UNIVERSITY OF TWENTE.

FOX IT
FOR A MORE SECURE SOCIETY

Identity ranking in digital evidence data.

Scalable identity extraction and ranking in Tracks Inspector

Master thesis

Master thesis project
Master Computer Science
Track Information Systems Engineering

Jop Hofste
j.hofste@student.utwente.nl
s0145122

Faculty of Electrical Engineering, Mathematics and Computer Science
University of Twente, Enschede, The Netherlands

26th October 2012

Abstract

The digital forensic world deals with a growing amount of data which should be processed. In general, investigators do not have the time to manually analyze all the digital evidence to get a good picture of the suspect. Most of the time investigations contain multiple evidence units per case. This research shows the extraction and resolution of identities out of evidence data. Investigators are supported in their investigations by proposing the involved identities to them. These identities are extracted from multiple heterogeneous sources like system accounts, emails, documents, address books and communication items. Identity resolution is used to merge identities at case level when multiple evidence units are involved.
The functionality for extracting, resolving and ranking identities is implemented and tested in the forensic tool Tracks Inspector. The implementation in Tracks Inspector is tested on five datasets. The results of this are compared with two other forensic products, Clearwell and Trident, on the extent to which they support the identity functionality. Tracks Inspector delivers very promising results compared to these products, it extracts more or the same number of the relevant identities in their top 10 identities compared to Clearwell and Trident. Tracks Inspector delivers a high accuracy, compared to Clearwell it has a better precision and the recall is approximately equal what results from the tests.
The contribution of this research is to show a method for the extraction and ranking of identities in Tracks Inspector. In the digital forensic world it is a quite new approach, because no other software products support this kind of functionality. Investigations can now start by exploring the most relevant identities in a case. The nodes which are involved in an identity can be quickly recognized. This means that the evidence data can be filtered at an early-stage.

Preface

After five and a half year of studying the only courses rested were the research topics and the final project. I always wanted to perform the last part of my study at a prominent IT company. After performing some research to IT companies in the Netherlands I gathered a couple of companies I was interested in. In november of 2011 I was on holiday in Aruba, Netherlands Antilles. Here I suddenly encountered a small building with the Fox-IT logo on it. I stepped inside and asked for the oppurtunity to perform my master thesis at Fox-IT. After some weeks I got the an e-mail from Fox-IT Netherland with an answer on my question. I had some meetings with Hans and Andre and we agreed by starting in march 2012 at Fox-IT. After we agreed I searched for a supervisor at the University of Twente and Maurice was able to be my supervisor. A couple of months later Djoerd was joined my group of supervisors. After working and writing seven months on my project, the master thesis was done and the result is here.

Jop Hofste

Acknowledgements

In march 2012 I started with my master thesis at Fox-IT. In the recent months there were some people who supported me in this research and I would like to thank them in this way.
First of all I would like to thank my supervisor Maurice and Djoerd from the University of Twente for their support and feedback during this process. The meetings, especially with Maurice were very useful, thanks to keep me motivated and for your valuable input.
I would also like to thank my supervisors at Fox-IT, Andre and Hans. Hans for his critical overview and his input about identity extraction, resolution and ranking. Andre for his help and support for the real implementation in Tracks Inspector and for introducing Tracks Inspector to me. I had a great time at Fox-IT and it was a pleasure to work in the Tracks Inspector team. Next to this, I would like to thank Rutger and Daniel for their support when I was implementing the proposed extensions of Tracks Inspector. Furthermore, I would also like to thank Bart and Brendan for their support by obtaining testcases and testing my implementation in Tracks Inspector.
Finally, I would like to thank Mart for his support and advice in my master thesis. His feedback was valuable.

Graduation committee

1. Dr. ir. M. (Maurice) van Keulen (University of Twente)
2. Dr. ir. D. (Djoerd) Hiemstra (University of Twente)
3. Dr. ir. J. (Hans) Henseler (Fox-IT)
4. Ing. A. (Andre) Post (Fox-IT)

Table of Contents

Chapter 1

Introduction

1.1 Background

This section describes the motivation for this master thesis project. The main reasons for the extension of Tracks Inspector are discussed. Furthermore, some background information about Fox-IT is given, the company where this master thesis was performed.

1.1.1 Motivation

In the world of forensics the amount of digital data that is involved in cases is growing. The analysis of this data is done by digital experts and can take a long time. Nowadays in each investigation digital evidence is involved because most of the people have a mobile phone, USB sticks, CD's / DVD's or any device that contains personal digital information. All these data carriers can hold information about possible suspects. If ordinary investigators want to know anything about a data carrier, they must ask specific questions to a digital expert what they are looking for and what they expect to find. It can take weeks before investigators get information from the digital expert, during which the investigation is on hold. In addition it is not sure that the digital experts deliver all the necessary data, because they do not know the context of the investigation.

Therefore there is a demand for a tool with which investigators can easily and quickly browse through digital data. In this way analysis in digital evidence data can be done fast by arbitrary investigators, who do not have many IT-skills. If this functionality is supported by a tool, many more cases can be investigated and probably more cases can be solved, because more investigators can do digital investigations.

Most of the current forensic tools like Encase and FTK focus on browsing or searching through evidence data, technical knowledge or training is needed to use this software [43]. The tools are also not user friendly [33]. The performance of them with huge amounts of data is also very bad [55]. If there are evidence units with a huge amount of data, it is almost impossible to browse through all data due the lack of time. Therefore other methods are needed to guide investigators to find the interesting evidence data. Current tools in general have two methods for investigation. Investigators are able to browse through evidence data, a tree list of the directory and files of the original evidence unit is given. With this view investigators can see the folder structure and which files are located in the folders. The extended search possibility is the second method. The complete evidence unit can be researched on file size, file name or the contents of a file. This is interesting if investigators know what they are looking for. It is likely that files of the complete evidence unit are grouped by type resulting in fast and easy investigations. All picture files grouped in one overview give the investigator an complete overview of all pictures in that specific evidence unit. Also all internet history that is found on the evidence unit grouped together should give an interesting overview of all visited websites and web applications by the users of the evidence unit, regardless of the browser choice.

It would be nice if investigators can see which people are involved in an evidence unit or case. Such an overview can list the most important people in a case. Probably all strings that are found in the predefined fields do not refer to real world people, that is why we call them identities. Identity overviews should give investigators a clear overview of the most important identities in a case. From that perspective investigators can discover data that is linked to a specified identity. This can lead to a completely new approach of finding evidence when the data is grouped by identities. Investigators can directly see which identity is the most important one and which data is related to this.

Fox-IT can assist detectives in their investigation with the development of Tracks Inspector(TI). This research is meant to provide more insight into the use of identities. Furthermore, this research will provide an answer as to how these identities can be integrated into TI, to further assist investigators in their research.

1.1.2 Fox-IT

Fox-IT[1] is a specialized IT security company with much knowledge about security and criminal investigations. The headquarters of the company is located in Delft, The Netherlands. About 150 employees are stationed there. The company was founded in 1999 by the current directors, Ronald Prins and Menno van der Marel. The company was started as the Fox-IT Forensic IT Experts BV. The two directors have both gained their knowledge at the Dutch Forensics Institute (Nederlands Forensisch Instituut). Because much knowledge has been gathered with regard to access digital data, they know the weaknesses of the contemporary security and forensics. Fox-IT wants to realize security solutions and to solve problems where security is an important issue. Fox-IT is known for their confidential methods to work with clients and client data, therefore all employees are screened by several agencies.

Fox-IT has currently three main business units; Crypto & High security, Cybercrime and Forensics. The Forensics department consists of three subdivisions.

1. The first division is training, here training courses are given to customers on several subjects like investigations on Internet and digital forensic research.
2. Forensic research is the second division, here research on potential evidence data is done for current customers on all kinds of data carriers.
3. Tracks Inspector is the division where a tool is made for analyzing evidence data, which is explained in detail in the next section.

Fox-IT has a reputation to be an expert in information security. Therefore customers expect that Fox-IT will treat their data in a secure way. Leaking information may never happen because data might contain state secrets. It would be a disaster for both the customer and Fox-IT if this where to happen. That is why Fox-IT works with physical and non-physical security measures to enforce the confidentiality.

1.2 Research

This section describes the objective and scope of this research project. In addition the main research question and the sub research questions are stated. Finally the approach is given.

1.2.1 Objective and scope

This project is an extension of Tracks Inspector on multiple levels. Tracks Inspector is a large project and therefore it is needed to define the objectives and the scope of this research project. Multiple objectives can be defined here. To be more precise; the main objective is clearly defined as follows:

[1] https://www.fox-it.com

> *"Extend Tracks Inspector with identities overviews and dashboards to assist investigators in their digital investigation. The identity dashboards must guide investigators to find the most interesting real world people in a case."*

Tracks Inspector analyzes data from physical data entities, data images, data archives or directories. The purpose is to extract personal information from these sources, for example the usernames of system accounts and mentions of identities in email archives. From this extracted information a list with identities can be generated. If multiple highly similar identities with more or less the same context exist, they can be merged or a likelihood per identity can be proposed, indicating how much they are the same. The first step is analyzing Windows system accounts and email archives to generate possible identities. Further the scope is extended to Internet history and meta fields of documents. Also mobile phone extraction is added to parse address books and chat messages. Deeper inspection of the contents of emails and documents can be done afterwards, with natural language processing(NLP) techniques. Probably a full-text search with a list of identities based on the content of documents and e-mails is a more easy than using NLP techniques. It is a safe method to extract more information out of the contents.

The focus in this research project lies on proposing interesting and usable identity dashboards to the user of Tracks Inspector. Accordingly, that means that the identity extraction and identity resolution must perform well. The underlying focus is to extract and merge identities from the available sources and to sort them in such order that the most relevant identities are shown first.

1.2.2 Research questions

In this research project a data model is developed that defines identities and their occurrences within Tracks Inspector. This is based on several sources, which are already parsed and also on other sources that are not parsed at the start of this project. The parsed sources are the sources Tracks Inspector supports. That means that Tracks Inspector has a processor for the source type and can insert meta data and the contents of the file in one of Tracks Inspector's tables. Also statistical methods are used to determine the relevance of identities. From this perspective the following main research question arises.

Main research question:

> *"How to extract, merge and rank identities from heterogeneous sources containing personal information, to propose a valuable identities overview?"*

To divide the main research question into several sub questions, the main research question is split into two stages. First stage is the extraction of personal information, questions 2 and 4 follow from this. The second stage is the merging stage. For this the data structure is needed (question 1) and a statistical method to merge the identities is needed (question 3). To test the correctness of the extracted results, it is compared to other software used in the domain of digital forensic research (question 5). For practical reasons the scalability must be assumed (question 6) and a user preference is to detach sources from an identity (question 7).

The sub research questions of this research project are listed below:

RQ1 What data structures need to be developed within Tracks Inspector, which represents identities and data mappings between source elements and the associated identities?

RQ2 How to extract identities from structured fields?

RQ3 What statistical method can be used to determine the relevance of identities?

RQ4 Which methods can be used to extract identities from unstructured text out of source elements?

RQ5 How accurate are the results Tracks Inspector delivers compared to other software in this domain?

RQ6 How to ensure the scalability of the identities extraction and merging process with respect to the source data?

RQ7 How to decouple sources from an extracted identity and assign them to another identity?

1.2.3 Approach

The approach is partly based on the "ideas about the future vision" on digital forensics by Garfinkel [26]. The point that is elaborated is "moving up the abstract ladder". He proposes the need for an identity management system included with identity resolution and disambiguation. Individuals must be modeled in such a way that their sources where they are build from can be computed. That means that all elements like names, email addresses, telephone numbers and other identification numbers can be linked to an identity. All available sources in Tracks Inspector that contain identity related information are used to propose identities to the user, these ideas come also from the supervisors of Fox-IT.

The implementation should be tested to determine the quality of the attached functionality. There are five datasets used for evaluation. Four datasets are gathered from the Enron corporation and one mailbox of the author is used. These datasets consist of mail archives. These archives consist of email messages and their corresponding attachments. The attachments can be other email messages, but also documents, video, images or whatever file format.

Three evaluation methods are used to evaluate the implemented functionality. The first method is a quantitative approach of evaluation and has as main goal to make a comparison between the results with regard to identities of Tracks Inspector and another commercial product called Clearwell. Tracks Inspectors functionality is totally different than Clearwell's and only a small part of both tools is compared. In the second method the top 10 identities are compared among Tracks Inspector and two other commercial products. This results in an overview of the similarities among the products and from this the performance of Tracks Inspector can be concluded. The scalability after implementing the identity extraction and resolution functionality is tested in the third evaluation method. The execution times of the datasets on both software versions(with and without identities processing) are compared to each other to conclude the impact of identity processing.

The first contribution is to demonstrate identity extraction from structured heterogeneous sources like system accounts, email, documents and mobile phone exports. In the field of structured and heterogeneous data the research towards and the practical testing about identity resolution after the extraction of identities is the second contribution. This project contributes also by proposing a method to use relevance feedback in manual interactive identity resolution. Another expected more social contribution is to assist investigators in their investigations, by proposing the most important identities with their corresponding sources and derived statistics. Investigators can quickly get a feel of which identities are involved in this case with as result that investigations finish faster. This result in the possibility that more criminal cases are resolved. It is an expected contribution because investigators are probably not able to use the software in practice before this thesis finished, so the user feedback is probably not on time.

1.3 Problem exploration and terminology

This research deals with a lot of terminology, which can have different meanings in different contexts. To prevent confusion, this section describes the most used terms in this report. First the most important problems are explored.

1.3.1 Problem exploration

Figure 1.1 shows an object diagram (UML technique [23]) with an example of the identity John Doe. The identity "John Doe" is created by the analysis of several sources. The identity is built from three sources. In these sources the phrase "John Doe" is found. The appearance of such a phrase in a source is called a mention. The identity "John Doe" probably refers to a real world person with this name.

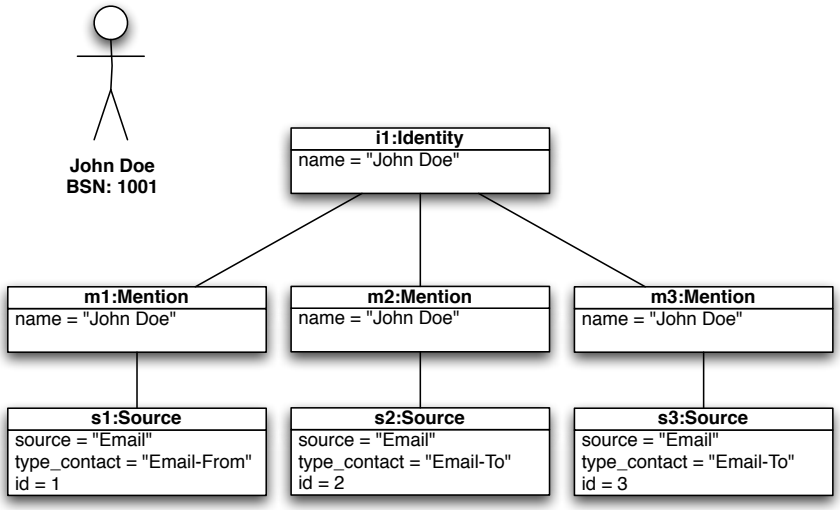

Figure 1.1: Object diagram - Identity example John Doe

The difference between a real world person and an identity is schematically illustrated in a class diagram listed in figure 1.2. An identity has a name and can have multiple mentions. A mention has a link to a source. An identity represents a real world person, so indirectly a real world persons can be linked to an identity. That means that the real world person is indirectly connected to the underlying mentions and their sources. There can be multiple identities that represent the same real world person. For example identities with the name "John Doe" and "JohnDoe123" can refer to the same real world person.

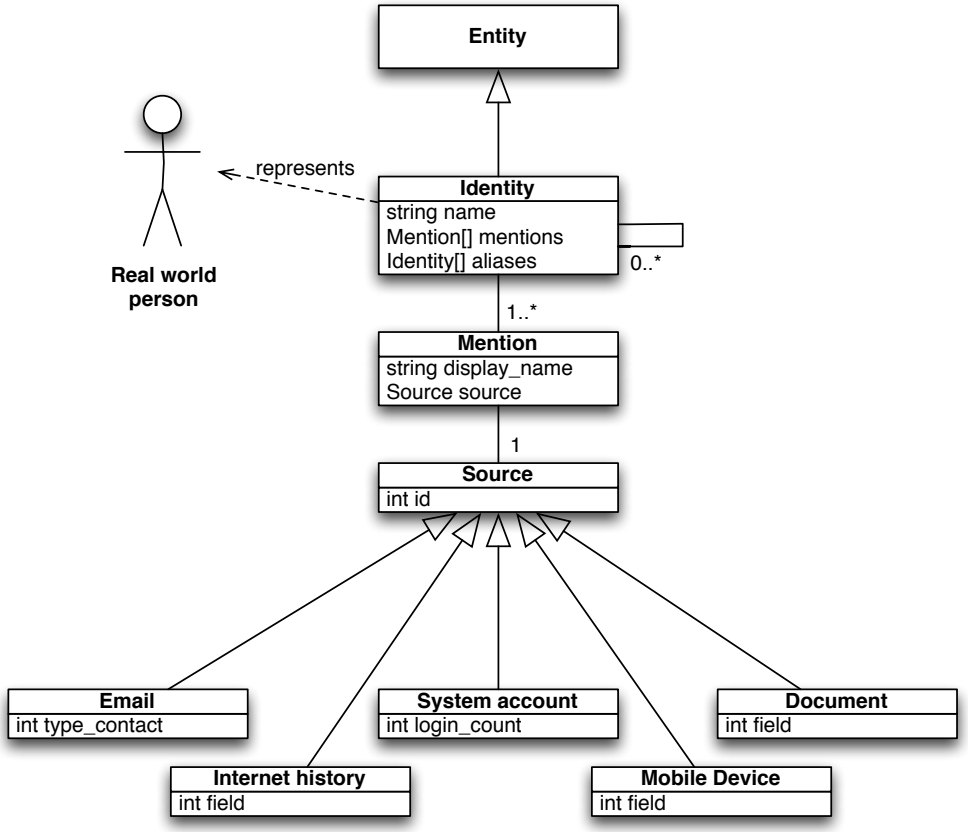

Figure 1.2: Class diagram of Identity and mention

With this proposed design, three issues can be encountered.

Issue 1 - Multiple mentions can refer to the same identity.
This issue is based on figure 1.3. To explain this example a sentence is proposed:

> **John Doe** says: "I like Vera". Vera says: "I like **John** too".

