2007

An Interface for Collaborative Digital Forensics

Rajarshi Das
THE FLORIDA STATE UNIVERSITY
COLLEGE OF ARTS AND SCIENCES

AN INTERFACE FOR COLLABORATIVE DIGITAL FORENSICS

By

RAJARSHI DAS

A Thesis submitted to the
Department of Computer Science
in partial fulfillment of the
requirements for the degree of
Master of Science

Degree Awarded:
Fall Semester, 2007
The members of the Committee approve the Thesis of Rajarshi Das defended on September 19, 2007.

Sudhir Aggarwal  
Professor Directing Thesis

Breno de Medeiros  
Committee Member

Zhenhai Duan  
Committee Member

The Office of Graduate Studies has verified and approved the above named committee members.
To Ma, Baba and Didi
My Family...
I would like to take this opportunity to thank the members of my thesis committee for their help and support.

I thank my Advisor/Major Professor, Dr. Sudhir Aggarwal, for having the faith in me and appointing me as his Research Assistant since Fall 2005. It was his constant guidance, encouragement, and support that proved invaluable during my graduate studies and research here at Florida State University. I truly appreciate the independence given to me by him to explore the possibilities in my area of research, and his direction that helped me in the work done in this thesis. I am grateful for being given the opportunity to present our research at a conference where our work was accepted and published.

Dr. Breno de Medeiros has been a source of great inspiration both as an instructor of the Network Security course that I took in Fall 2005, as well as a security expert in our research group. His knowledge of security, both theoretical and applied, was essential to the work done related to security in this thesis.

Dr. Zhenhai Duan’s research interest is broadly in computer networks with one of his research focus being on networking security. I thank him for evaluating this thesis and offering suggestions for improving it. I consider myself fortunate for having him on my thesis committee.

I would also like to thank my good friend and erstwhile research partner, Daniel Beech, whose passion to work hard and produce good work rubbed off on me. I enjoyed working with him and some of the material presented here in this thesis is based on shared work with him.

Next, I would like to thank my current research partner, Edson Manners, for his help in the work done in this thesis. He was responsible for publishing and maintaining our software and user guide on SourceForge.net. I would also like to thank Leonardo Skipper Kermes (Leo) for all the support provided for setting up the hardware and software.
I thank Mr. Eric Thompson, Founder and CTO of AccessData® for giving us free access to his commercial digital forensic tools for this research. His wonderful ideas and suggestions were instrumental in the design and development of the DNA Project funded by the National Institute of Justice under grant 2005-DE-BX-K034.

I remember my family - my Mother, Father, Sister and brother-in-law, whose inspiration and support made my graduate studies at Florida State University possible. Last but not least, I thank Sambuddha, Nabanita, Jasbinder, Rana, Rupesh, Chandrima, Mukesh, Sanhita and Bodhana for being a friend during some of my most difficult of times.
# TABLE OF CONTENTS

List of Figures ................................................................. viii  
Abstract ................................................................. ix  

1. INTRODUCTION .......................................................... 1  
   1.1 Contributions ......................................................... 2  
   1.2 Summary ......................................................... 2  

2. Background ........................................................... 4  
   2.1 Current Practices .................................................. 4  
   2.2 Collaborative Digital Forensics and an Interface .............. 5  
   2.3 The need for Digital Forensic Models ......................... 6  
   2.4 Summary ......................................................... 6  

3. Purpose and Objectives ................................................ 7  

4. System Design .......................................................... 10  
   4.1 The Web Component ............................................... 11  
   4.2 User Authentication ............................................. 13  
   4.3 Features and Supporting Components ......................... 15  

5. The Session Server .................................................... 20  

6. Integration with the AccessData decryption tools ................. 24  
   6.1 File Management in the DNA silo ........................... 24  

7. Requirements and Constraints ......................................... 29  
   7.1 Interfacing with the Digital Decryption Tools ............... 29  
   7.2 Job Management Process ...................................... 29  
   7.3 Extensibility and Portability .................................. 30  
   7.4 Security Constraints ........................................... 32  

8. Passphrase Research .................................................. 34  
   8.1 Challenges in Passphrase Breaking .......................... 34  
   8.2 Research Activities ............................................ 36  

