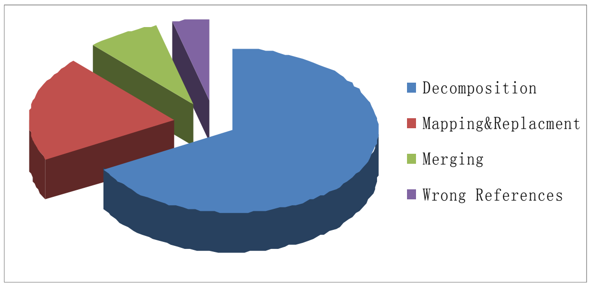

Figure 3 The factors affecting the performance of the three approaches

Figure 3 shows the factors that cause the difference among the three approaches. Threads decomposition takes a large percentage, for the new conversation detection is influenced seriously by it, so is the conversation tracing. The other three factors take an important role in keeping Parent/Child relationship. Figure 4 shows the relative distance of ART affected by each approaches. For comparison, the conversations with correct Parent/Child relationship through all the approaches are not displayed. If a conversation is not detected, the D_ART is 100%. Obviously, our D_ART is more near 0%, so it is closest to the real ART.

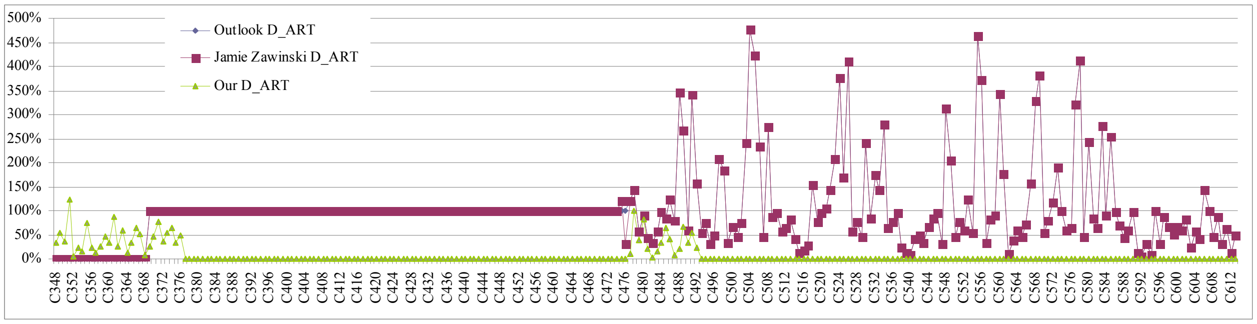

Figure 4 The distance of ART for three approaches

6. Conclusion

As an important medium of digital evidence, email artifacts reconstruction and analysis is necessary. We designed and evaluated a mechanism for conversation reconstruction based on mail threading. There is quite a few works about threading multi-person's emails before ours. In the mechanism, the mails without Message-IDs can be threaded in the correct positions and a new conversation started by the "Reply To" button also can be detected by threads decomposition. It keeps the Parent/Child relationship as far as possible.

However, there are several challenges. It relies on send-time of the mails to threading. When the mail's send-time is a forgery, the results may be affected even introduced a loop in the tree. While mapping, if more than one mail can be found, the send-time nearest one we chose may not be correct. We endeavour to find some ways to reduce the false alarm of the new conversation detection caused by wrong decoded Chinese characters, which is a part of the future work. Also, we plan to design a new way to visualize the thread tree which contains many dummies.

Acknowledgement

This work is supported by the Natural Science Foundation of Zhejiang Province of China under No.Y106176, and the science and technology research Planned Projects of Zhejiang Province of China under Grant No.2007C33058.

References

[1] Jamie Zawinski (2002). "Message Threading". *http://www.jwz.org/doc/threading.html.*

[2] Bernard J. Kerr (2003). "Thread arcs: An email thread visualization". *Technical Report RC22850 (W0307-148). IBM Research Report*. IBM Research, One Rogers Street, Cambridge, MA 02142.

[3] Gina Danielle Venolia, Carman Neustaedter (2003). "Understanding sequence and reply relationships within email conversations: a mixed-model visualization". *In Conferenceon Human Factors in Computing Systems*.

[4] Adam Perer, Ben Shneiderman (2005). "Beyond threads: Identifying discussions in email archives". *Technical Report A264044*, Human Computer Interaction Lab, Maryland University, College Park.

[5] D.G. Deepak P, V. Varshney (2007). "Analysis of enron email threads and quantification of employee responsiveness". *In Proceedings of the Text Mining and Link Analysis Workshop on International Joint Conference on Artificial Intelligence*, Hyderabad, India.

[6] Salvatore J. Stolfo, Shlomo Hershkop, Chia-Wei Hu, Wei-Jen Li, Olivier Nimeskern, Ke Wang (2006). "Behavior-based modeling and its application to Email analysis". *ACM Transactions on Internet Technology (TOIT)*, v.6 n.2, pp. 187-221.

[7] Salvatore J. Stolfo, Shlomo Hershkop, German Creamer, Ryan Rowe (2007). "Automated social hierarchy detection through email network analysis". *In Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, San Jose, California, pp. 109-117.

[8] Gabor Cselle, Keno Albrecht, Roger Wattenhofer (2007). "BuzzTrack: Topic Detection and Tracking in Email". *In Proceedings of the 12th international conference on Intelligent user interfaces*, Honolulu, Hawaii, USA, pp. 190 – 197.

[9] Mark Crispin, Kenneth Murchison (2002). "Internet Draft: IMAP THREAD". *http://www.jwz.org/doc/draft-ietf-imapext*-thread-12.txt.

[10] B. Klimt, Y. Yang (2004). "Introducing the enron corpus". I*n First Conference on E-mail and Anti-Spam (CEAS)*.