In this sentence the mention "John Doe" can be extracted and proposed as the identity John Doe. The mention "John" in this example is a reference to the mention "John Doe" in this context. This results in an identity with multiple mentions. The mention "John" is coupled indirectly to the identity John Doe, because it is a reference to the mention "John Doe". The type_contact attribute of the email source is NULL, because this sentence can be from the body of the email and probably not from the "E-mail-From" or " Email-To" part of the email. These fields most of the time only contain names of persons or companies, but no complete sentences.
The other mention (most right in the figure) "John Doe" that is linked to this identity John Doe has a a Email-To field as source. This field has also a coupling to an e-mail address, because the Email-To fields consist of a display name and an e-mail address. The format "John Doe <john.doe@example.com>" is a widely used format for this. The mention "John Doe" on the right side is a mention of the identity John Doe here. This implies that the identity John Doe is linked to the e-mail address john.doe@example.com, because the mention has a coupling to an

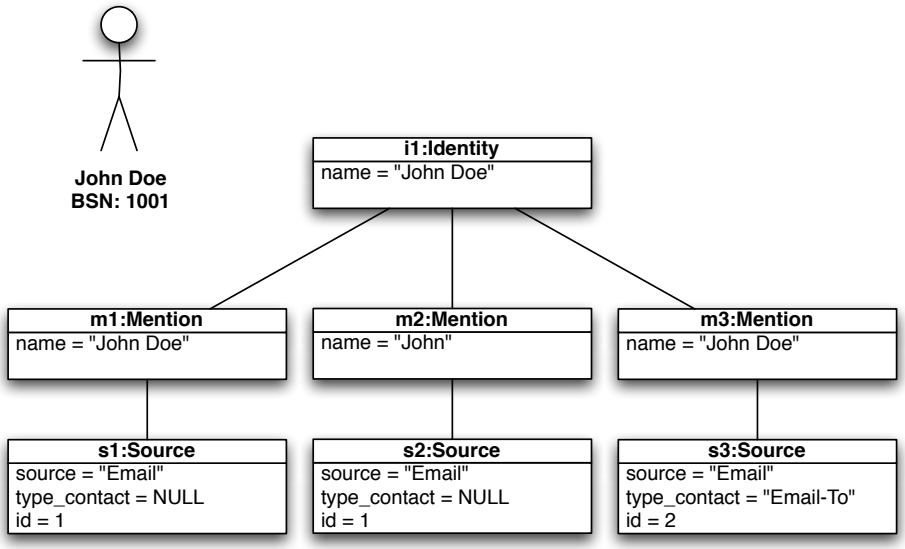

Figure 1.3: Object diagram - Multiple mentions can refer to the same identity (Issue 1)

Email-to source. Links to telephone numbers are also available when mentions are extracted from phone books or contact lists.
There is also another side case in this scenario. If John was replaced by 'him', the field of anaphora resolution is touched [49]. The phenomenon of referring to a mentioned entity is called an anaphora. Anaphora resolution is an method to find that previous mentioned associated entity.

Concluding, multiple different mentions can refer to the same identity, if the mentions are references to each other. The mentions do not need to be the same, but they can be the same. In this example both are illustrated in the figure. Other information can be included as well, when this information is available in the sources where the mentions are extracted from.

Issue 2 - Multiple identities can refer to the same real world person.
If multiple identities are extracted, they can refer to the same real world person. For example identity "John Doe" is found as sender and recipient from some email messages and an identity "JohnDoe123" is extracted from a cookie of Facebook. They can both refer to the same real world person "John Doe", see figure 1.4. Both of the identities "John Doe" and "JohnDoe123" refer to the same real world person. Suggest if they are both extracted from the email-from fields, then he has two aliases or two mail accounts with different sender names (email-from names).

Issue 3 - A single identity can refer to different real world persons.
An identity John Doe, that has multiple mentions can also refer to different real world persons (see figure 1.5). If an identity is found with the name "John Doe" and in this world (or case) exist two or more persons with the name "John Doe" without any other information it can not be concluded where the parts or the complete identity "John Doe" refers to. Both of the real world persons "John Doe" can be involved in this identity. This is a disambiguation problem. It probably does not happen in many cases, but if it happens, investigators must be aware and they probably must be able to detach sources from an identity. Items or complete sources (like a complete email archive) should be coupled to other identities which can refer to another real world

person. Mentions are coupled to the same identity if there is no reason to disambiguate. It can be possible that there occur some mismatching if equal mentions are merged without observing the context.

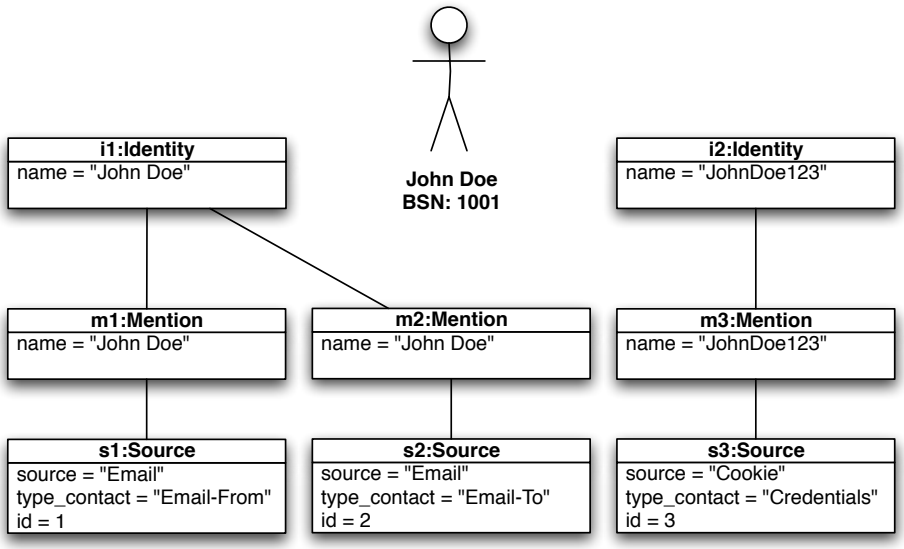

Figure 1.4: Object diagram - Multiple identities, same real world person (Issue 2)

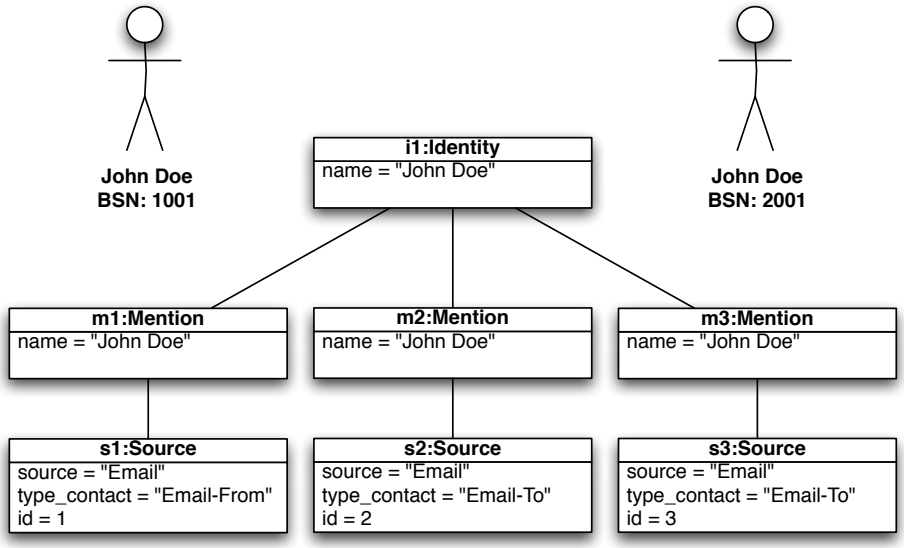

Figure 1.5: Object diagram - Double person problem, identity disambiguation (Issue 3)

1.3.2 Terminology

Now that the issues have been described, a terminology list will follow below. The most used terms are described.

Fox-IT Fox-IT is a company headquartered in Delft, The Netherlands. The company is known for their advanced knowledge about (encryption) security, cybercrime and forensics.

Tracks Inspector(TI) Tracks Inspector(TI) is a solution developed by the forensics department of Fox-IT. Investigators can use the tool to analyze digital data.

Real world person A real world person is a unique human being in the world, who can be referred to. A real world person has a name, but this is not the real identifier of a person. Real world people with the same name can be distinguished from each other by their Citizens Service Number(CSN) in Dutch known as the *Burgerservicenummer* (BSN). Real world people have multiple identifiers like the Citizens Service Number, the first name and/or the last name. There can exist multiple real world persons who have the same name, but who are not the same.

Identity An identity is an object which is intended to refer to one real world person. It is an abstract composite representation of references to a real world person. Analyzing sources where references to real world persons are mentioned generates this representation. An identity is an object, which is identified by an identity name. Two identities with the same name cannot exist. In this context an identity is different then an identifier. An identifier identifies one specific person [10], an identity tries to do this but can refer to multiple persons, if not enough information is available because there are real world persons with the same name. To separate such identities, it should be possible to disconnect sources from identities and couple them to an other identity. New identities can also be introduced to manage real world persons with the same name.
The name of an identity is unique. It cannot be concluded directly that an identity with name X refers to the real world person with the name X. Because the identity X can be generated from other sources where the specified person named X does not occur in. That means that an identity is not an identifier for a real world person, but it can represent a real world person.

Identity name A string which represents the name of an identity. This is the unique identifier of an identity.

Mention A mention is the appearance of an identity name in a context. An identity can have multiple mentions and each identity has statistics about the appearance of mentions in source data.

Entity An entity is a reference to a real world object. Same as an identity, but this can also refer to a a location an organization or misc. Entities represent real world objects, meaning that a single entity can refer to multiple real world objects. Examples of an entity are "Amsterdam" or "the blue car".

System Account A system account is an object with properties like a name, login count and last logged in time. Operating systems contain system accounts so that multiple users can use the same operating system, but they can have separate home directories, programs and desktop views.

Disambiguation Disambiguation is the process of determining what a mention or an identity refers to. In a sentence or context, similar mentions can refer to other real world objects. For example in a sentence like "John Doe sends a email message to his colleague John Doe", here the mentions John Doe refer to different real world objects.

Evidence unit An evidence unit is an image, directory or physical device with contains digital evidence data. Evidence units are input for Tracks Inspector and are separately processed. Evidence units can be assigned to at most one case.

Case A case is a named group, a synonym for a case in this context is a project. A case has an identifier and optional descriptions. Evidence units can be assigned to a case. Users of the system can have different privileges in different cases.

Custodian A custodian is an owner of an evidence unit. On most devices properties of the owner cannot be found. Therefore Tracks Inspector users can add a custodian to an evidence unit.

1.4 Structure of the report

In chapter 2 an overview of the current tools and methods in the field of forensic data analysis is provided. The most important literature with regard to information extraction, string matching algorithms and relevance feedback are also discussed. Chapter 3 proposes the requirements for the extensions of Tracks Inspector regarding identities. In chapter 4 the identity extraction is explained in detail. This extraction which extracts the identities per evidence unit is described here. Chapter 5 describes the identity resolution steps. The method and system design are described for the automatic identity resolution, executed at case level. The relevance determination for the identities is described in chapter 6. A method that add weights to every identity is explained in this section. Chapter 7 describes the user interface and the user interaction that is available to investigators using Tracks Inspector. The merging of identities is especially explained here. In chapter 8 the evaluation is proposed. The methods for the evaluation, the validation and the associated results are described. The testdata that is used with their statistics is also proposed in this section. Chapter 9 draws the conclusions and give a number of recommendations. The appendices shows ERD diagrams of the database extensions and the comprehensive results of the experiments. Appendix C goes into more detail on Tracks Inspector. The global architecture is described and the file types and formats Tracks Inspector supports are explained.

Chapter 2

Literature review

In this chapter the literature is explored. In the first section the current state of forensic data analysis is described. The analysis of email / internet history and the operating system Windows are listed. The current tools on the forensic market are also discussed. In the second section the literature about information extraction is explored. The third section describes the most widely used string matching algorithms and after that three string matching methods are proposed. The fourth section gives an overview of relevance feedback.
This chapter also answers RQ4: "Which methods can be used to extract identities from unstructured text out of source elements?". The field of the extraction of identities from unstructured text is analyzed here.

2.1 Forensic data analysis

This section describes the current literature about forensic data analysis. Internet history, user account extraction and file meta data analysis are discussed.

2.1.1 Email data analysis

In forensic research, quite some investigation towards email data has been done. Iqbal et al. [35] propose a solution to find the authorship of an (anonymous) email. They try to find frequent patterns in the email messages by using data mining techniques. In another study [62] they propose an Email Mining Toolkit, which can quickly process email archives. This results in a list with VIP-contacts and it sorts the emails in logical order for direct inspection by an investigator. The project members have direct contact with New York Police detectives for feedback and testing. Social network philosophy and email data analysis can also be compared to find patterns in email archives like in social networks. It is possible to generate such network-based view, based on filters like keyword, date time and custodians [32].

2.1.2 Internet history analysis

Internet history is stored data about peoples behavior on the internet. This data gives insight in the websites where the user has browsed through. Internet history is available on the most operating systems where browsers are used. In this literature the focus lies on the most widely used operating system Microsoft Windows [3]. The forensics tool "Netanalysis" [70] that performs an analysis on browser data to extract the internet history from it, browser like Internet Explorer(v5-9.0), Mozilla Firefox(v1-7.0), Google Chrome(v0.2-14.0), Safari and Opera are supported. The tool shows the history and cache data which are stored on the disk. The software also includes HstEx, which can recover deleted browser data so that this can also be analyzed. The location of all the browser data varies greatly among the different operating systems and browsers. From a high perspective the browser data can be divided in four categories.

- The cookie data are files stored in hidden folders in the user directory. The cookie files are plain text files and some information is stored in index.dat files.

- Secondly the Internet history data, this data is stored in index.dat files. The history is tracked per day, per week and there is a master file where the overall history is stored in.
- The cache files also known as the temporary Internet files also contain history data. Here data of visited web pages are stored to load pages faster.
- The last part is other data, this varies per platform but one example is the last typed urls.

In the most modern browsers this feature is also used to auto complete urls. When you type "uni" in the address bar you get a list with options of pages you visited before. Options can be like "university of Twente" or "university of Amsterdam" or whatever page you visited with that string inside.

2.1.3 Windows data analysis

In Windows there are multiple sources to extract forensic data from. The *user accounts* information are interesting. This information is stored in a so-called Security Accounts Manager(SAM) file. This is a registry file and if this is parsed, all the current user accounts can be extracted. The login count and if the account is active and some other information can be extracted. This however only works for local users. If a domain account is used for logging in, then the account information is not stored in the SAM file. There is a registry entry [2] named *Profile list* which provides entries for users which have ever logged on into the system. For each different account a sub key is added to this registry key.
The meta data of *files* is also interesting. Properties of documents which can be extracted are; the language of the file, the last authors, the created / last modified date and the title of the document [13, 14]. This information can be valuable if investigators want to know the last modified documents for example.

2.1.4 Digital forensic tools

There are quite some digital forensic tools available for investigation. One of them is Xiraf, a forensic warehouse system developed by the Netherlands Forensic Institute (NFI). This tool uses a XML Database layer upon an Oracle database solution for the storage and XQuery as the query language to get information out of it. It seems that the tool can extract much data, but it also seems that usability here is not taken into account. If you want to use the tool it seems quite complex for an average investigator. Xiraf can only execute queries when the processing of the evidence units is done completely [1]. The SANS Investigative Forensic Toolkit is also a forensic application which supports all kinds of evidence images and file systems. It has several (external) software packages included to analyze the data. It is available[1] for download and runs on the Ubuntu operating system. Another well-known toolkit is the Forensic Toolkit(FTK) by Access Data [22]. This is a scalable forensic solution which processes all kinds of data. Also password recovery and detecting encrypted files are supported. The tool is database driven so crashing of one of the components will not lead to completely reprocessing of all data. The tool shows views with much technical depth. Guidance Software has developed Encase that is also one of the most used software packages in the forensic world. This software can be used for multiple purposes. The first one is acquisition, to create forensic images of data carriers. Another functionality is the analysis and thereafter the reporting of the data, but also file recovery is supported by EnCase [9]. For both FTK and EnCase special training is usually required because the tools are very complex. There are also open source solutions in the field of digital forensic tools. The Digital Investigation Framework is one solution; it has a file browser included and supports most file systems. It also reconstructs and analyzes the registry of Windows. Another nice feature is a timeline analysis, which provides a view where operations through the time can be analyzed [4].

2.2 Information extraction

Information extracting (IE) is a widely used concept that has different meanings and subdomains. In this context the domain is an environment with user-oriented data, which means that probably

[1] http://computer-forensics.sans.org/community/downloads

every piece of data can have one or more connections to real world persons or identities. Information extraction here means extracting information out of the data from predefined fields [68] or natural text. In natural language text processing the extraction can be done by a pipe and filter design pattern presented here [29]. This is a typical example of how these systems analyze natural text. Information extraction from web pages is also researched in the last years, here also natural language processing based tools are used [37].
Two of the sub domains of information extraction are Named Entity Extraction and Named Entity Disambiguation, both are discussed below.

2.2.1 Named Entity Extraction

Named Entity Extraction (NEE) or also called Named entity recognition is finding and extracting entities from unstructured text [18]. The main idea is to derive structured data from unstructured text, so that unstructured text becomes more clear. Entities are references to a real world object. It is a generalization of identities that refer to real world persons. Towards this topic most research showed the extraction of four different entities; person names, organizations / companies, locations or miscellaneous [8, 50]. The misc classifying means classified as none of the other categories, but anything that has a name.

In NEE there are two widely spaced fields; the rule based and the statistical based. These fields can also be combined without the use of dictionaries [46]. This research belongs also to the "Message Understanding Conferences"(MUC). These conferences are important because they offer test data for the participants to test their information extraction or language processing methods. More specific in the direction of person names extraction, Minkov et all [48] show the extraction of entities out of informal texts like emails. They conclude that informal texts have less informative types then formal text. This means that it is more difficult to determine the entities in informal text instead of formal text.

The university of Waikato, New Zealand, provides a tool[2] that allows you to use the semantics from Wikipedia in your own application [47]. It also extracts the relevant topics out of the context. To detect links between words it first performs a candidate selection(select most important word / word groups), then a link disambiguation(check the meaning of the candidate selection words) followed by link detection. The link detection is possible, because Wikipedia uses cross-linking in articles, so with machine learning techniques it is possible to detect links between words. A commercial tool *Stilus NER*[3] uses automatic named entity recognition that makes use of semantic tagging. This tool uses rules to determine possible variants of entities. For example if the dictionary has an entry of the name John Doe, all possible variants are determined for this entity, like "Doe, John", "J. Doe" etcetera. Pouliquen et all [53] have done research towards multi-language person name recognition. They propose a method to perform proper name recognition by storing the known names in a dictionary. To extract new mentions from unstructured text they use trigger words per language. Trigger words announce a person name; examples are 'Dr.' or 'professor'. Words before or after these trigger words are used to determine the possible person name. This is very language-dependent because in Germany for example every noun starts with a capital letter.

Focusing on the Dutch language, Tjong Kim Sang has done research towards the Dutch language in named entity extraction [65], he concludes that Carreras et all [12] have the best method for this language. They use Adaboost, a machine-learning algorithm to perform first the entity recognition and then a classification step.
For Dutch named entity extraction there are also some databases available, which can be used as dictionary. In The Netherlands the institute Meertens provides a surnames database[4]. The network for "onomastics" provides also first name and surname databases[5] with the most common Dutch names.