9. Conclusions and Future Work .......................................... 39
## LIST OF FIGURES

3.1 The DNA™ Architecture. ................................................. 9
4.1 Screenshot of the Interface. ............................................. 14
4.2 User Account Organized as cases. ................................. 19
5.1 System Design showing all Components. ...................... 23
6.1 The AccessData Rainbow Tables interface and configuration window. 25
6.2 Processing of a Job in DNAOnline. ............................... 28
7.1 An overview of the implementation desired. .................. 31
7.2 Securing the Decryption Tool Host. ............................... 33
8.1 A Sentence (S) formed as a tree of Noun Phrase (NP) and Verb Phrase (VP) nodes. Verb node denoted by $V$ and Noun denoted by $N$ ............ 37
B.1 The DNAOnline login screen. ........................................ 43
B.2 The main menu displayed after user login. ...................... 44
B.3 The first step in creating a new case. ............................ 45
B.4 The second step in creating a new case (Auxiliary information). 46
B.5 Biographical information file upload screen. .................. 47
B.6 Word list file upload screen. ........................................ 48
B.7 Screen to specify URLs to be web crawled. ..................... 49
B.8 The cases as displayed by the Case Manager. .................. 50
ABSTRACT

This thesis presents a novel interface for collaborative Digital Forensics. The improvement in the process management and remote access apropos of the use of current Digital Forensic tools in the area of Digital Forensics is described in this thesis. The architecture presented, uses current technology and implements standard security procedures. In addition, the development of software modules, elaborated later on in this thesis, makes this architecture secure, portable, robust, reliable, scalable and convenient as a solution. Such a solution, presented in this thesis, is not specific to any Digital Forensics tool or operating platform making it a portable architecture.

A primary goal of this thesis has been the development of a solution that could support law-enforcement agency needs for remote digital decryption. The interface presented here aims to achieve this goal. The use of two popular Digital Forensic tools and their integration with this interface had led to a fully operational portal with 24X7 digital decryption processing capabilities for agents to use.

A secondary goal was to investigate ideas and techniques that could be helpful in the field of “passphrase” generation and recovery. The implementation of certain computational models to support in this research is under way. The interface has been designed with features that would be part of the foundational work of developing new passphrase breaking software components.

Establishing a dedicated setup for the Digital Forensic tools and creating a secure, reliable and user-friendly interface for it, has been a major component of the overall development effort in creating the portal.
CHAPTER 1

INTRODUCTION

In this digital world, software solutions provide services that aid in almost every aspect of human life. Their implementation and usage has led to enhancements and improvements in the day-to-day processes followed by individuals, institutions and organizations alike.

The widespread use of such technologies has led to an unwanted yet unavoidable situation where they are being used for malicious purposes. Criminals have found new and powerful tools that let them obfuscate information they want to protect, have secure communication channels with their accomplice(s), perpetrate an attack on large digital networks [1] as well as cause harm to vulnerable individuals [2]. As these new technologies come up, they provide new vulnerabilities for these individuals to exploit. An agency trying to enforce the law or just trying to keep their system uncompromised has to keep up with the changing trends and be alert about attacks.

There has been a need for procedures that enables law enforcement agencies to investigate these acts of crime and help prosecute them in a court of law, by providing the necessary evidence. Forensics, which comes from the Latin word *forensis*, meaning *of the forum* [3], was practiced in the ancient Roman empire. “Entering the English vocabulary in 1659, the modern meaning of forensics is now limited to the areas of legal and criminal investigations.” [3]. The implementation of this science in the digital domain has been termed as “Digital Forensics”. “Digital Forensics” can be defined as “The use of scientifically derived and proven methods toward the preservation, collection, validation, identification, analysis, interpretation, documentation and presentation of digital evidence derived from digital sources for the purpose of facilitating or furthering the reconstruction of events found to be criminal, or helping to anticipate unauthorized actions shown to be disruptive to planned operations.” [4].
1.1 Contributions

The proposed design of an “interface for collaborative digital forensics” aims to provide a solution for a secure, remote and effective digital evidence acquisition process. The features associated with this design have been created to improve the collaboration and information exchange required during a digital forensics investigative process. The design should be flexible enough to adapt to the changes that may occur.