[2] http://wdm.cs.waikato.ac.nz/
[3] http://www.daedalus.es/en/products/stilus/stilus-ner/
[4] http://www.meertens.knaw.nl/nfb/
[5] http://www.naamkunde.net/

Rule-based entity extraction

Rule-based entity extraction consists most of the time of two main parts. A set of rules and a set of policies that control the multiple rules [58]. In this field there are rules for single and multiple entities. Single entities are for example person names or company names. With multiple entities, rules can extract structured records from the context. These rules are more complex and for example used in the WHISK tool [61]. Another rule-based tool is GATE [21] that also provides a custom rule scripting language. The framework has a core library and a set of language engineering modules which can be customized or extended.

Statistical based entity extraction

In statistical based entity extraction the best-known models are the Hidden Markov models, the Maximum Entropy Markov models and the Conditional Random Fields. A Hidden Markov model is a stochastic process that implements the Markov process with hidden states. It can be compared with a Bayesian network, which has sequences of variables which have their dependencies [54]. The Maximum Entropy Markov models use Hidden Markov models and combine this with maximum entropy. The differences with the Hidden Markov models are that the transitions and observations are replaced by a function that results in the probability of the current states, given the previous state and observation. Further training can be more effective in Maximum Entropy Markov models [44]. Conditional Random Fields is a framework to segment and label sequences. The indirect dependencies between entities can be graphical represented, but also the features of these entities [38, 45]. The Standford Natural Language Processing Group[6] implements the Conditional Random Fields method. The Standford Named Entity Recognizer is an open source implementation of named entity extraction.

2.2.2 Named Entity Disambiguation

Named Entity Disambiguation(NED) is a sub domain of information extraction; in literature the synonym entity-name resolution is used. NED is identifying named entities, like peoples names or locations. That means linking a mention of an entity in structured or unstructured text to a known entity. This is also based on the context where the mention is found [20]. Entity resolution is the process to determine if entities refer to the same real world objects [64]. Compared to identity resolution(section 2.2.3) this is a more general concept.
AIDA is a framework that supports entity disambiguation. It can process natural-language text and tries to map mentions of names onto entities like people, locations and places [34]. As source the DBpedia, Freebase or YAGO can be used. YAGO [63] is a semantic knowledge database that gathers its information from Wikipedia, WordNet and GeoNames. It contains 447 million facts on about more then 9 million entities. The resulting knowledge database is a large step by adding knowledge about individual persons and organizations. It supports also the RDF Schema standard so it can be used simply in other semantic applications.

2.2.3 Identity resolution

Identity resolution is a special type of Named Entity Disambiguation focused on people where multiple different data sources are analyzed to find identity matches. It must be possible to mark identity P on source Y to identity Q on source Z as the same identity. That means that all identity information of P and Q can be merged to extend the whole identity. Identity resolution in the literature focus on the real identifier(the identity) to identify one person. In this research the identity definition is slightly modified, see the terminology list in section 1.3.2. The definition used here is the definition from the literature, the real identifier. A more extended profile of identities can be made with identity resolution. Nowadays there are two main approaches for identity resolution. There are simple duplicate detection techniques and on the other hand machine learning / artificial intelligence techniques that rely on probabilities [24]. Some of the simple duplication detection techniques are discussed in section 2.3. Many data items probably lead to the same identity, for example links or URI's can point to the same location, but also email addresses that

[6] http://nlp.stanford.edu/software/CRF-NER.shtml

can lead to the same person, for example a company email address and a private email address of a specified person [31]. These links can be defined in the Web Ontology Language(OWL) with an *owl:sameAs* tag. The Web Ontology Language is a language in RDF/ XML format to describe the semantic web. The semantic web is a rather vague concept. It means that users can find and combine information easily, but also to share this information by using for example the OWL language.

Record linkage is more or less a synonym of identity resolution. Record linkage is a more general term for identity resolution and is primarily focused on finding records that refer to the same entities. In this field also common identifiers are used to link records [71].

Master data management is a master set of data, which is used in organizations. This data is the starting point for all other applications in the organization, which need entity or identity data. The amount of data in organizations is exploding, that is why there is a need to manage the data consistently, so that there is one data set that is trustworthy so that other applications or data sets can rely on it [40].

OYSTER is an open source project sponsored by the university of Arkansas. It supports record linking and master data management processes. This is more a entity resolution tool than aimed at identity resolution. It seems to have a lot of support in the whole identity field, like resolution and capturing of identities [28]. There is also a people search engine which uses identity resolution named Pipl [52]. They also make a distinction between people / identities with the same name, this is based on the location and if available the age of the specified person to make the distinction between the results.

2.2.4 Validation of Named Entity Disambiguation

To validate the entity disambiguation or identity resolution there is a statistical method to demonstrate the success rate [41].
Focusing on identities the precision means the union of the relevant identities and the retrieved identities divided by the retrieved identities. The numerator of this fraction can also be indicated as the correct founded identities in this search. The denominator means the total retrieved identities in this search. Precision is the percentage of correct found identities in a search query.
The recall has the same numerator, but the dominator is the number of relevant / correct identities. That means that recall is the percentage of correct found identities as part of the total correct identities.
The F measure shows the accuracy of the search, it can be considered as the weighted average of both, precision and recall. The values of the F-measure can between 0 and 1. If the value is closer to 1, that means a better accuracy.

$$precision = \frac{\#(correct\ identities\ \cap\ retrieved\ identities)}{\#(retrieved\ identities)} \tag{2.1}$$

$$recall = \frac{\#(correct\ identities\ \cap\ retrieved\ identities)}{\#(correct\ identities)} \tag{2.2}$$

$$F = 2 \cdot \frac{precision\ \cdot\ recall}{precision\ +\ recall} \tag{2.3}$$

2.3 String matching algorithms

The known string comparison algorithms from the literature are discussed here. Thereafter three proposed matching algorithms are compared by some examples.
Snae [60] did an analysis of the most common used name matching algorithms. He divided the matching algorithms in four categories; spelling and string based algorithms, phonetic / sound based algorithms, composite algorithms(composite of phonetic or sound) and hybrid algorithms(spelling and sound combined). Another algorithm for person name matching uses three

comparison methods [53]. The first one is a bi-gram similarity that compares chunks of two characters of the sentences. The second one is the tri-gram similarity, which is the same as the previous one but instead compares three characters. The last comparison first strips all vowels of the words and thereafter it performs a bi-gram similarity, this is also called the consonant bi-gram similarity. Christen [15] provides recommendations that assists practitioners in selecting a name matching technique which is suitable for the situation. He concludes that there is no best technique which suits every situation. The characteristics of the data and the processing power available have to be considered.

2.3.1 String comparison methods

Identities are matched by identity matching functions. These functions are based on string metric functions used in literature. Many research has been done on string metric methods. In the literature there are four specified well known functions [17, 24].

- The edit-distance like functions, these are functions like the (Damerau-) Levenshtein distance and the Hamming distance. These functions compare the distance between two strings by counting the number of operations needed to transform the first one into the other one [39].
- The token-based distance functions are functions that have as a starting point two bags of words. The union and intersection between these bags is the distance between the strings. An example of this function is the Jaccard index [30].
- The third method is the hybrid distance function. In this function the strings are split by the space character and for each word it is checked if the word exists or a part of it in the other list of words. These occurrences are counted [51]. A well-known function is the bi-grams, or more general, the so called q-grams distance function. This function cuts the sentence or word after q characters and results in all possibilities after these separations. The similarities in these chunks are compared if two or more sentences are compared [67].
- Phonetic functions are functions that are based on the sound of the word. These kind of functions check if strings are phonetically similar. Soundex [57] is probably the most well known in this category and is primarily intended to match surnames of people. This could be a nice addition if the display names of identities are parsed in tuples, so that the name is divided into the first name, prefix and surname. Each part of the tuple can be analyzed individually.

2.3.2 Proposed string matching

Inspired by the literature above, three basic matching functions are proposed. In table 2.1 the display name differences which occur in probably practice are compared. Below the three proposed methods are explained in more detail.

1. exactMatching
   In exact matching strings are case-sensitive compared, the only preprocessing that is done before this comparison is to remove the spaces before the whole string and after the whole string, thus the spaces between the words will remain here. If this comparison results in true, then you can state that with reasonable assurance these identities are the same.
2. semi-exactMatching
   Semi-exact matching is a case-insensitive matching, and trimmed matching. This function is an extension of the exact matching function. The functionality which is added is that all spaces are removed from both strings and the comparison is done in lowercase.
3. possibleMatching
   A less exact matching, but perhaps more practical is the so called possible matching. Possible matching is also a semi-exact matching, extended with the check if one string is completely inside the other string (See table 2.1 - comparison 5). A match where the punctuation is

fully stripped(comparison 6) and then checked if it is completely inside the other string. Furthermore characters which have umlauts or cedillas are converted to the original characters and after that the comparison is done, see example 4. This matching function has some side marks, if for example the last name is found and identified as an identity. The next stages in the processing might be identify multiple possible identities to this identity, because there are multiple users with the same last name available in a specific piece of data present. These considerations are given more attention later on.

ID	Comparison	exactMatch	semi-exactMatch	possibleMatch
1	A: "John Doe" B: "John Doe"	True	True	True
2	A: "John Doe" B: "john doe"	False	True	True
3	A: "John Doe" B: "JohnDoe"	False	True	True
4	A: "John Doe" B: "John Doë"	False	False	True
5	A: "John Doe" B: "John Doe // Comp."	False	False	True
6	A: "John Doe" B: "John D."	False	False	True
7	A: "John Doe" B: "John Deo"	False	False	False
8a	A: "John Doe" B: "John"	False	False	True
8b	A: "Doe" B: "John Doe"	False	False	True
9	A: "J. Doe" B: "John Doe"	False	False	False

Table 2.1: String comparison examples

Another option is to use the implemented soundex algorithm in MySQL[7]. The current implementation works well for the English language, using this also for other languages can lead to wrong results.

2.4 Relevance feedback

Relevance feedback is an old but an useful technique. Relevance feedback allows users to give feedback on search results. This improves the queries after this feedback. This feedback can be the rating of individual results of a query or to determine which results are interesting and not interesting in the results view of a query. Relevance feedback is often used in information retrieval systems, especially in content-based systems [7, 56]. User feedback is an interesting factor in relevance feedback, also the type of user that gives feedback matters [6]. There are three types of relevance feedback, described below.

1. Explicit feedback
   Explicit feedback is a manually feedback method [69]. Users are able to grade or mark results. One example is to grade the result, for example with stars from 1 - 5, or to mark result as relevant or not relevant.

2. Implicit feedback
   Implicit feedback is based on artificial intelligence techniques. After implicit feedback the

[7] http://dev.mysql.com/doc/refman/5.5/en/string-functions.html#function_soundex

system can try to find patterns in the given feedback to determine the intentions of the user [19]. Logging of the user behavior help to give implicit feedback. For example the categories someone visit often can be showed higher in a list or more prominent in the screen, instead of other categories. There is no interaction needed from the user of the system, implicit feedback will happen behind the scenes and the user is not explicitly informed. Implicit feedback just stores all the user interactions.

3. Blind feedback
   Blind feedback (also known as pseudo relevance feedback) is a feedback method which automates the manual part of explicit feedback. Therefore there are no assessors needed. It seems like implicit feedback, but the difference is that after a query the top X of results will be stored and marked as relevant. No other logging of relevant items is stored. First a small set of documents are retrieved according to the query. The system tries to get better results with the small feedback included. The system tries to get more accurate results and propose this to the user. This method will lead to better results [11, 25].

A well known relevance feedback algorithm is the "Rocchio Algorithm". In this algorithm the relevant and non relevant documents are separated. The optimal query in this algorithm is the difference between the centre of the relevant documents and the centre of the non relevant documents [42].

2.5 Discussion

This section has described the literature which is relevant and related to this research. The information extraction, string matching algorithms, relevance feedback and the current status of forensic data analysis are discussed.

In the field of forensics there are currently many forensic tools which support digital data analysis. Email data, Internet history data and Windows data analysis are discussed including the current software which can analyze this type of data. On the basis of this it can be concluded that there is no 'easy to use' software which focuses on arbitrary investigators in contrast to digital experts. For digital experts there are tools available like FTK, Encase and Xiraf.

The most important topics in information extraction related to this research are discussed here. The following observations were made about the named entity extraction, the technique to extract entities from unstructured text. 1) This technique together with named entity disambiguation can be used to extract and disambiguate entities from unstructured text. 2) A special type of named entity disambiguation is the identity resolution which can be seen as a very complex problem in this research. 3) Identity resolution focus on determining identity matches among different sources. The methods to determine precision and recall are also proposed and can be used to prove the correctness of the found identities.

The most well known string matching algorithms are discussed, like edit-distance, token-based and phonetic functions. The functionality of the function is explained. For identity resolution purposes, custom string matching functions are proposed to determine the equality of strings.

There are multiple types of relevance feedback which are discussed here. With relevance feedback users are able to give feedback after a search query. In association with string matching algorithms like phonetic functions, users can be assisted more adequately by merging identities, because the system can propose more appropriate results. When relevance feedback is implemented it stores the users feedback and uses it to give a more appropriate result after a search query.

Chapter 3

Requirements

The main goal of Tracks Inspector is to assist investigators in their investigation. Investigators need to decide quickly if a specified person must stay in preliminary imprisonment or not. If the tool Tracks Inspector could offer the identities functionality, investigators are able to see overviews of identities. Because most of the time the amount of data is overwhelming, there must be smarter ways to do investigations. Investigators can be assisted with case wide identities overviews. These overviews propose all identity related information together and ordered. The identities overviews must be clear and must show the most relevant identities in descending order.

This chapter proposes the product requirements of the identity extensions of the Tracks Inspector tool. The functional and non-functional requirements are listed. In the next section the global idea of identity extraction and resolution is explained.

3.1 Global system design

This section describes the identity extraction and resolution process in more detail. Figure 3.1 shows the architecture of the identity extraction and resolution process. This global process is modeled like a pipes and filter architecture [27]. The arrows are the pipes and the four blocks below are the filters. The input data is the input of the system and the output data is the output that is generated after running through the pipes and filters.

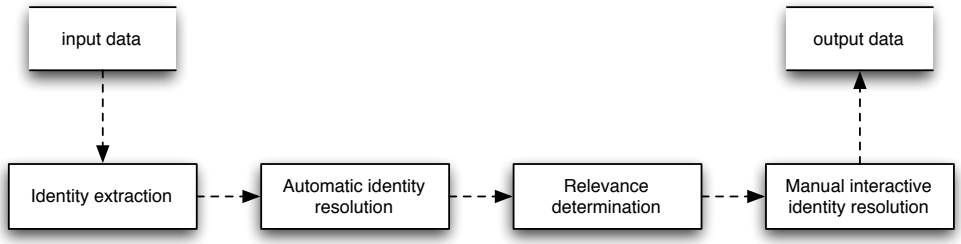

Figure 3.1: TI high level identity extraction and resolution design

The high level design is more elaborated with interim results in figure 3.2. The **input data** are evidence units. In the processing stage of the evidence units the **identity extraction** takes place. For each individual evidence unit the identity extraction is executed. That means that the identity extraction process results in separate identity lists per evidence unit. If the processing of an evidence unit is not completely finished these lists can change continuously. After the complete

processing we assume these lists as given. Evidence units can be part of a case. If they are no part of a case the process is terminated here. If evidence units are part of a case the **automatic identity resolution** takes place. In this filter the individual results of the evidence units are merged to get a merged identity list. This list shows an overview of identities per case. After this the **relevance determination** is done. Per identity a relevance count is determined and the result is sorted on the relevance count in descending order. This result is proposed to the users of the Tracks inspector. Users can perform **manual interactive identity resolution**, that means that users can manually merge identities. For example, if an user wants to merge identity 3 and 4, he/she can perform this action in the user interface. This last step of identity resolution is a manual process. The system can assist users by merging identities to group similar identities together. Users can also bookmark identities, with as result that bookmarked identities are always sorted on the top of the list.

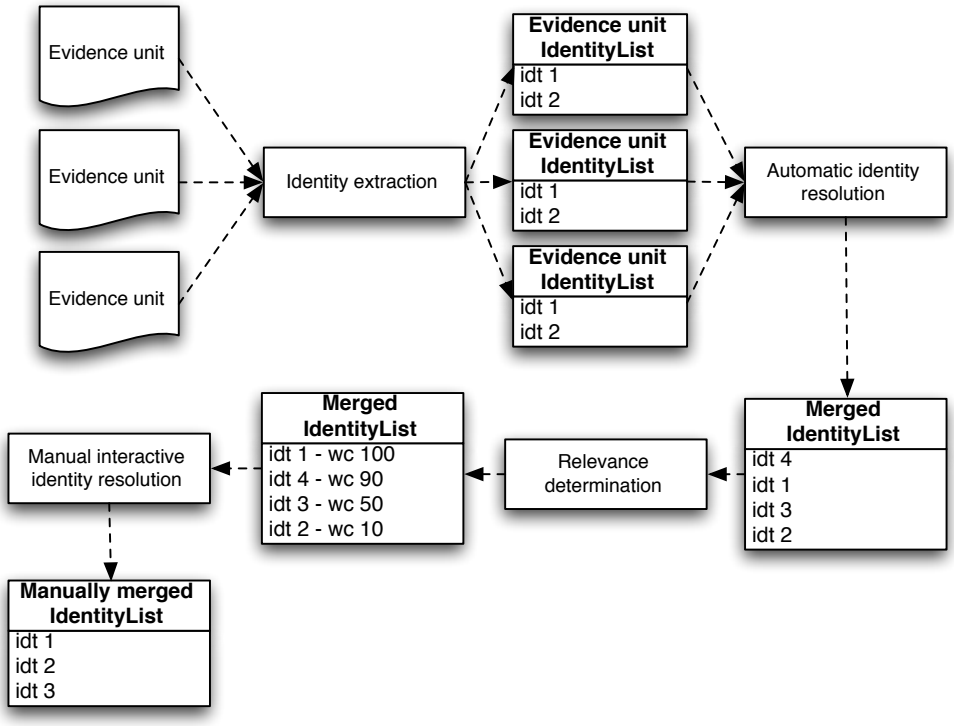

Figure 3.2: TI identity extraction and resolution process

3.2 Functional requirements

This section describes the functional requirements sorted by the stage where it belongs to. The stages where are addressed to are described in the previous section and can be found in figure 3.1. All requirements are prefixed with an identifier starting with a capital R.

3.2.1 Identity Extraction

R1 The system should extract identities per evidence unit.
Extracted identities can be from many sources. The list below shows the items were we focus on.

- The system should extract identities from system accounts.
- The system should extract identities from the 'from' and 'to' fields from emails.
- The system should extract identities from meta data of documents.
- The system should extract identities from internet history.
- The system should extract identities from chat history.
- The system should extract identities from address books.
- The system should extract identities from outputs of telephones, like UFED or XRY outputs.

Example items were we do not focus on:

- The system should extract identities from cookies.
- The system should extract identities from registry information.
- The system should extract identities from user ID's in de NTFS file ownership.

R2 The system must assign a relevance to each extracted identity, to define the participation of an identity in an evidence unit.

3.2.2 Automatic identity resolution

R3 The system should merge identities based on the "semi-exactMatching" string matching algorithm if the associated evidence units are in the same case.
When the matching function results in a match, identities should be merged. If the result is false, a new identity will be created.

R4 The system should merge the statistics of identities if identities are merged.
The sum of the statistics of the identity included with their aliases is computed.

R5 The system must propose the list of the most relevant identities. This list must be sorted by the relevance in descending order.

R6 For the relevance calculation of identities the system must support at least a statistical weighted method.

- The weights of the statistical weighted method should be easily configurable.

R7 Users must be able to see where the identities relevance is based on. The statistics about the sources where an identity is constructed from must be well proposed. Sources can be system accounts, emails, meta data from documents etcetera.

3.2.3 Manual interactive identity resolution

R8 Users must have an analysis dashboard, where they can see a identity/evidence unit matrix. The matrix must propose the appearance of identities per evidence units, resulting in a clear view to see the most important identities and the associated evidence units.

R9 Users must be able to merge identities manually by selecting first the identity which is the parent and after that selecting the children.

R10 Users must be able to manually add an identity to an existing merged identity.

R11 Users must be able to detach an identity from a merged identity.

3.3 Non-functional requirements

This section describes the non-functional requirements, also divided in the three stages.

3.3.1 Identity Extraction

R12 If identity processing fails for whatever reason, this may not lead to a blocking system.

R13 The results of identity processing must be the same after reprocessing.
Evidence units can be reprocessed, all extracted data about the evidence unit are removed and the extraction process starts from the beginning. All data must be removed well.

R14 The results of identity processing must be the same on different hardware or at different operating systems.

R15 The scalability of the identities gathering and merging must be linear with the amount of data.

R16 If the evidence unit processing is finished the relevance of the identities and the statistics may not change. That means that after the processing the results per evidence unit does not change anymore, for whatever reason.

3.3.2 Automatic identity resolution

R17 If evidence unit processing is finished the amount of identities should be static, it can not possible that the identity resolution stage constructs more identities then extracted in an evidence unit. Less identities would be possible if identities are merged.

3.3.3 Manual interactive identity resolution

R18 The responsiveness of the front-end should be fast(<2 sec) if an overview of identities is requested.

3.4 Requirements mapping

The requirements described above results in three divided areas. The first area is the extraction of relevant identities from input data. Identity extractors must be developed to gather identities from input data. For data that have predefined names or any personal display name fields, a quick one to one mapping can be done to extract identities from that input data. The identity extraction will be discussed in detail in chapter 4.
The second area is how to merge the extracted identities per evidence unit in one combined overview per case. This is discussed in the identity resolution chapter (chapter 5).
The third area is how to propose the identities overview to the user and how to assist users to manually merge users. The user interface and user interaction are proposed in chapter 7.

Chapter 4

Identity extraction

This section describes the sources where identities are extracted from. Also the methods used for extraction are described. The system design is also showed; the tables which are added to the database schema are described here. The functionality added to the current architecture is also shown in the system design section. At the end of the chapter the choices are discussed. This chapter partially answers RQ1: "What data structures need to be developed within the Tracks Inspector, which represents identities and data mappings between source elements and the associated identities?". This chapter also answers RQ2: "How to extract identities from structured fields?", the structured fields where the identities are extracted from are described included the method.

4.1 Design decisions

Figure 4.1 shows the processing of an evidence unit. First all nodes / files are discovered on the specified evidence unit, this results in a tree representation of the evidence unit. Thereafter each file in the tree is processed. In this stage the file type is determined, based on the mime type of the file. Based on this file type the type of processor is determined. In the processing stage the file information is inserted in the evidence database. At the end of many file processors an identity

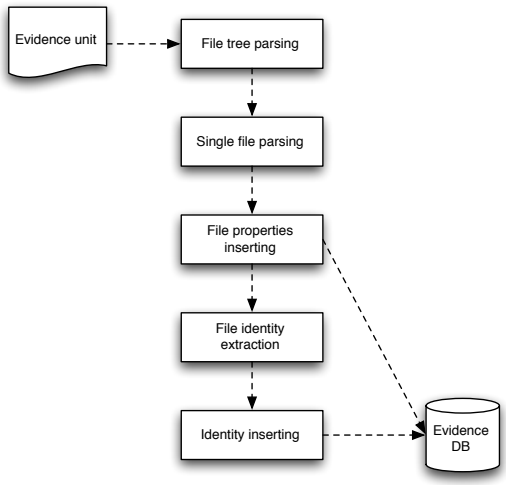

Figure 4.1: TI evidence unit identity processing / extraction

processor is located. Here the identity extraction is executed. It depends on the file processor how identities are extracted, this is discussed below in more detail per processor. The mapping between the source elements and the identities can also be made, after the identity inserting in the evidence database.

A choice is made for an approach which extends several separate processors to support the identity extraction. A choice could also have been made for another more centralized approach. It was a consideration that was made very soon. The centralized method should treat every file or node again. The idea of this approach was to make a generic post processor after the file processing. This gives much overhead because the file must be accessed again. The identity extraction results in extracted identities. The identities are objects which link files to identities. A choice is made for extending the current processors because the supported file types diverge a lot. So the extraction is encapsulated per file type in each processor. The system design for the identity extraction is discussed in section 4.3.

4.2 Sources

This section describes the sources where identities are extracted from. Each source has different fields that are used for the extraction of identities. These are discussed below.

4.2.1 System accounts

The operating system details and the operating system users(system accounts) are parsed. Local system accounts are parsed at the time of writing. If desktop computers are connected to a company network, most of the time domain accounts are used. These accounts are stored at a server and leave some traces on desktop machines. The *usernames* of the system accounts are used as input for the identities.

4.2.2 E-mail

For each email message the type of email is determined. Types of emails are incoming, sent or draft. This information is used for the identity extraction. In table 4.1 the email types and the fields are listed in a matrix. The headers of email messages are parsed. From this data the To-field, CC-field, BCC-field and From-field are parsed. If there is no header available (like sent and draft emails), the fields are parsed from the message itself.

Fields / Type	**To-field**	**CC-field**	**BCC-field**	**From-field**
Incoming email	Yes	Yes	Yes	-
Sent email	Yes	Yes	Yes	Yes
Draft email	Yes	Yes	Yes	Yes

Table 4.1: Email fields parsing for identities

The To- / CC- / BCC- fields are parsed for the incoming email, the sent email and the draft email. One of these fields contains the name or email address of the receiver of this email. For the sent and draft email types the from field is also parsed, because this contains the potential sender of the current mail(box) and should be interesting. For the incoming email the from field is not parsed, because then much noise enters the system. It is not really interesting where incoming email comes from because the principal persons are probably not located in these fields.
The fields (To, CC, BCC and From) all contain two items. The first item is the email address, the second one is the display name. This name is filled in by the email client most of the time. For the from field, the display name is often the name stored in the configuration of the email client. The To-, CC-, BCC-names are mostly derived from the address book or from a list of previous senders.
If the display name is not empty, the display name is translated to an identity. The mapping between the display name and the email address is also stored. If the display name is empty, the

email address is used and translated to an identity. The identity is constructed from the display name. The link between the display name and the email address is only stored, a new identity or mention for the email address is not made. The link is used to be able to reverse engineer the identity extraction process. It is possible to see the emails for example where the identity is extracted from.

4.2.3 Documents

For all documents the meta data is extracted from the files. There are two fields that can be used to extract identities. The first field is the meta_author or author field. This field indicates the original author of the document. This field normally does not change after the first time saving action. The second field is the last saved by field, indicating the user of system account which was logged on when the document was saved for the last time. This is something different than the last accessed time, which indicates the user who accessed the document for the last time. The strings parsed from these two fields are translated to identities. The mapping between these documents and the generated identities are also stored.

4.2.4 Internet history

The identity related information that is parsed from these files is the user name. On desktop computers, the user name can for example be the system account name of the operating system. Internet history is parsed from different types of devices. The user name can be empty or maybe manually set in some (mobile) browsers or operating systems. The user names are translated to identities as well. On identity level it is called the visiting URLs. The mapping between the identity and the internet history row are linked in the database.

4.2.5 Address books

At the time of writing only the contacts from the mobile device outputs are used as input for the address books. These contacts were in the contact list of the mobile device. Address book items mostly have a display name. The display name is used as input for the translation to identities. If there is an empty name the email address or phone number is used for the identities if one of those is available. The mapping is also stored between the generated identity and the original address book item.

4.2.6 Chat

The interesting fields in chat logs are the used display name(s) and the user name. An email address or telephone number can also be available. The display names are used for the translations to identities. If people change their display name, a new identity is also generated. The mappings between chat messages and the identities are also stored. A mobile device has a custodian, the owner of the device. This custodian is used to have a sender if chat messages are parsed from mobile devices. The output did not have a sender for chat messages like SMS / MMS.

4.3 Method & system design

The previous section described the sources used for the extraction of identities. This section describes the method and the system design for the identities extraction.

To support identity extraction in Tracks Inspector the database model must be extended. The evidence database host has a separate database for each evidence unit. This schema must be extended to store the identities which are extracted at processing time of an evidence unit. All above sources(section 4.2) are processed by processing units, see figure C.1 in the appendix. For each node, which can be a file or a directory; there are different node-processors. An overview of a processor is stated in figure 4.2. The pre-processors search evidence units on the operating system, the system users and the installed software. After that the node-processors check for each node which processor must be accessed. In the near future this architecture becomes plug-in based to make it easy to support other file or archive types. The processors which are extended for the

identities implementation are marked with the letters IDT(Identities) on the right side of each processor block. The information about the identities is stored in the evidence database.

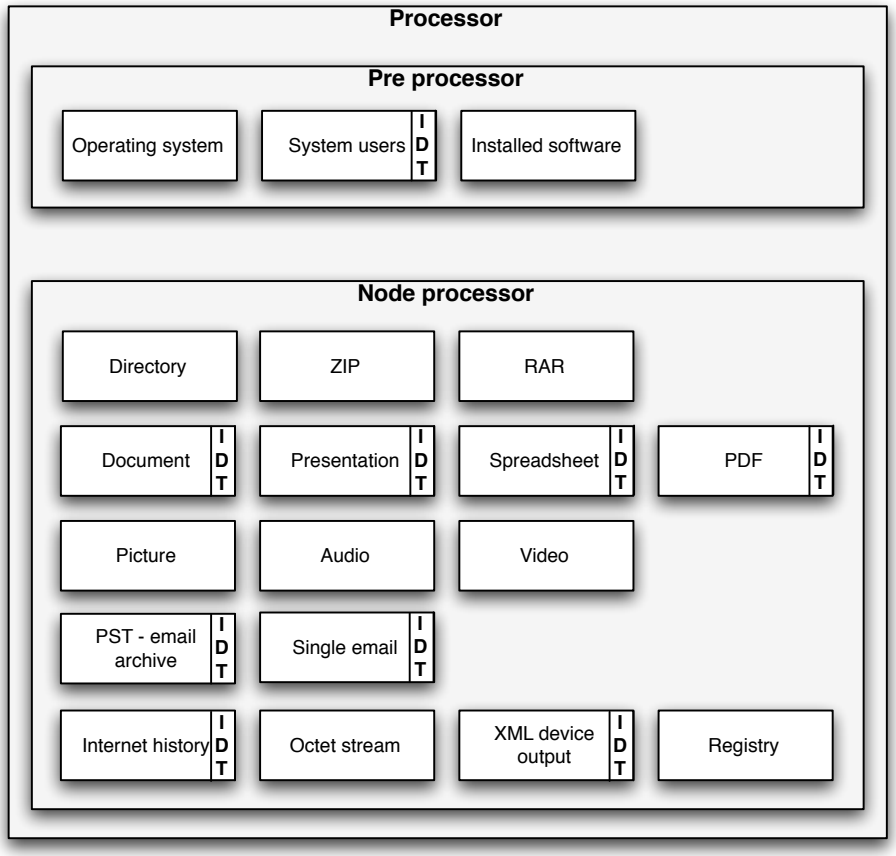

Figure 4.2: TI Processing - Subprocessors

For the extension of Tracks Inspector the evidence database is extended. The figure in the appendix A.2 shows the detailed entity relationship diagram(ERD) of the evidence database. Only the tables which are related to the identities are shown, otherwise the ERD become too large. A simplified ERD is listed in figure 4.3. One of the most important tables is the identities table. This table stores the display names of the identities with some side information if available. This column estimated weighted count is used for the sorting of evidence units on evidence unit level. In the ERD there are in total six mapping tables. These tables all have mappings from sources to identities. The sources which are mapped here are: documents(also presentation/ spreadsheets/ pdf), system accounts, emails, internet history, contacts and address books. The system_accounts table stores the system accounts that are parsed in the pre-processing stage. In the emails and email_contacts tables the emails with their associated information are stored.
In the textuals table the information about documents, spreadsheets, presentation and PDF files is stored. The internet history table stores the internet history rows with the associated protocol and host, the parameters are not ignored. These are stored in the url_get column in the internet history table.
The communication table has all communication stored inside. Communication can be calls, SMS,

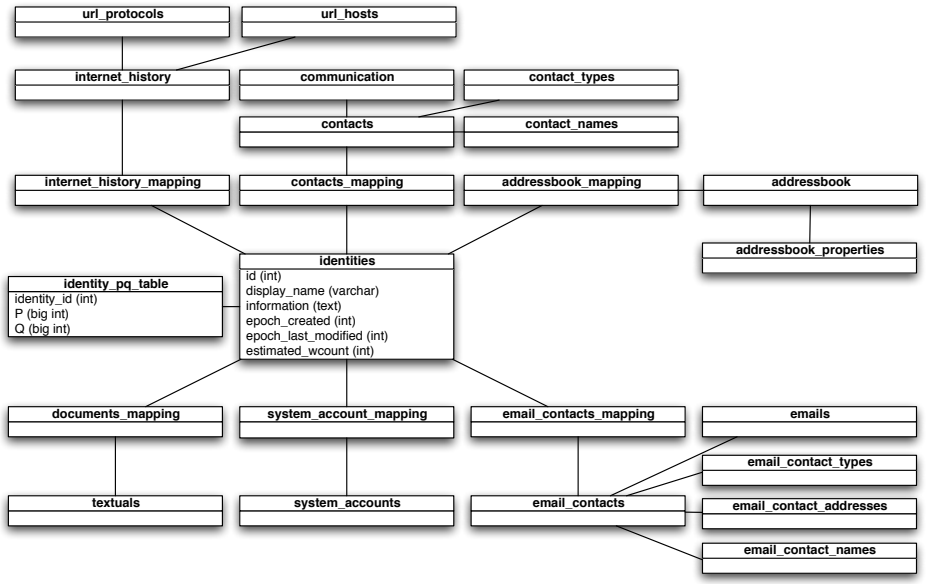

Figure 4.3: TI Processing - Evidence DB ERD

MMS or other chat messages. The contacts table stores the contacts where the communication is among. The tables which store the contact types and contact names are linked to the contacts table.
The address book table stores information parsed from address books. The display name that is found is stored here. The properties of the address book items are stored in the address book properties table. These are disjunct from each other because now contacts in an address book can have multiple phone numbers or other properties. The identity_pq_table stores the mapping between an identity and a P and Q range. P's and Q's are integers which identify a node in Tracks Inspector. The table is used for determining the P and Q ranges which are attached to this identity. This table is introduced for multiple reasons, the amount of nodes is very huge most of the time, joining all the tables which are stated in the diagram will takes too much time and is slow. The second reason for introducing this table is that ranges can be stored. If a complete container is attached to an identity, then the children of this node no longer need to be inserted in this table because they are implicit linked to the identity.

4.4 Discussion

This section discusses the choices made in the database design and for the identity extraction in the processing stages.

Extensions can be considered by adding additional sources, for each additional source a post identity processor can be build. Also extracting more information from current sources can be a valuable extension. The extension of the identity model can give users more detailed information. For example an identity can be extended by members like telephone numbers and email addresses. In the front-end view users can get more complete information then. Now only statistics and files where the identity is extracted from are visualized.

With the development of these database extensions, the principles of database normalization by Codd [16] are taken into account. One of the principles is to minimize the data redundancy,

therefore columns which may have identical information and which have not the integer type are separated. Examples are the url_protocols, url_hosts, contact_types and the contact_names tables. Another principle is to minimize the database redesign when extending the database. The database can be extended with a new source for the identities because it is very loosely coupled. To add a new source, just add a mappings table and the other tables only need an ID where an identity can be mapped to. The table schema of the new source does not matter. Furthermore, another principle is to avoid bias towards any particular pattern of querying. If information is clearly separated the database can answer more varied queries.

In every processing unit an extension is made to extract and insert identities. This method is chosen because the data sources are very different and one big identity processing unit is overkill because then file request or parsing must done twice. The specific extraction per file type is well encapsulated in this way. Now when a file is inspected, directly(before closing) the identity processing is done. This will not lead to any performance issues because the functions are very small and fast.

Chapter 5

Identity resolution

This chapter proposes the extended design of the databases and system design to implement the desired design in Tracks Inspector. For the two identity merging processes it is proposed how they can be improved. Finally some conclusions are drawn. This chapter partly answers RQ1: " What data structures need to be developed within the Tracks Inspector, which represents identities and data mappings between source elements and the associated identities?", thus how the identities are stored at the case host level (merged level).

5.1 Design decisions

The identity resolution phase has as main goal to detect identity matches among the multiple evidence units and to merge identical identities. The resolution phase consists of two stages; the first stage is to determine if identities are equal or not.
The methods to determine the equality between strings are discussed in section 2.3. If identities are found to be equal by the string comparison methods, the identities are automatically merged. Two disparate identities are created when identities are not equal.

5.2 Method and system design

This section describes the method and the system design for the automatic identity resolution implemented in Tracks Inspector.

Figure 5.1 visualizes the interaction between the case host and the evidence database host. The interaction calls are numbered in sequential order. The case host has as main target to gather the identities from all evidence units assigned to the specific case and to store all the statistics about the gathered identities. The front-end can ask the case host for a sorted list with identities to display several dashboards or overviews to the user. To process this at execution time takes too long if there are many (huge) evidence units. The following list describes the interactions between the case host and the evidence database host more detailed.

1. gather_identities(eu_id)
   The case host executes a Remote Procedure Call(RPC) and asks the evidence database host. The gather_identities is called with as argument the evidence unit ID. That means that the case host executes for each evidence unit a "gather identities" call.

2. get_idts
   The evidence DB host receives the call and translates it to a SQL query on the database that is associated with the received evidence unit ID. The query asks all existing identities with their specific source statistics. The results are statistics which are stored per identity and indicates where the identity emerged from.

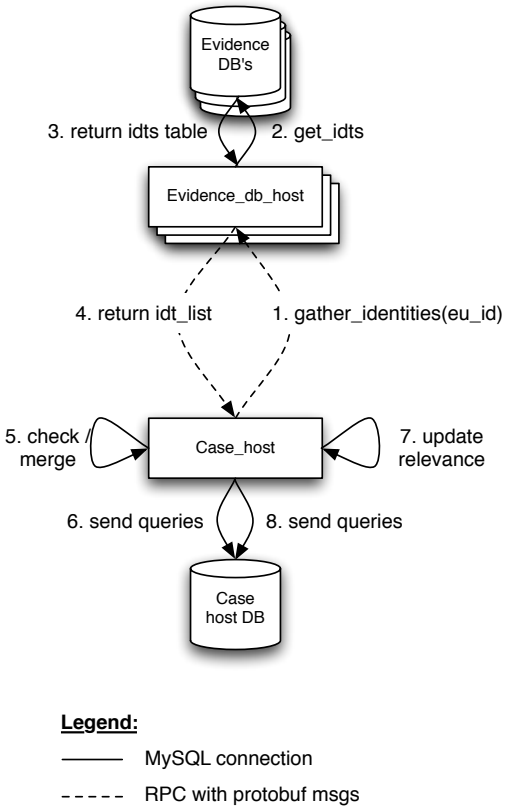

Figure 5.1: TI Case host & Evidence DB identities interaction

3. return idts table
   The database answers the previous query and sends this back to the evidence database host. The results are already sorted by the database by an estimated weighted count.