The second contribution of this thesis is the proposition and implementation of methods that aim to increase the probability of a successful passphrase breaking attempt. The implementation of computational models as well as the trial of new methodologies in the area of passphrase breaking requires information related to the current investigative process. Providing such information in a suitable format to a passphrase generation scheme, would certainly help in getting better and faster results in passphrase breaking. This thesis, based on the previous research done on human-memorable passwords\[5\], contends that most passphrases generated by individuals would be human-memorable as well, making these proposed methods for gathering and providing information very useful indeed.

The two contributions described above are related to each other by the increasing need for efficient passphrase breaking mechanisms. During a digital forensics investigation, the need for methods to gather and transfer all the data available to be processed by passphrase breaking tools is coupled with the need for these tools, using an interface, to effectively collaborate with other tools available, and communicate back results to the law enforcement agents or corporate security professionals, as the case may be. The integration of the proposed design and implementation of these methods as features in this design would certainly make the digital forensics process efficient and collaborative.

1.2 Summary

The rest of this thesis is organized as follows. Chapter 2 serves as a background for the current practices followed in the use of tools that aid in the digital forensic process and discusses the need for digital forensic models. Chapter 3 defines the purpose of this thesis, as well as outlines the objectives related to this thesis. The design of the system as well as the components that are a part of the design is discussed in detail in Chapter 4. An important component that was designed and developed to make the interface secure and
reliable is described in Chapter 5. The mechanisms involved in the integration of the interface with the decryption tools is described in Chapter 6. The requirements and constraints associated with the design of the interface are elucidated in Chapter 7. Chapter 8 reviews the foundational work in the area of passphrase breaking. Finally, Chapter 9 provides some concluding thoughts and mentions some future work.
CHAPTER 2

Background

The process of effective digital evidence acquisition is a very important phase in the digital forensics investigation of an event caused due to malicious intent. The information due to its evidentiary value is prized by law enforcement agencies and corporate security groups. This information can help expose security holes in an organization, as well as help in understanding the manner in which the event had occurred and the individuals responsible. Such individuals would obviously want to cover their tracks. They do so by using the technology available that allows them to destroy or at the least obfuscate any information pertaining to them.

The following describes the current practices followed by law enforcement agencies and corporate security groups during such investigations. A brief mention of the need for digital forensic models is also made.

2.1 Current Practices

People with deleterious intentions use the technologies and tools available to break the law and also protect themselves from being caught and prosecuted. Some of these tools include features that encrypt data using the whole suite of encryption algorithms available. This prevents access to this data to almost everyone except the individual that has the knowledge of the key or cryptovariable that was used to obfuscate the information.

Law enforcement agents as well as corporate security professionals use procedures laid out by their organization to investigate events as well as audit their systems to test conformance with the security policies of the organization. These procedures are mostly guidelines about where to look for information. They also outline the steps that need to be followed for the collection, validation, identification, analysis, interpretation, documentation, preservation and presentation of digital evidence. During the collection process one may have the device
that supposedly has the data that could probably be presented as evidence but it becomes a major challenge to gain access to it as it may be encrypted. This in the absence of enough resources and tools becomes a bottleneck in the investigative process.

Law enforcement agents and corporate security professionals use tools, available either as Open Source or proprietary licenses, which can make an attempt to decrypt the data available on a system. The tools apply certain attack methodologies that have known to be effective in decrypting such data without the knowledge of the key or cryptovariable. It has to be mentioned that there are many variables that play a significant role in choosing the tools to be used for making the attempt to decrypt. These include, but are not limited to the Operating System that the system is running, the application that was used to encrypt the data in the first place, the encryption algorithm that was used by the application.

Most of these tools have their unique system requirements and often require considerable setup to be effective. Also, each tool comes with its own user interface and features that require some user training for proper usage. These tools usually run at a central site where they are managed by the administrators or are installed as a stand-alone application on the workstations of law enforcement agents and corporate security professionals.