4. return idts list
   The evidence DB host sends the result of the query to the case host. This message is an identity list message with multiple identities inside.

5. check / merge
   At this stage the identity list of an evidence unit has arrived at the case_host. At this stage the case host loops through all gathered identities. The case host checks if an identity already exists with the same display name in the associated case. The strings are compared with the stringmatching function "semi-exactMatching" proposed in section 2.3.2. If the identity does not exist, the identity is inserted in the case_identities table. If the identity exists or not the statistics of this identity are inserted into the identity_source_statistics table. In this table tuples are stored in the form of (identity_id,evidence_id,source_id,value, value_system_acc_logins). The queries are composed and ready for dispatching.

6. send queries
   The queries to insert the statistics are send to the case host database. The inserting of identities will happen in the stage before(stage 5), because the identity id's are needed to

insert the statistics.

7. update relevance (Relevance determination)
   After gathering the identities the relevances are updated. All identities which are newly inserted or which are updated are looped again. For each identity the statistics are obtained from the identity_source_stats table and the weighted counts are computed. The update queries to update the weighted_counts of the identities are generated.

8. send queries (Relevance determination)
   The update queries for the case_identities table are send to the case host database and will be executed to update the weighted counts of the identities.

The last two steps are not really part of the identity resolution, but more of the relevance determination methods. These steps are discussed in section 6.

For the above mentioned method the database of the case host must be extended. The extended and improved design can be found in appendix A.1. The cases table is the main table of this database. All case information is stored here. The evidence_case_mapping table maps evidence units to a case. The evidence_units table stores some information the case host must know about the evidence units, like the name, comments and the icon. The icon is an icon id that is chosen by the user, when identifying an evidence unit. The evidence_unit_internal_status is a bookkeeping table for the processing progress of an evidence unit. This is needed to know if the identities of an evidence unit are stable. If the processing is done there is no need to gather the identities again of an evidence unit. The case_identities table stores the unique identities per case. Per identity the weighted count is stored to order the identities. Further identities can have a master, which means that the current identity is an alias of another identity. The identity_sources table stores the sources where the identities are extracted from. Sources can be for example: system account, email sent, email received or document author. The identity_source_stats table stores per identity the statistics. The identity_id is the id of the identity in the case_identities table. The evidence_id is the id of the specific evidence unit. The source_id is the source where this statistic row is based on. This source_id is an identifier for a source in the identity_sources table. The value is the statistics value. This is an absolute count of the source, thus the amount of system accounts or the amount of sent emails. The value_system_acc_logins is used in a later stage, the amount is always 0 except if the source is a system account. Then the login count that is parsed is stored here.

5.3 Identity merging methods

This section builds further upon the string comparison methods of section 2.3 and describes the chosen method and the alternatives.

In section 2.3 three different matching algorithms are stated: "exactMatching", "semi-exactMatching" and "possibleMatching". Based on examples the differences among these string matching algorithms are demonstrated. The exactMatching is the most strict comparison method and the possibleMatching more loosely.

In this research project identity merging happens in two different stages. The first stage is in the case host while the identities are gathered from the evidence units and inserted in the case host identity database. For each new insert of an identity, it is checked if the identity already exists. The method used here is the proposed "semi-exactMatching" method. Only identical identities (lowercase comparison without spaces) are merged and also the results of this are combined.

The second stage or method is the manual merging. Users are able to merge identities on case level. In the user interface people can determine the main identity and join the selected aliases.

For the user it seems that the identities are merged. When a user selects the main identity, some alternatives can be proposed in a temporary view. Therefore the usage of the proposed "possibleMatching" method can be used, but probably also extended with some other methods like the Levenshtein distance or the Soundex algorithm. Other software products did not have support for any of these matching or merging algorithms for identities. Most of the software products do not support any type of identity extraction or resolution. In the near future some functionality to determine the most equal identities can be added. Research and practical experience have to show which techniques are best suited for this problem and what method gives the best results of possible aliases.

5.4 Conclusions

The identity resolution implementation in Tracks Inspector was discussed here. The entity relationship diagram of the database in the case host is also proposed. Next to the task of storing the gathered identities it also has the responsibility to cache the identities. If all evidence units are processed it would not be necessary to gather identities from the evidence database hosts, because all information did not change anymore and all information and statistics are stored in the case host database. The flow of gathering identities from the evidence database hosts was also extensively discussed in this chapter.
On the merging of identities also a quick look was taken. The two different stages where and how identities are merged are discussed here. Tests will prove the correctness and the usefulness of the chosen approach.

Chapter 6

Relevance determination

This chapter gives an overview of the methods used for the relevance determination. Research question 3(RQ3) is called: "What statistical method can be used to determine the relevance of identities?". This question is answered in this chapter. The statistical methods are discussed and an example of the current implementation is given.

6.1 Design decisions

The main idea is to develop a method that can determine a relevance count for every identity. The result of this stage is a factor that determines the participation of an identity in a case. The challenges here are to decide when an identity is important or not. There are mainly two options here. The first option is to assume that every evidence unit has the same importance. In this way every evidence unit is equally present in the common identity overview. That means that regardless the size or the amount of items in an evidence unit, the evidence units have the same importance. The second option is to observe the total amount of extracted identities and to determine the relevance of an identity, based on the common total. The second option was chosen in this case because there can be very small evidence units which are less important than large evidence units. For example an SD card with some documents compared with a large hard disk with hundreds of thousands of emails and documents must not have the same participation in the overall identities overview.

The methods which calculate the relevance are discussed in the next section. A weighted method is chosen to determine the relevance. A non-weighted method is probably not an option because all found identities have a different influence and are extracted from different sources. Therefore all sources have a so called weight, which is used to be multiplied by the count of entities found.

6.2 Methods

In this section the statistical methods for computing the relevance of the identities is discussed.

In the current statistic science there are several methods to calculate relevances. In this research the focus lies on the Weighted Sum model [66]. This is a method that evaluates an average sum based on different sources with their corresponding value. Below the definitions are given for this case and the weighted count(one item of the total sum) calculation is proposed in equation 6.1.

Definitions:
S = set with the number of identity occurrences per source
V = set with the number of system account logins per identity
W = set of all sources weights
I = set of all identities
n = # unique sources
i = id of an identity, unique for each identity

k = id of an source, unique for each source
S_{ki} = the number of occurrences for identity i with source k
V_{ki} = the number of system account logins for identity i with source k
W_k = value indicating the weight for source k
I_i = identity with id: i
wc I_i = weighted count of identity i

$$wc\ I_i = \sum_{k=1}^{n} S_{ki} \cdot V_{ki} \cdot W_k \tag{6.1}$$

In the equation the weighted count of an identity(I_i) is calculated. To calculate this, a sum is made of all sources($\sum$). The value, the number of occurrences of this identity in the current source for this identity is get (S_{ki}). This is multiplied by the extra value per source / identity, the number of system account logins(V) for the current source and identity (V_{ki}), this value is now only available for the system account source (the login count is used). For all other sources this value is 1, which has no influence by multiplying it. It will be assumed that if V_{ki} != 1 then S_{ki} = 1. This means that duplicate system accounts can not exist on a single evidence unit. After that the result is multiplied by the associated source weight(W_k), every source has a weight that determines the importance of that source.

To determine the real relevance(in percentages) of an identity the total weighted count is needed of all identities belonging to the specific case. The formula for this calculation is listed in 6.2.

Definitions (extended):
m = # identities (I)
T_{wc} = total of all identities weighted counts in a case

$$T_{wc} = \sum_{i=1}^{m} wc\ I_i \tag{6.2}$$

To make these equations more clear three example tables are listed: table 6.1, table 6.2 and table 6.3. These tables are used to give a practical example of the usage of the equations.

identities (I)		
identity_id (i)	**display_name**	**weighted_count (wc I_i)**
1	John Doe	-
2	Jane Doe	-

Table 6.1: Relevance calculation example - identities table

sources (W)		
source_id (k)	**name**	**weight (W_k)**
1	"System account"	1
2	"Email sent"	5
3	"Email received"	1
4	"Document author"	3
5	"Document last saved"	1
6	"Visited URL"	1
7	"Contact entry"	10
8	"Call/message"	1

Table 6.2: Relevance calculation example - sources table

source statistics (S & V)				
identity_id (i)	**evidence_unit_id**	**source_id (k)**	**value (S_{ki})**	**value_system_acc_logins (V_{ki})**
1	5	1	1	65
1	5	2	10	-
1	5	3	80	-
2	5	1	1	14
2	5	3	100	-

Table 6.3: Relevance calculation example - source statistics table

To determine the weighted_count of the Identity "John Doe"(wc I_1), the sources summation is done. For the first source (system_account) the calculation is as follows: $(S_{ki} \cdot V_{ki} \cdot W_k) = 1 \cdot 65 \cdot 1 = 65$. The second source calculation: $(S_{ki} \cdot V_{ki} \cdot W_k) = 10 \cdot 1 \cdot 5 = 50$. The third source calculation is: $(S_{ki} \cdot V_{ki} \cdot W_k) = 80 \cdot 1 \cdot 1 = 80$. More sources are not available in the source statistics table. The sum of the weighted counts is: 195. That means that the weighted count of this identity is set to 195. The weighted count can be calculated for identity with id 2 on the same manner. The result is: 114. To determine the relevance of identity with id 1, the percentage must be computed by counting the total of the identities weight counts. The total weight is 309. The relevance of I_1 is thus 195 / 309 * 100 = 63,11 %.

6.3 Discussion

The calculations of the weighted counts based on the weighted sum model and subsequently the relevance are discussed. To make it more clear a small example is given how this should work in practice. The practice will tell us if the weight method and the weights are accurate, maybe this must be configured per case. This can probably be future work if there is a need for. There are also methods which can propose weights, these methods are however not chosen. Users of Tracks Inspector probably want to manage the weights by themselves, because they want to configure the most and less important sources.

Chapter 7

User interface

This chapter describes the user interface which is developed to give users of Tracks Inspector a clear view on the identities and to easily manage them. This chapter also discusses the implementation opportunities for RQ 7: "How to decouple sources from an extracted identity and assign them to another identity?".

7.1 Dashboards and views

This section describes the three views Tracks Inspector supports. There is a general view of identities, an analysis view and for each single evidence unit a detail view. Below the views are described in detail.

Next to the extraction and resolution steps, the user interface is also introduced to show the identities overviews. The extracted identities are displayed on case level. The screenshot (figure 7.1) displays an overview of these identities. The identities are sorted by relevance as discussed in the relevance determination chapter(chapter 6). The total count of identities is showed behind the title between the parentheses. The top 20 identities are showed and after scrolling down more results are retrieved from the back-end.

Another overview is the analysis tab in Tracks Inspector, see figure 7.2. This tab shows an matrix with the importances of identities and evidence units. The x-axis represents the identities and the y-axis represents the evidence units. The cells can have four different values or categories: Low, Medium, High and Very high. The values are based on the weighted count of the identities. The minimum and maximum weighted count for that set is computed and between the minimum and maximum the four groups are linear created. The cells with the highest importance shows that the identity is highly involved in the corresponding evidence unit. It is possible to click on cells which result in a small list of statistics. These are the same as in the first general overview, which is described in the previous paragraph.

The identity detail view shows a comprehensive overview of an identity. An example of such an overview can be found in figure 7.3. The statistics where this identity is generated from are shown. In addition the associated evidence units are also listed in the section below. Per evidence unit the count of nodes which are associated are displayed. This gives a clear view of the evidence units where the identity is extracted from. If this identity is a merged identity, the identities where the identities are merged from are also proposed in this view. The original identity can also be viewed. This identity is also the identity which cannot be detached from the merged identity because it acts as the "parent" identity, the parent node for the child identities.

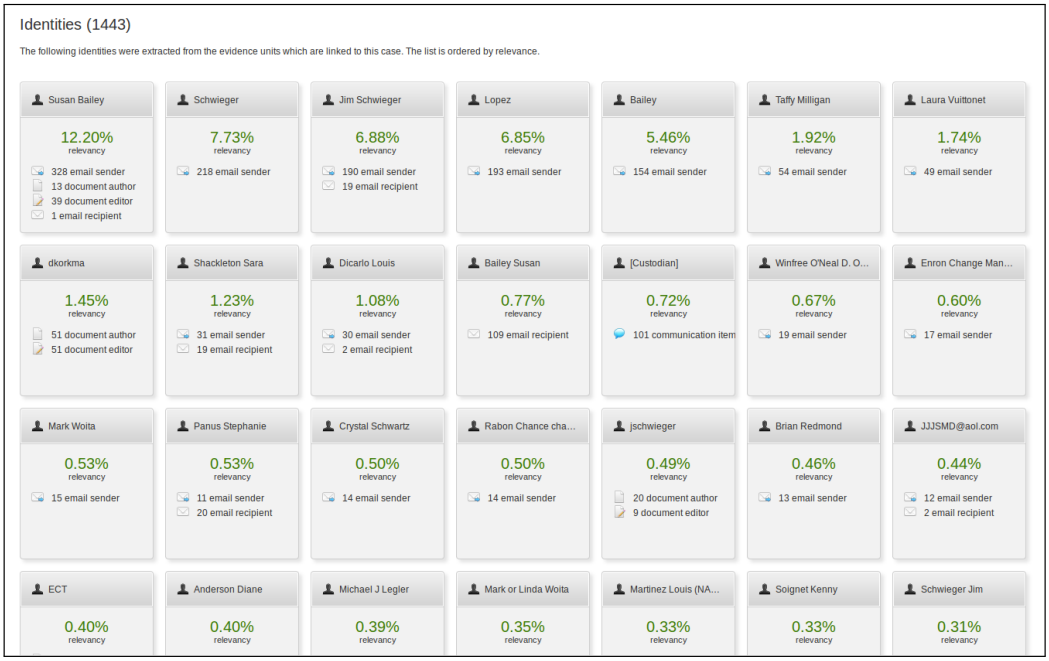

Figure 7.1: TI screen shot - Identities

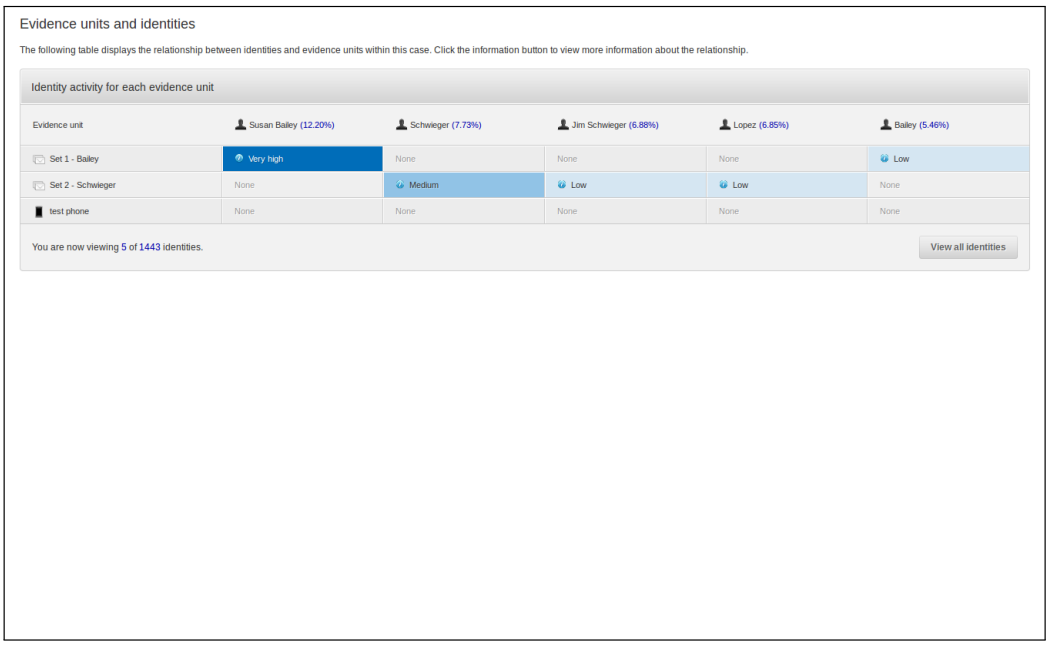

Figure 7.2: TI screen shot - Identities analysis

7.2 User interaction

In this section the possible manual user interaction is explained. The, at time of writing, not implemented functionality for detaching sources is also discussed.

To support investigators in their research, there is a possibility to set an identity as master. Identities can be pinned with such a flag, in Tracks Inspector a "star" icon is used to indicate this. Master identities are always on the top of every identity overview list. It is easy to use this identity in investigators research now because they are always on the top. It is a way to bookmark identities. Off course investigators can unset the master flag for an identity.

When investigators manual identify equal identities they can merge them together. The statistics of merged identities are also merged. An example of a merged identity can be found in figure 7.3. If investigators want to merge identities they first determine the parent identity. In the general overview of identities it is possible to make a selection of identities. The first identity that is selected is used as the parent identity. The merged identity derives the name from the parent identity. All other selected identities become a child of the parent identity.

The functionality for detaching sources from an identity is at the time of writing not implemented. This functionality can be needed if disambiguation appears. Identities in a case are merged if the names are equal. But if there exists two John Doe identities in a case and they refer to different real world persons, they are still merged. To alter this it should be possible to manually create an identity, for example a John Doe2. The ability to detach sources from the identity should be a nice feature. The sources like emails or documents can be detached from the original identity and attached to the new created identity. If this functionality should be implemented the most disambiguation examples can be avoided. This is based on node level, so complete nodes like a complete mail message or a document can be detached. Parts of emails can not be detached, only full nodes.

Tracks Inspector proposes a functionality to filter nodes in the node overviews. In Tracks Inspector these are called facets. An identity facet is also implemented. This can be used to filter nodes on a specified identity. For example if the email overview is loaded, the identity facet proposes the top ten most identities from that specific evidence unit for that node type. The identity John Doe can be selected and all mail messages which contain or are related to the identity John Doe are listed. Investigators can filter easily the important messages related to the identity. The difference with a facet like the sender or recipient is that if a mail message contains an attachment with as author John Doe, the mail message is listed in the overview. In contrast to the sender / recipient facet, the mail message is not listed there or the mail message must contain the identity John Doe in other parts of the mail message.

7.3 Discussion

In this chapter the possibilities to interact with Tracks Inspector on identities level have been described. The most desired functionality is implemented. The functionality for detaching is not implemented because this is currently too detailed and specific for users of Tracks Inspector. The idea on this is discussed above and can be implemented in the future. It should also be possible to detach sources completely from an identity. This results in a pool with nodes that are uncoupled from an identity. They can be coupled to another identity afterwards.

The current functionality gives users of Tracks Inspector more insight in the identities which are involved in the case. Feedback from the field should arrive soon about the usage of the identities. The manual interaction can be adapted if necessary.

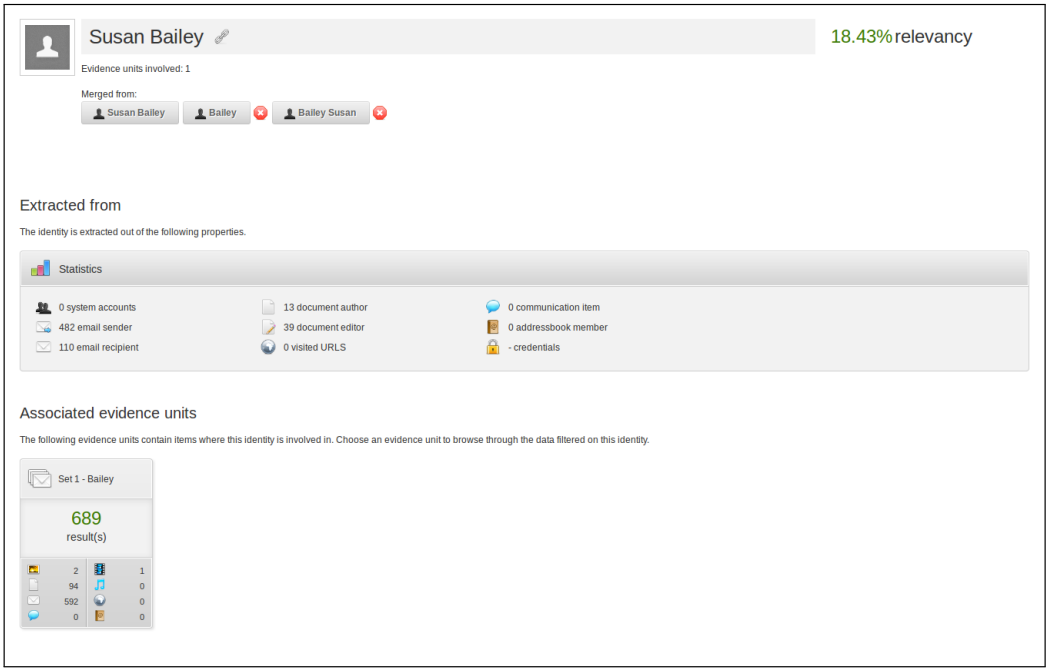

Figure 7.3: TI screen shot - Identity detail view

Chapter 8

Evaluation

8.1 Experimental setup

This section describes the experimental setup. The method and the test data are described in this section included with the experiments that are performed for the evaluation. The main goal is to answer RQ 5 in this section: "How accurate are the results Tracks Inspector delivers compared to other software in this domain?". Also RQ 6 about the scalability of Tracks Inspector is answered here: "How to ensure the scalability of the identities extraction and merging process with respect to the source data?".

8.1.1 Method

To evaluate and to verify the correctness of the implementation, this section describes the method we used to test the implementation. In the experiments we mainly focus on mail archives. This choice is made because there are tools which also support something similar to identity extraction. The commercial tools Clearwell[1] , FTK[2] and Trident[3] can be used to analyze email archive files and to compare the extracted counts. Tracks Inspector can also test non mail archives, but other forensic software did not have real identity extraction functionality. That is why a choice has been made for mail archives in these experiments.

In these experiments both qualitative and quantitative testing are performed. The experiments performed are listed in section 8.1.2. For all experiments we used some predefined datasets as input. These are listed in section 8.1.3. The data used is derived from Enron and from an own company mailbox. The Enron images were available at the Enron Corporation website. In experiment 1 the precision and recall are computed, this has been discussed in section 2.2.4. The main purpose of this experiment is to show how Tracks Inspector performs compared to other commercial products focusing on the amount of identities extracted(quantitative) and the correctness of the found identities(qualitative). All datasets are loaded into the software and the outputs are shown in the results section.

The software products did not serve equal overviews of identities. In each experiment it is demonstrated what type of output we used from the software. The potential differences in the output are also discussed. A more extended description of the differences in outputs is discussed in the validation section 8.1.4. All tests are executed at case level, that means that it is possible to merge multiple evidence units. For all these experiments we assume that there is at most one evidence unit in a case. This implies that there is no identity merging between evidence units. All test data is separately tested, that means that the experiments are apart and not affected by other results.

[1] http://www.clearwellsystems.com/
[2] http://accessdata.com/products/digital-forensics/ftk
[3] http://www.discoverthewave.com/products/107

Next to the qualitative and quantitative identity testing, the scalability of Tracks Inspector with regard to identities is also tested(experiment 3). Two versions of the software are build, one with identities processing enabled and another version with the functionality disabled. In both versions the test data is imported and processed. The execution times are compared between the two versions.

8.1.2 Experiments

EXP 1 - Quantitative - Counting the number of extracted identities
This experiment determines the number of identities that are extracted. The number of identities means the number of unique identities that are extracted from the entire set. The systems that are compared are Clearwell versus Tracks Inspector. Tracks Inspector does not attempt to rebuild Clearwell, it is only a small part of both functionality that is compared. Clearwell only supports the gathering of email, the software does not process attachments if the attachments are not emails. The part of Clearwell that is compared is the participants extraction, the participants names will be used by comparing identities. Tracks Inspector has a counter to determine the total number of identities. In Clearwell a few lists of participants are merged to get the total count of identities. This results in an overview that shows the differences between the software packages based on identity extraction. With that starting point, the precision and recall is computed for each software package. That means that it can be concluded afterwards which software package has the best precision and recall. Before the precision and recall calculation, analysis is done to determine the total set of identities. For the two sets with the least number of identities, the golden reference is manually determined by inspecting the results and the original source data. For the other sets this task is too much work. The list with distinct identities(the golden reference) is manually analyzed on non-relevant identities. These are filtered out and the result is the base reference for this experiment. For this experiment Tracks Inspector is compared to Clearwell only, because this software has some identity support, the others do not support this well. The datasets listed in table 8.1 are used for this experiment. The sets are PST archives and contain email messages.

To determine the precision and recall it is needed to define the relevant and not relevant items in the datasets, this is a manual process. Strings like "None" or "<No recipients>" are manually filtered out and annotated as not relevant. In this experiment the number of identities is determined for all datasets. For two sets also the golden reference is determined. By manually analyzing the results especially from Clearwell, it seems that there are often duplicates. A duplicate is a pair of identities which are not merged. These duplicates are not real duplicates but the items differ mostly on spaces. This seems an error on Clearwell's part, because two identities which only differ on a space at the end of the name seems to be equal. Therefore all white spaces in the names are removed before determining the total set of identities.

EXP 2 - Qualitative - Compare the top 10 most common identities
In this experiment the top ten most identities are compared. First the names are equalized. After that it can be concluded which products mis some identities or which products extract different identities than others. This experiment must lead to a list of future optimizations for Tracks Inspector if the results of Tracks Inspector lack some identities. If not it can be concluded on which points Tracks Inspector performs better than the other email analysis tools. The tools Tracks Inspector, Clearwell and Trident are compared in this experiment. FTK also supports some participants extraction, but there are no possible export options to analyze the output. For Tracks Inspector a relevance determination method is proposed in chapter 6. This has not yet been done for the results of the other tools. For Clearwell and Trident two comparison methods have been used. One with the original output from the tools, that is just a sorted list based on the total count of occurrences of the identity. For the second method a derivative of the identity relevance calculation formula used, thereafter the list is sorted on the relevance count. For determining the top 10 for Clearwell and Trident the following formula is used to calculate the relevance of an identity:

Definitions:
SE = set of all email sender statistic values
SR = set of all email recipient statistic values
I = set of all identities
i = id of an identity, unique for each identity
SE_i = the count of emails where identity i is the sender
SR_i = the count of emails where identity i is a recipient
W_{se} = value indicating the weight for emails where the identity is the sender
W_{sr} = value indicating the weight for emails where the identity is the recipient
I_i = identity with id: i
wc I_i = weighted count of identity i
m = # identities (I)
T_wc = total of all identities weighted counts in a case

Predefined weights:
$W_{se} = 5$
$W_{sr} = 1$

$$wc\, I_i = (SE_i \cdot W_{se}) + (SR_i \cdot W_{sr}) \tag{8.1}$$

$$T_{wc} = \sum_{i=1}^{m} wc\, I_i \tag{8.2}$$

$$Relevance\, I_i = wc\, I_i \,/\, T_{wc} \cdot 100 \tag{8.3}$$

The formulas are analogous to the relevance determination of identities in Tracks Inspector. It is not the same formula because this set did not contain statistics about other sources like documents, system accounts, internet history, address books or communication items. Tracks Inspector supports that, therefore this formula is a part of the formula used in Tracks Inspector(equation 6.1) for the relevance determination. If the identity is the "email sender" in an email, it is multiplied by the weight factor (W_{se}). That means the email sender is five times more relevant than an email recipient, because the weight factor for an email recipient is one(W_{sr}). The weighted count of identity I(wc I_i) as part of the total weighted count(T_{wc}) determines the relevance percentage of the identity(Relevance I_i). The total sum of identity relevances is 100% because the sum of all weighted counts are together equal to the total weighted count.

EXP 3 - Scalability - Tracks Inspectors performance after introducing identities
To test the scalability of Tracks Inspector, some performance tests are executed in this experiment. The execution time of the complete evidence unit processing is measured. Both with and without identities processing is tested to give insight in the execution time of the identities. For these tests the following setup is used: a desktop with Intel Core i3-2100 CPU @3.10GHz x 4 with 8 GiB memory. Every dataset is tested three times and the average execution time is stored. For the largest dataset another more powerful setup is used, it is a machine with 24x Intel(R) Xeon(R) CPU E5645 @2.4GHz and with 80 GiB memory.

EXP 4 - Qualitative - Real forensic case analysis
In this experiment an analysis on a real forensic case is performed. The case consists of 12 evidence units. The owner of the most evidence units is known. With this experiment it will be shown how well Tracks Inspector extracts the associated owners as identities. For this, the ranking on case level of the extracted identities is shown. Further the analysis step of Tracks Inspector shows a matrix of the identities plotted against the (owners of the) evidence units. The matrix shows the degree of how much an identity is involved in an evidence unit. Every cell has a value from

"None" to "Very High". Before this analysis is done, the original forensic researcher of this case lists the most important persons in this case. The processing of this case is done in a distributed environment of five physical machines which all have different tasks. The hosts(see figure C.1) are distributed among the servers.

8.1.3 Test and development data

This section describes the test data. To test the implementation in Tracks Inspector five datasets are stated, which are listed in table 8.1. The first three datasets and the fifth dataset[4] were gathered from the Enron corporation. The table lists for each product the number of emails that are processed by the tool, for Clearwell also the unique emails are shown, because Clearwell supports de-duplication techniques. Mailboxes contain often duplicates, because replied emails contain often the original message which is replied on. Both Clearwell and Tracks Inspector support attachment processing of emails. Clearwell has the functionality to detect duplicates in the total set of emails. All emails which are as attachment extracted and which are already in the dataset are filtered out. Unique emails means that there is no full identical email left in the set of emails.
The fourth dataset is a company mailbox from ourselves. These five mailboxes can be assumed as default mailboxes, because Enron datasets are exports from an old company and these sets are often used for testing purposes [36, 59]. The mailbox from ourself is just an export from a company mailbox without customizing. The sets are kept relatively kept small, else any manual analysis to detect non relevant items would not be possible anymore.

		# emails			
Dataset	**size (MB)**	**TI**	**Clearwell**	**FTK**	**Trident**
Set 1 (Enron Bailey)	26	2178	2178 (1446 unique)	2178	2178
Set 2 (Enron Schwieger)	72	1422	1422 (1018 unique)	1422	1422
Set 3 (Enron Stokley)	120	1353	1348 (1346 unique)	1348	1348
Set 4 (Mailbox Jop Hofste)	125	1109	882 (836 unique)	859	882
Set 5 (Complete Enron set)	8689	498541	517431 (254654 unique)	606758	517406

Table 8.1: Overview of test images (PST files)

8.1.4 Validation

The validation of the experiments is done by precision and recall which are discussed already. The differences in the precision and recall values between the analysis programs are used to draw conclusions about the Tracks Inspector implementation of identities. But also to draw conclusions about the quality of other programs that extract identities.

In the first experiment the number of identities is compared, the differences between the total set of identities and the extracted list by the programs are compared. This results in percentages per dataset for precision and recall.
In the second experiment the correctness of the found identities is validated. To accomplish this dataset "Set 1" is taken. The dataset is used as input for three software products: Tracks Inspector, Clearwell and Trident. The outputs of the programs are compared. The total amount of extracted identities can be very huge. Therefore the focus lies on the top 10 of identities sorted by the identities which have the highest relevance, described in the previous section. The differences between the outputs are analyzed and if it is not clear why some outputs give other results, manual analysis is done to explain this. Off course these results should be the same among the products. No other methods are available to validate the correctness of the found identities.

[4]The Enron data set is gathered from http://enrondata.org/content/data/

The validation of the third experiment is done by comparing the results between the software with identities enabled and the software with the identities disabled. After this comparison it can be concluded if the identities processing takes many processing time or not.

8.2 Results

This section describes the results of the experiments discussed in the previous sections. It shows per experiment the test results and gives insight in the differences between the software products.

8.2.1 Experiment 1 - Quantitative - Counting the number of extracted identities

The results of the first experiment are listed below. The experiment is preprocessed by stripping all the whitespace from the identity names.

Table 8.2 shows the result of the first experiment. The table shows the number of identities extracted by Tracks Inspector and by Clearwell per dataset. It is interesting to see that Clearwell extracts most of the time a lot more identities than Tracks Inspector. To determine the quality below a more relevant comparison is made.

EXP1 - identities extraction

Dataset	#idts TI	#idts CW
Set 1	592	958
Set 2	786	2353
Set 3	370	697
Set 4	151	162
Set 5	20040	85536

Table 8.2: Experiment 1 - Results Tracks Inspector vs. Clearwell

The two datasets with the least number of identities are used for the determination of the precision and recall. The sets 3 and 4 are used here and the results are listed in the tables 8.3 and 8.4. The contents of the second column are identical, because this is common per dataset. The relevant identities are determined manually as described in the method of experiment 1. The third column shows the retrieved identities by the software, this is divided in relevant(R) and non relevant(NR) items. An identity is classified as relevant if the identity refer or is equal to an identity that is part of the golden reference set(column two). If the identity do not refer to an identity in the golden reference set, it is classified as non relevant. If multiple identities refer to the same identity in the golden reference set, only one is counted and the others are non relevant. This is because the identity is already found and an identity can occur once in the golden reference set. The identity may occur at most once in the set of relevant identities that are retrieved by the software. The last columns show the precision and recall based on the second and third columns. Focusing on the results of set 3, the precision of Clearwell is very low. This is because Clearwell extracted many possible identities, but just a small part of them are relevant. These extracted relevant identities are part of the predetermined relevant identities. The precision of Tracks Inspector is much higher. The recall values are both really high, that means that both software packages extract the most relevant identities. The precision results of set 4 differ not much. The recall values lies further apart, Clearwell has a recall value of 100% which means that it found all predefined identities. The most diverse contrast is the precision between Tracks Inspector and Clearwell of set 3. The results are also visualized in graphs. The precision is showed in figure 8.1 and the recall values are showed in 8.2. Tracks Inspectors values are red and the Clearwell values are yellow. These graphs visualize the differences between the products. It can be easy to conclude that Tracks Inspector scores better on the precision values. The recall values do not differ a lot between the software packages.

The experiments result in three interesting tables, which primarily show the differences between Tracks Inspector and Clearwell. The recall of Tracks Inspector and Clearwell are almost the same on the test sets. Analyzing Clearwell's results in more detail shows that there are quite more duplicates, but also identities which do not appear in the results of Tracks Inspector by analyzing these datasets. After analyzing the differences between the results it seems that Clearwell strips all postfixes. For example the original identity "Taylor Mark E (Legal)" appears in Clearwell as "Taylor Mark E" but in Tracks Inspector as "Taylor Mark E (Legal)". Another example is "Thompson, Virginia" what is extracted as "Thompson" in Clearwell, the mention "Virginia" can not be found any more in Clearwell. Clearwell does not only strip punctuation parts at the end, there are also examples where some parts of the surname are removed. Tracks Inspector does not remove these parts. It can be speculated which technique is better, but extracting the original data like Tracks Inspector seems better for the evidence.

EXP1 - Detailed results set 3

Dataset	**# relevant idts**	**#idts retrieved (R/NR)**	**Precision**	**Recall**
Tracks Inspector	208	370 (199 / 171)	53.78 %	95.67 %
Clearwell	208	701 (197 / 504)	28.10 %	94.71 %
		Average:	40.94 %	95.19 %

Table 8.3: Experiment 1 - Detailled results Tracks Inspector set 3

EXP1 - Detailed results set 4

Dataset	**# relevant idts**	**#idts retrieved (R/NR)**	**Precision**	**Recall**
Tracks Inspector	140	151 (131 / 20)	86.75 %	93.57 %
Clearwell	140	162 (140 / 22)	86.42 %	100 %
		Average:	86.59 %	96.79 %

Table 8.4: Experiment 1 - Detailled results Tracks Inspector set 4

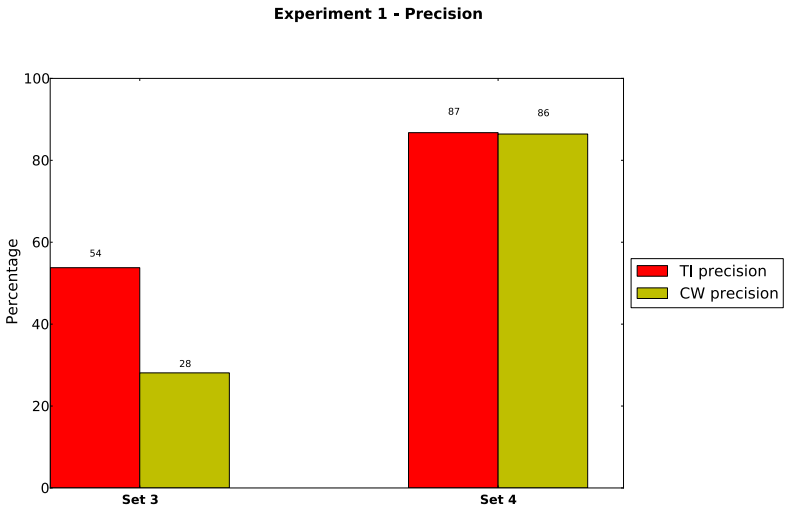

Figure 8.1: Experiment 1 - Precision results

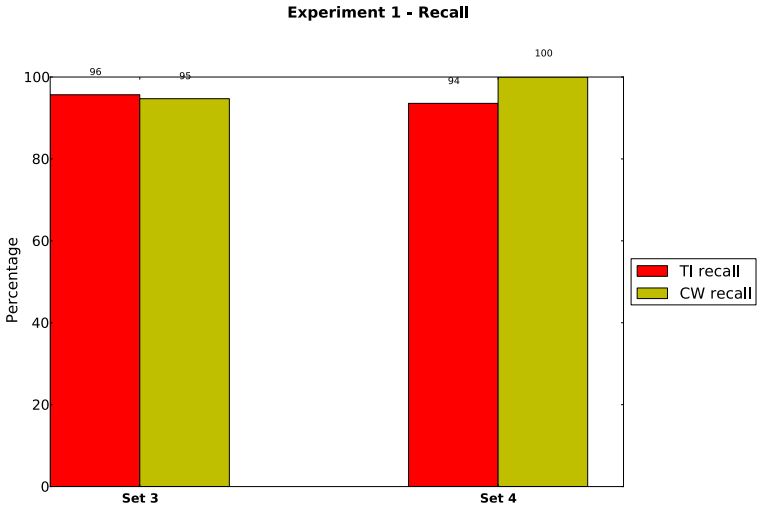

Figure 8.2: Experiment 1 - Recall results

8.2.2 Experiment 2 - Qualitative - Compare the top 10 most common identities

The comprehensive results of this experiment are listed in five tables in appendix B. These tables show the results of the top 10 identities extracted from the test sets by the three software packages. When comparing the tables, similarities arise among the results. Table 8.5 shows a union of all top 10 lists and displays per identity in which results it occurs. The common list of identities consists of 17 identities. There are six identities which occur in all the results. One identity appears in

four results: Taffy Milligan. Another identity appears in three results: Enron General Announcements. The other nine identities appear only in one or two time(s) in the top 10 result. Probably all identities which are part of a top 10 list, and which are not part in another one, can be found in the complete set of identities. The identity "dkorkma" can not be found in another top 10 list than the Tracks Inspector results. This can be inferred from the fact that Tracks Inspector also uses documents for identity extraction. The identity is gathered from document meta data and did not appear in email messages. The identity "dkorkma" was the exchange account name of Deb Korkmas, which was concluded after inspecting the mail archives.

In comparison with experiment 1, experiment 2 focuses on which identities are extracted instead of how much identities are extracted. The results of experiment 2 show that there are many similarities among the results the software produce. Each software package has at least 60 percent similarity in the top 10 results, based on the test data from set 1. Tracks Inspector performs well compared with Clearwell and Trident. The identities which are not found by Tracks Inspector are "Enron General Announcements", "Boyd Samantha", "donay.carmony@ourclub.com", "Heard Marie" and "susan.bailey@enron.com". Almost all of the not found identities are available at Tracks Inspector but not in the top 10. Most of them are part of the complete identities set of Tracks Inspector, but have a lower position outside the top 10. The identity "donay.carmony@ourclub.com" is not found by Tracks Inspector at all. After inspecting the results of Tracks Inspector this seems a bug in Tracks Inspector and was fixed soon. The identities "None" and "<No Recipient>" are also not extracted by Tracks Inspector, but these are examples of non relevant identities. Clearwell for example assumes that every email message has at least one sender and one recipient. If one of these is not available placeholders like "None" or "<No Recipient>" are used and arrive in the output. Probably also the converting of the sender or recipient can not be parsed well and is therefore replaced by a placeholder.

8.2.3 Experiment 3 - Scalability - Tracks Inspectors performance after introducing identities

The results of this experiment are listed in table 8.6. The complete results are derived from the tables in the appendix. Table B.6 and table B.7 contain the original results. The most notable result is the difference of processing time of dataset 3 (Stokley). Without identities it tends to be slower then with identities enabled. After inspecting that result in appendix B.2 the identities processing is probably not the reason, therefore in this table the result is excluded. The differences between the other results are very small, generally the complete identity extraction takes a couple of seconds for small sets and for the large set less than 5% of the total execution time.

The full Enron dataset (Set 5) is processed on another machine because the other setup is less capable for these huge sets. Experiment 3 gives an insight in the processing time of Tracks Inspector with some predefined datasets as input. The differences in execution time are not entirely the same. Dataset 3 has an lower average execution time with identities processing enabled. The individual results show that the execution times diverge a lot. Probably a bug or anomaly in the processing of documents, videos or images ensures the divergent times.

Focusing on the overall results it can be concluded that deviation in execution times between with and without identities processing is very minimal. The processing of identities in these datasets took on average less than five percent of the total execution time, which is plausible. This gives a good indication of the influence of identity extraction in Tracks Inspector.

[5]For the complete Enron set another setup is used: 24x Intel(R) Xeon(R) CPU E5645 @2.4GHz with 80 GiB memory

	Identity name	Tracks Inspector	Clearwell	Clearwell original	Trident	Trident original
1	Susan Bailey	Y	Y	Y	Y	Y
2	Bailey	Y	Y	Y	Y	Y
3	Shackleton Sara	Y	Y	Y	Y	Y
4	Dicarlo Louis	Y	Y	Y	Y	Y
5	Bailey Susan	Y	Y	Y	Y	Y
6	Panus Stephanie	Y	Y	Y	Y	Y
7	Taffy Milligan	Y	Y	-	Y	Y
8	Enron General Announcements	-	Y	-	Y	Y
9	Anderson Diane	Y	-	-	Y	-
10	None	-	Y	Y	-	-
11	<No Recipient>	-	Y	Y	-	-
12	Boyd Samantha	-	-	Y	-	Y
13	dkorkma	Y	-	-	-	-
14	Outlook Migration Team	Y	-	-	-	-
15	dona.carmony@ourclub.com	-	-	-	Y	-
16	Heard Marie	-	-	Y	-	-
17	susan.bailey@enron.com	-	-	-	-	Y

Table 8.5: Experiment 2 - Comparison union of identities

		Average processing time (seconds)	
Dataset	**size (MB)**	**TI without identities**	**TI with identities**
Set 1 (Enron Bailey)	26	762.6	763.0
Set 2 (Enron Schwieger)	72	764.1	768.5
Set 3 (Enron Stokley)	120	-	4157.8
Set 4 (Mailbox Jop Hofste)	125	1111.5	1114.6
Set 5 (Complete Enron set) [5]	8689	23553.5	24653.8

Table 8.6: Experiment 3 - Test results of the performance of Tracks Inspector

8.2.4 Experiment 4 - Qualitative - Real forensic case analysis

This experiment leads to two results. The first result is the overall ranking of the known identities. These results are visualized in table 8.7. All "owners" are also found as identities and the ranking of these identities is relative high. All of them are involved in the top 40 of identities on case level.

The other result is a matrix which can be found in table 8.8. Below this table the associated legend is given. The colored cells consist of values of the identity plotted against its own evidence unit. There is no evidence unit which has an identity which has a higher value than its owner. The ranking is determined by distributing the cell values. The values are determined by the proportion of the cell relative to the entire evidence unit. The maximum and the minimum weighted counts are determined. The space in between is divided into four groups: Low, Medium, High and Very High.

This experiment shows that the the software proposes the most interesting identities. A forensic researcher listed the most relevant identities for this case. Next to the owners of the evidence

units, some other identities were also found, which are partly involved in the case. In addition some identities were not found by Tracks Inspectors identities extraction. After some discussion with the forensic researcher it was concluded that it was impossible to find these identities. Because these identities he only found in the pagefile (a file closely to the physical RAM which stores temporary information) and some phone backups that are not supported by Tracks Inspector till the time of writing.

Owner name / identity	**Rank**
idtA	3
idtB	2
idtC	12
idtD	32
idtE	4
idtF	21
idtG	1
idtH	7
idtI	19
idtJ	11

Table 8.7: Idenities ranking

Evidence unit	**Owner**	**idtA**	**idtB**	**idtC**	**idtD**	**idtE**	**idtF**	**idtG**	**idtH**	**idtI**	**idtJ**
EU 01	idtA	3	1	1	1	1	1	1	-	-	1
EU 02	idtB	1	4	-	1	1	-	1	-	-	1
EU 03	idtC	1	1	1	1	1	1	1	-	1	1
EU 04	idtD	-	-	-	1	-	-	-	-	-	-
EU 05	idtE	1	1	-	1	2	1	1	1	1	1
EU 06	idtF	-	-	-	-	-	-	-	-	-	-
EU 07	-	-	-	-	-	-	-	-	-	-	-
EU 08	-	-	-	-	-	-	-	-	-	-	-
EU 09	idtG	1	1	1	1	1	1	4	-	1	1
EU 10	idtH	-	1	1	-	-	1	-	1	1	1
EU 11	idtI	1	1	-	1	1	1	1	1	1	1
EU 12	idtJ	1	1	1	1	1	1	1	-	1	1

Table 8.8: Heatmap case V

Legend:
- None
1 Low
2 Medium
3 High
4 Very High

Chapter 9

Conclusions and recommendations

This concluding chapter draws the conclusions and gives recommendations. A lot of fields and problems have been discussed in this master thesis. This section draws the conclusions, summarizes the sub research questions and answers the main research question. Also some side remarks on decisions made during this project are made. After that the recommendations chapter lists the recommendations and describes some extensions and improvements for future work.

9.1 Conclusions

The first research question(RQ1) is called: "What data structures need to be developed within Tracks Inspector, which represents identities and data mappings between source elements and the associated identities?". The data structures are developed and especially discussed in chapter 4 and 5. The mappings between the source elements and the identities are stored in so called "mappings" tables. These tables contain the coupling between the identity and the source element where the identity is extracted from. This is used to gather the statistics about the sources per identity. Another structure that is used for the data mapping is de identity PQ table. This table contain information about the node ids (P's) where the identity is involved. Multiple nodes can be covered in an identity, therefore also node ranges can be stored.

Research question 2(RQ2): "How to extract identities from structured fields?" is discussed in chapter 4. A choice has been made for the implementation of post processing methods for each processor. This way the IO usage is minimized compared to a single identity processor where each file must be analyzed through. For each node processor the fields used for translating identities are defined. For example the name of an address book node is used for creating an identity. No interpretation of the name values used for the identity is done. Only empty names are filtered. The identity extraction relies on the output of multiple other parsers, like UFED and XRY . If names are proposed to Tracks Inspector it is possible that there is an identity constructed only from a telephone number, without any name, if the input is wrong. Interpretation of these values are not done here.

The third research question(RQ3) is called: "What statistical method can be used to determine the relevance of identities?" A statistical weighted method is chosen, which uses different predefined weights for each source. Chapter 5 discusses this in more detail. The weights are predefined and can not be changed in the front-end of Tracks Inspector. If it seems that the results are not completely as desired, the weights can be changed in the back-end.

Research question 4(RQ4) is defined as: "Which methods can be used to extract identities from unstructured text out of source elements?". For identities extraction from unstructured text methods like Named Entity Extraction(NEE) and Named Entity Disambiguation(NED) are needed. In chapter 2 a survey is given of the methods which can be used for this type of extraction(NEE) and validation(NED). There is no practical implementation made with this methods in Tracks In-

spector, because the results are often that with some certainty can be determined what an entity is. A found entity can, for example be classified as 60 percent person, 20 percent location and 20 percent misc. It is very difficult for Tracks Inspector to interpret such information. Because when do you assume that a found entity is a person? It would be strange when an identity like "Budapest" or "Amsterdam" appears in the identity list. Therefore this functionality is not implemented in Tracks Inspector at the moment.

The fifth research question(RQ5) reads: "How accurate are the results Tracks Inspector delivers compared to other software in this domain?". The results Tracks Inspector delivers are compared to other digital forensic tools. The evaluation for this is extensively described in chapter 8. The experiments show that Tracks Inspector extracts less identities sometimes compared to Clearwell, but it seems that the results of Clearwell are not always correct. After this experiment it can be concluded that Clearwell sometimes strips a last part of a name. This happens often when the display names contain postfixes where the company name is between parenthesis, also it just aborts sometimes after some words and destroys the last part. It can be concluded because there was a company mailbox involved that is manually analyzed. In this evaluation the focus lies on extracting identities which can refer to real world persons. A company mailbox is used, to verify the identities which refer to real world persons.

Research question 6(RQ6) is called: "How to ensure the scalability of the identities extraction and merging process with respect to the source data?". The scalability is tested in section 8. The execution times are compared between identities processing enabled and identities processing disabled. It seems that the identity processing took a very small part of the execution time, less than 1 percent. The scalability of the identities extraction seems very well.

The last research question (RQ7): "How to decouple sources from an extracted identity and assign them to another identity?" is discussed in chapter 7. It is at time of writing not implemented but it can be a nice feature. There must be devised if the original links between the sources and the identity must be saved, so that these action can be reverted when users detach sources from an identity. Users would be able to create manual identities. The mapping between a source element and an identity could be changed by the user. This seems easy enough to implement, but it would be too difficult to understand for arbitrary investigators.

The main research question is: "How to extract, merge and rank identities from heterogeneous sources containing personal information, to propose a valuable identities overview?". This master project describes the research towards identity extraction and identity resolution. It also describes the implementation of identity extraction and identity resolution in Tracks Inspector. The sources where identities are extracted from are described. In this research the focus lies on identity extraction from structured data instead of unstructured data. As told TI's node processors are extended to support identity extraction, which result in several sources where identities are extracted from. The identities overview is available on case level, the identities extracted on evidence unit level are gathered and equal items are merged. The ranking of identities is also described in detail, to propose an valuable identities overview to the users of Tracks Inspector. Investigators can now start from this perspective with their investigation.

This research gives a contribution in the world of digital forensic tools. There are no tools which support extracting and ranking of identities based on these many sources. Clearwell is a software package that supports identities based on the participants of emails, this functionality is very minimal compared to Tracks Inspector. The second contribution is to show an approach of extracting identities from heterogeneous sources. There are no software packages which support similar functionality. Also proposed, is a method to assist users by merging identities with relevance feedback and to manage identities by bookmarking identities as master identities. The overall contribution is to make it easier for investigators to encounter the most relevant identities in a case, which will result in faster and more efficient investigations.

9.2 Recommendations

In this section the recommendations and the future work is described. This can be used for extending this research or to implement more functionality in Tracks Inspector with regard to identities.

To extend the functionality in Tracks Inspector it is possible to implement the proposed relevance feedback method along with some string comparison algorithms like SoundEx. This functionality would support users of Tracks Inspector when they want to merge identities. If users select an identity in the identity overview dashboard, the list with other identities must be sorted on the most corresponding identities. The feedback users give is stored and can be used to give users more relevant research. If for example it seems that users hardly ever use identities with very short names, these identities can then be moved down the sorted list.

Detaching sources from identities could be another nice feature. So disambiguation can be avoided and the results Tracks Inspector delivers will become more accurate. This functionality will probably not be used a lot, but when people with the same name exist in a specific case, it is a valuable feature.

One of the first recommendations is to support different sources. As discussed in the requirements chapter(ch. 3) the support for cookies, user ID's of the NTFS file ownership and more information from the registry is not supported yet. This information can be added in a relative short time. Especially the support for cookies would be a valuable extension of Tracks Inspector. The usernames and aliases used for multiple websites can be gathered from this source. The user ID's should be interesting when there are multiple users involved in a system. User ID's and group ID's can be used to relate files to an specific system account. These files can also be added to the corresponding identity. The registry contains multiple values about when system users login can be gathered, this information can be added to the system user information.

In the digital forensic world nowadays timelines are designed in investigations. A timeline can assist users to propose a view of what a suspect has done in the course of time. This could be a nice feature in Tracks Inspector. The recommendation is to make a timeline in the identity detailview. All timestamped information of that identity is sticked to the timeline. If login information is parsed from the registry this can also be added to the timeline. The activity of the identity in the whole case can be proposed now. If there are for example two evidence units involved in a case, a notebook and a mobile phone, the composite overview can be generated of the activity of the user. The opportunity is available to implement this in Tracks Inspector, because there is much data available at the moment about activity of an user. Conversations from a mobile phone are parsed already and have a timestamp. This information can be used as starting point when implementing this functionality.

It is possible to stay under the radar as suspect. When the suspect uses a different alias for each email, for example JohnDoe1 and the next time he send an email he used JohnDoe2, it is possible that all these identities become not relevant. Because all these identities have less sources. There are multiple ways to use different aliases, but a search like function, as proposed in the relevance feedback some paragraphs before should solve the biggest problem. For people who use totally different aliases each time another method should be devised.

The problem which is already noted in the problem exploration section, about an identity which can refer to multiple real world persons. For example if two persons with the name "John Doe" exists in an environment. Bhattacharya proposes a method called collective entity resolution. He uses clustering techniques and the Gibbs sampling techniques, what discovers the most relevant identities. These techniques can be more explored and are maybe an interesting complementation for Tracks Inspectors identity resolution [5].

List of Figures

List of Tables

Bibliography

[1] W. Alink, RAF Bhoedjang, P.A. Boncz, and A.P. de Vries. Xiraf–xml-based indexing and querying for digital forensics. *digital investigation*, 3:50–58, 2006.

[2] S. Anson and S. Bunting. *Mastering Windows network forensics and investigation*. Sybex, 2007.

[3] Net Applications.com. Desktop operating system market share. Website, February 2012. http://marketshare.hitslink.com/operating-system-market-share.aspx?qprid=8&qpcustomd=0.

[4] ArxSys. Open source digital investigation framework. http://www.digital-forensic.org/. Last visited April 2012.

[5] I. Bhattacharya and L. Getoor. Collective entity resolution in relational data. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):5, 2007.

[6] S. Bockting, MJ Ooms, D. Hiemstra, PE Van Der Vet, and TWC Huibers. Evaluating relevance feedback: An image retrieval interface for children. 2008.

[7] L. Boldareva, D. Hiemstra, and W. Jonker. A scalable and efficient content-based multimedia retrieval system. In M.F. Moens, R. de Busser, D. Hiemstra, and W. Kraaij, editors, *Proceedings of the Third Dutch-Belgian Information Retrieval Workshop (DIR 2002)*, pages 28–37, Leuven, Belgium, December 2002. KU Leuven. Imported from EWI/DB PMS [db-utwente:inpr:0000003403], http://www.law.kuleuven.ac.be/icri/seminars.php?id=7.

[8] A. Borthwick. *A maximum entropy approach to named entity recognition*. PhD thesis, New York University, 1999.

[9] S. Bunting and W. Wei. *EnCase computer forensics: the official EnCE: EnCase certified examiner study guide*. Sybex, 2006.

[10] JL Camp. Digital identity. *Technology and Society Magazine, IEEE*, 23(3):34–41, 2004.

[11] C. Carpineto, G. Romano, and V. Giannini. Improving retrieval feedback with multiple term-ranking function combination. *ACM Transactions on Information Systems (TOIS)*, 20(3):259–290, 2002.

[12] X. Carreras, L. Marquez, and L. Padro. Named entity extraction using adaboost. In *proceedings of the 6th conference on Natural language learning-Volume 20*, pages 1–4. Association for Computational Linguistics, 2002.

[13] B. Carrier. *File system forensic analysis*. Addison-Wesley Professional, 2005.

[14] Harlan Carvey. *Windows Forensic Analysis*. Syngress, 2009.

[15] P. Christen. A comparison of personal name matching: Techniques and practical issues. In *Data Mining Workshops, 2006. ICDM Workshops 2006. Sixth IEEE International Conference on*, pages 290–294. IEEE, 2006.

[16] E.F. Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 13(6):377–387, 1970.

[17] W.W. Cohen, P. Ravikumar, and S.E. Fienberg. A comparison of string distance metrics for name-matching tasks. In *IIWeb*, pages 73–78, 2003.

[18] W.W. Cohen and S. Sarawagi. Exploiting dictionaries in named entity extraction: combining semi-markov extraction processes and data integration methods. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 89–98. ACM, 2004.

[19] W.W. Cohen, R.E. Schapire, and Y. Singer. Learning to order things. *J Artif Intell Res*, 10:243–270, 1999.

[20] S. Cucerzan. Large-scale named entity disambiguation based on wikipedia data. In *Proceedings of EMNLP-CoNLL*, volume 2007, pages 708–716, 2007.

[21] H. Cunningham. Gate: an architecture for development of robust hlt applications hamish cunningham, diana maynard, kalina bontcheva, valentin tablan department of computer science university of sheffield. 2002.

[22] Access Data. Forensic toolkit 4 whitepaper. http://accessdata.com/downloads/media/FTK_DataSheet_web.pdf. Last visited April 2012.

[23] M. Debbabi, F. Hassaïne, Y. Jarraya, A. Soeanu, and L. Alawneh. *Verification and Validation in Systems Engineering: Assessing UML/SysML Design Models*. Springer, 2010.

[24] A.K. Elmagarmid, P.G. Ipeirotis, and V.S. Verykios. Duplicate record detection: A survey. *Knowledge and Data Engineering, IEEE Transactions on*, 19(1):1–16, 2007.

[25] W. Fan, M. Luo, L. Wang, W. Xi, and E.A. Fox. Tuning before feedback: combining ranking discovery and blind feedback for robust retrieval. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 138–145. ACM, 2004.

[26] S.L. Garfinkel. Digital forensics research: The next 10 years. *digital investigation*, 7:S64–S73, 2010.

[27] D. Garlan and M. Shaw. An introduction to software architecture. *Advances in software engineering and knowledge engineering*, 1:1–40, 1993.

[28] Inc Geeknet. Oyster entity resolution. Website, 2012. http://sourceforge.net/p/oysterer/wiki/Welcome.

[29] M. B. Habib and M. van Keulen. Information extraction, data integration, and uncertain data management: The state of the art. Technical Report TR-CTIT-11-06, Centre for Telematics and Information Technology University of Twente, Enschede, 2011.

[30] J. Han, M. Kamber, and J. Pei. *Data mining: concepts and techniques*. Morgan Kaufmann, 2011.

[31] T. Heath and C. Bizer. Linked data: Evolving the web into a global data space. *Synthesis Lectures on the Semantic Web: Theory and Technology*, 1(1):1–136, 2011.

[32] H. Henseler. Network-based filtering for large email collections in e-discovery. *Artificial Intelligence and Law*, 18(4):413–430, 2010.

[33] H. Hibshi, T. Vidas, and L. Cranor. Usability of forensics tools: a user study. In *IT Security Incident Management and IT Forensics (IMF), 2011 Sixth International Conference on*, pages 81–91. IEEE, 2011.

[34] J. Hoffart, M.A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 782–792. Association for Computational Linguistics, 2011.

[35] F. Iqbal, R. Hadjidj, B. Fung, and M. Debbabi. A novel approach of mining write-prints for authorship attribution in e-mail forensics. *digital investigation*, 5:S42–S51, 2008.

[36] B. Klimt and Y. Yang. Introducing the enron corpus. *Machine Learning*, 2004.

[37] A.H.F. Laender, B.A. Ribeiro-Neto, A.S. Da Silva, and J.S. Teixeira. A brief survey of web data extraction tools. *ACM Sigmod Record*, 31(2):84–93, 2002.

[38] J. Lafferty, A. McCallum, and F.C.N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

[39] V.I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710, 1966.

[40] D. Loshin. *Master data management*. Morgan Kaufmann, 2009.

[41] J. Makhoul, F. Kubala, R. Schwartz, and R. Weischedel. Performance measures for information extraction. In *Proceedings of DARPA Broadcast News Workshop*, pages 249–252, 1999.

[42] C.D. Manning, P. Raghavan, and H. Schutze. *Introduction to information retrieval*, volume 1. Cambridge University Press Cambridge, 2008.

[43] D. Manson, A. Carlin, S. Ramos, A. Gyger, M. Kaufman, and J. Treichelt. Is the open way a better way? digital forensics using open source tools. In *System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on*, pages 266b–266b. IEEE, 2007.

[44] A. McCallum, D. Freitag, and F. Pereira. Maximum entropy markov models for information extraction and segmentation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 591–598, 2000.

[45] A. McCallum and W. Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 188–191. Association for Computational Linguistics, 2003.

[46] A. Mikheev, M. Moens, and C. Grover. Named entity recognition without gazetteers. In *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*, pages 1–8. Association for Computational Linguistics, 1999.

[47] D. Milne and I.H. Witten. An open-source toolkit for mining wikipedia. In *Proc. New Zealand Computer Science Research Student Conf*, volume 9, 2009.

[48] E. Minkov, R.C. Wang, and W.W. Cohen. Extracting personal names from email: Applying named entity recognition to informal text. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 443–450. Association for Computational Linguistics, 2005.

[49] R. Mitkov. *Anaphora resolution*, volume 134. Longman London, 2002.

[50] M.F. Moens. *Information extraction: algorithms and prospects in a retrieval context*, volume 21. Springer-Verlag New York Inc, 2006.

[51] Alvaro Monge and Charles Elkan. The field matching problem: Algorithms and applications. In *In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 267–270, 1996.

[52] Pipl.com. Pipl - people search. Website, 2012. http://pipl.com/help/identity-resolution/.

[53] B. Pouliquen, R. Steinberger, C. Ignat, I. Temnikova, A. Widiger, W. Zaghouani, and J. Zizka. Multilingual person name recognition and transliteration. *Arxiv preprint cs/0609051*, 2006.

[54] L.R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[55] V. Roussev and G.G. Richard III. Breaking the performance wall: The case for distributed digital forensics. In *Proceedings of the 2004 digital forensics research workshop (DFRWS 2004)*, 2004.

[56] Y. Rui, T.S. Huang, M. Ortega, and S. Mehrotra. Relevance feedback: A power tool for interactive content-based image retrieval. *Circuits and Systems for Video Technology, IEEE Transactions on*, 8(5):644–655, 1998.

[57] R. Russell and M. Odell. Soundex. *US Patent*, 1, 1918.

[58] S. Sarawagi. Information extraction. *Foundations and trends in databases*, 1(3):261–377, 2008.

[59] J. Shetty and J. Adibi. The enron email dataset database schema and brief statistical report. *Information Sciences Institute Technical Report, University of Southern California*, 4, 2004.

[60] C. Snae. A comparison and analysis of name matching algorithms. *International Journal of Applied Science. Engineering and Technology*, 4(1):252–257, 2007.

[61] S. Soderland. Learning information extraction rules for semi-structured and free text. *Machine learning*, 34(1):233–272, 1999.

[62] S.J. Stolfo and S. Hershkop. Email mining toolkit supporting law enforcement forensic analyses. In *Proceedings of the 2005 national conference on Digital government research*, pages 221–222. Digital Government Society of North America, 2005.

[63] F.M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM, 2007.

[64] J. Talburt. *Entity resolution and information quality*. Morgan Kaufmann Pub, 2010.

[65] Erik F. Tjong Kim Sang. Introduction to the conll-2002 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2002*, pages 155–158. Taipei, Taiwan, 2002.

[66] E. Triantaphyllou. *Multi-criteria decision making methods: a comparative study*, volume 11. Kluwer Academic Publishers Dordrecht, 2000.

[67] E. Ukkonen. Approximate string-matching with q-grams and maximal matches. *Theoretical Computer Science*, 92(1):191–211, 1992.

[68] I. Veldman. Matching profiles from social network sites. Master's thesis, University of Twente, October 2009.

[69] R.W. White, J.M. Jose, and I. Ruthven. Comparing explicit and implicit feedback techniques for web retrieval: Trec-10 interactive track report. In *Proceedings of the Tenth Text Retrieval Conference (TREC-10)*, pages 534–538. NIST, 2002.

[70] Craig Wilson. Netanalysis forensic internet hisory analysis, 2009.

[71] W.E. Winkler. *Matching and record linkage*. Wiley Online Library, 1993.

Appendix A

ERD Diagrams

A.1 Case host Database ERD

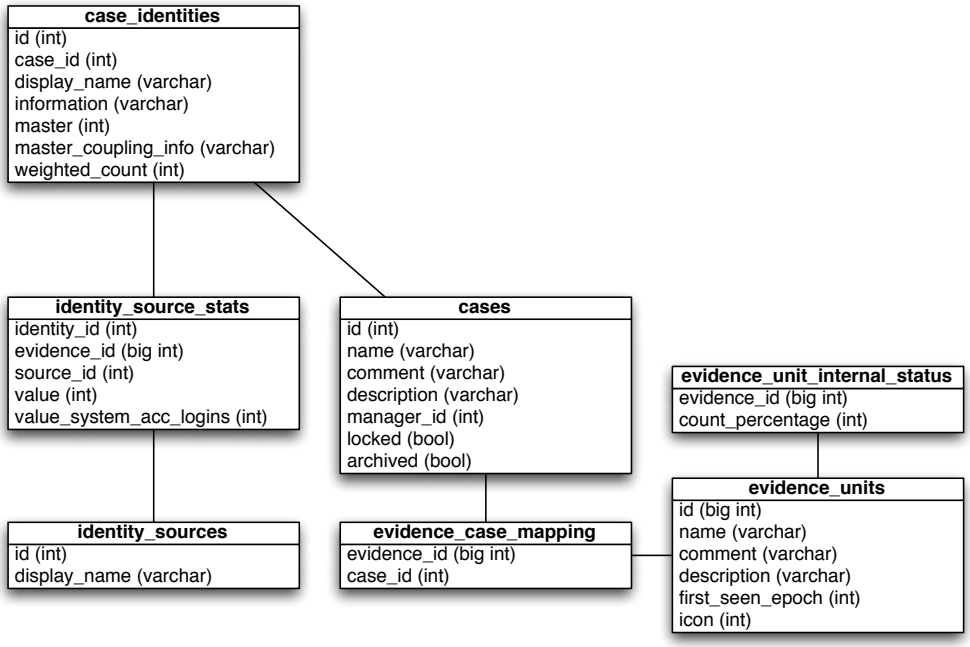

Figure A.1: TI Case host DB - Identities related tables ERD

A.2 Evidence Database ERD

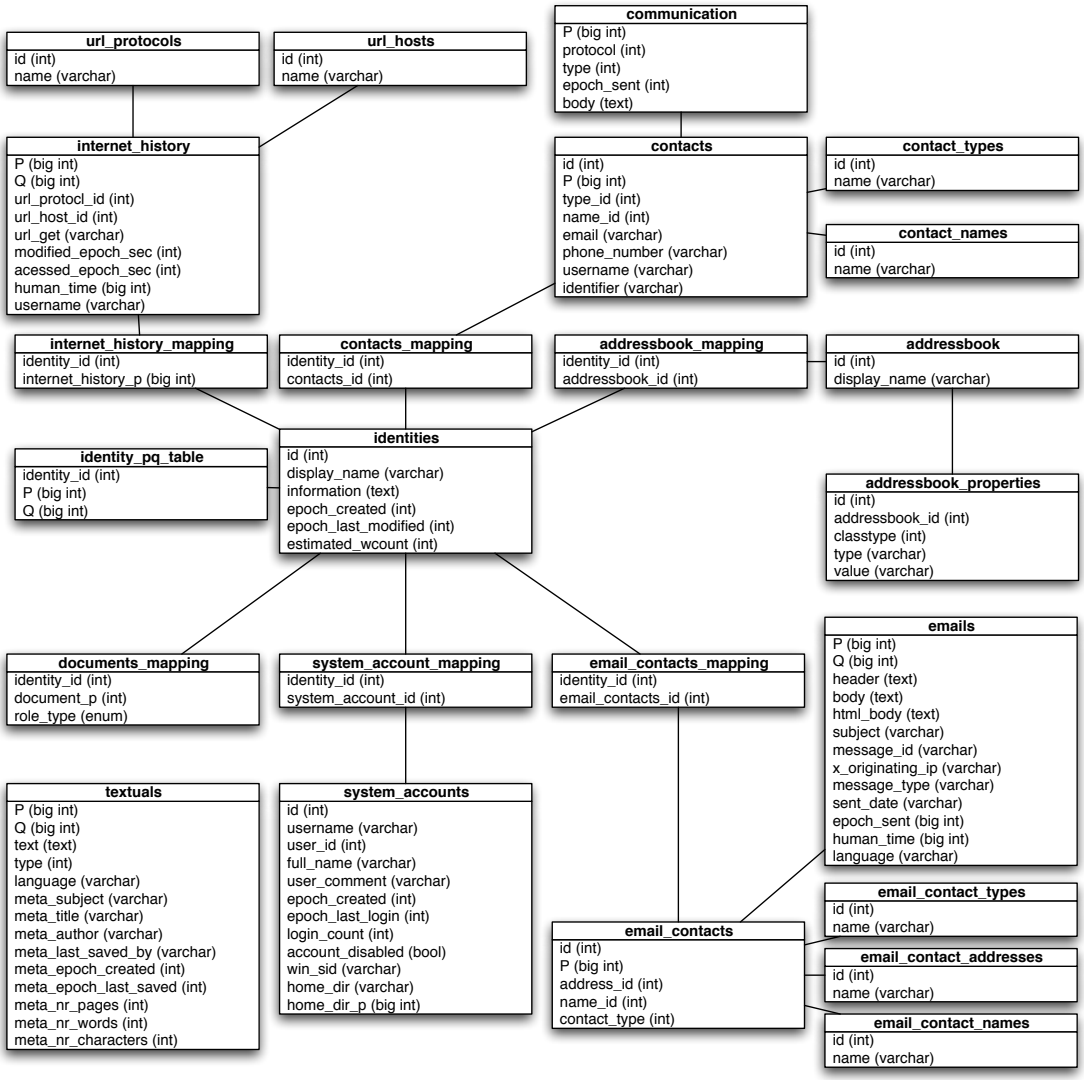

Figure A.2: TI Evidence DB - Identities related tables ERD

Appendix B

Experiment results

B.1 Results experiment 2

This section lists the comprehensive results of experiment 2. The last column of each table shows the relevance. The relevance determination of Tracks Inspector is discussed in chapter 6. For the other software packages(Trident and Clearwell) the determination of relevance is explained in section 8.1.2. Tracks Inspector based their relevance also on other sources than the email sender and email recipient.

	Identity name	system accounts	email sender	email recipient	doc author	doc editor	visited URLS	relevance
1	Susan Bailey	0	328	1	13	39	0	29.83 %
2	Bailey	0	154	0	0	0	0	13.36 %
3	Taffy Milligan	0	54	0	0	0	0	4.69 %
4	dkorkma	0	0	0	51	51	0	3.54 %
5	Shackleton Sara	0	31	19	0	0	0	3.02 %
6	Dicarlo Louis	0	30	2	0	0	0	2.64 %
7	Bailey Susan	0	0	109	0	0	0	1.89 %
8	Panus Stephanie	0	11	20	0	0	0	1.30 %
9	Anderson Diane	0	9	11	0	0	0	0.97 %
10	Outlook Migration Team	0	8	0	0	0	0	0.69 %

Table B.1: Tracks Inspector results - Set 1 (Enron Bailey) - Top 10 identities results

	Identity name	email sender	email recipient	relevance
1	None	748	0	32.21 %
2	Susan Bailey	191	7	8.29 %
3	<No Recipients>	0	920	7.92 %
4	Bailey	148	1	6.38 %
5	Bailey Susan	0	261	2.25 %
6	Shackleton Sara	31	53	1.79 %
7	Dicarlo Louis	30	16	1.43 %
8	Taffy Milligan	32	1	1.39 %
9	Enron General Announcements	26	0	1.12 %
10	Panus Stephanie	14	57	1.09 %

Table B.2: Clearwell results - Set 1 (Enron Bailey) - Top 10 identities results

	Identity name	email sender	email recipient	relevance
1	<No Recipients>	0	920	15.79 %
2	None	748	0	12.84 %
3	Bailey Susan	0	261	4.48 %
4	Susan Bailey	191	7	3.40 %
5	Bailey	148	1	2.56 %
6	Shackleton Sara	31	53	1.44 %
7	Panus Stephanie	14	57	1.22 %
8	Boyd Samantha	1	50	0.88 %
9	Dicarlo Louis	30	16	0.79 %
10	Heard Marie	7	29	0.62 %

Table B.3: Clearwell original results - Set 1 (Enron Bailey) - Top 10 identities results

	Identity name	email sender	email recipient	relevance
1	susan bailey	328	14	36.13 %
2	bailey	154	0	16.82 %
3	taffy milligan	54	0	5.90 %
4	shackleton sara	31	32	4.08 %
5	bailey susan	0	172	3.76 %
6	dicarlo louis	30	0	3.28 %
7	enron general announcements	26	0	2.84 %
8	panus stephanie	14	46	2.53 %
9	anderson diane	11	0	1.20 %
10	dona.carmony@ourclub.com	10	0	1.09 %

Table B.4: Trident results - Set 1 (Enron Bailey) - Top 10 identities results

	Identity name	email sender	email recipient	relevance
1	susan bailey	328	14	26.35 %
2	bailey susan	0	172	13.25 %
3	bailey	154	0	11.86 %
4	shackleton sara	31	32	4.85 %
5	panus stephanie	14	46	4.62 %
6	taffy milligan	54	0	4.16 %
7	susan.bailey@enron.com	0	32	2.47 %
8	dicarlo louis	30	0	2.31 %
9	boyd samantha	0	30	2.31 %
10	enron general announcements	26	0	2.00 %

Table B.5: Trident original results - Set 1 (Enron Bailey) - Top 10 identities results

B.2 Results experiment 3

This section list the original results of experiment 3. Each data set is three times tested. All measurements are prefixed with an m.

		TI without identities processing time (seconds)			
Data set	**size (MB)**	**m1**	**m2**	**m3**	**avg**
Set 1 (Enron Bailey)	26	769.2	758.7	760.0	762.6
Set 2 (Enron Schwieger)	72	760.2	766.6	765.5	764.1
Set 3 (Enron Stokley)	120	7498.2	4234.7	7548.1	6427.0
Set 4 (Mailbox Jop Hofste)	125	1138.5	1106.0	1090.0	1111.5
Set 5 (Complete Enron set) [1]	8689	23198.5	23812.7	23649.3	23553.5

Table B.6: Test results of the performance of Tracks Inspector (without Identities)

		TI with identities processing time (seconds)			
Data set	**size (MB)**	**m1**	**m2**	**m3**	**avg**
Set 1 (Enron Bailey)	26	761.3	763.8	763.8	763.0
Set 2 (Enron Schwieger)	72	772.5	765.7	767.3	768.5
Set 3 (Enron Stokley)	120	3943.3	4231.3	4298.8	4157.8
Set 4 (Mailbox Jop Hofste)	125	1140.2	1103.2	1100.4	1114.6
Set 5 (Complete Enron set) [1]	8689	24526.7	24070.3	25364.5	24653.8

Table B.7: Test results of the performance of Tracks Inspector (with Identities)

[1] For the complete Enron set another setup is used: 24x Intel(R) Xeon(R) CPU E5645 @2.4GHz with 80 GiB memory