2.2 Collaborative Digital Forensics and an Interface

Considering the facts mentioned above, the need for a system that is able to integrate with any such tool and yet have a user-friendly interface for users to these tools is immediate. As encryption applications become more and more sophisticated, the use of standalone workstations for brute-force decryption attempts would not be very effective. An interface hosted at a central location with the capability to support the collaboration between various entities involved in the analysis phase of a digital forensics investigation needs to be designed and implemented.

This thesis is based upon a similar notion of creating a user-friendly user interface that would be integrated with two commercially available digital decryption tools (described in detail in Chapter 3), to create a solution that supports law enforcement agency needs for password breaking. The development of an interface which gives its users “access” to a dedicated set of machines performing digital decryption is described in detail in this thesis.
2.3 The need for Digital Forensic Models

Although this topic is beyond the scope of this Thesis it is important to understand an active area of research in the academic world, that aims to ameliorate procedures followed in this field. The way Digital Forensic Science is implemented has a direct impact on

- The prevention of further malicious events occurring against the intended “target”.

- The successful tracing back of the events that occurred which led to the crime, and determining the guilty parties involved.

- Bringing the perpetrators of the crime to justice.

- The improvement of current prevention mechanisms in place to prevent such an event from occurring again.

- Improving standards used by corporate security professionals to secure their respective corporate networks.

- How everyone “plugged” into this digital environment can increase their awareness about current vulnerabilities and prevention measures.

There has been a need for a standard methodology used for all Digital Forensics investigations. There have been many initiatives made to have models that have a general process to be followed for such investigations [6]. Research done by the scientific community has been fairly recent, and has concentrated mostly upon coming up with good models that can be practiced [7]. Yet, it can be safely said that these models are mainly ad-hoc and much needs to be accomplished in this particular domain.

2.4 Summary

Access to digital decryption tools is usually not available remotely. Users of these tools have to “transport” the encrypted files to the site where these tools are running, to be processed. A majority of these tools do not provide an online interface that allows users to have remote access to these tools. A new system needs to be designed that can act as a secure, reliable, efficient and user-friendly online interface to these tools. The following chapter presents the purpose and the objectives of such a system.
CHAPTER 3

Purpose and Objectives

This thesis is based upon the work done for a National Institute of Justice funded project, known as the DNA Project (Distributed Network Technology for Large-Scale Code Breaking). The primary goal of this project has been the establishment of a simple and easy to use silo at Florida State University (FSU) that is aimed towards supporting law enforcement agency needs for password breaking. A silo in this context is a collection of software and hardware components created for a specific purpose and accessible by authorized individuals.

The primary purpose of this thesis has been the development of a solution that would be able to assist law enforcement agents as well as corporate security professionals in the process of effective digital information (evidence, audits, etc.) acquisition through an interface to digital forensics tools. These tools are available via Open Source or proprietary licenses and have unique user interfaces and features that usually require significant user training. These tools may also have special requirements for the compatible operating platform and environment. These tools, as mentioned in Section 2.4, usually do not have a secure and reliable online interface that can be accessed remotely. The solution proposed in this thesis is based on the work done in creating a working implementation for the DNA project. The design and implementation of such a solution has been based on two popular commercially available digital decryption tools, namely Distributed Network Attack (DNA)TM and AccessData Rainbow Tables™. The architecture developed in this process is expected to be generic enough to be used with other digital forensics tools without many changes.

The work done under the DNA Project has been in support of an existing digital decryption tool called Distributed Network Attack (DNA)TM developed and maintained by AccessData Corporation® [8]. AccessData has been involved in building increasingly
sophisticated digital forensics tools that include digital decryption tools such as DNA™. For this project it was envisioned that agents would be able to submit encrypted files (later termed “jobs” in this thesis) through a secure, reliable and user friendly portal and have these files processed by AccessData code breaking tools at the backend. There were two commercially available tools used in this project and are described as follows.

The first tool AccessData Rainbow Tables™, are a form of pre-computed, brute-force decryption attacks for certain types of documents such as Microsoft® Office. Based on a “faster cryptanalytic time-memory trade-off” [9], this tool can decrypt files extremely quickly. All files that are submitted to the portal are first processed by this tool.

In the event of the Rainbow Tables application not being able to decrypt the file, the DNA™ tool is put in use. Running in a distributed environment and developed to employ the idle CPU time of volunteered machines across a network, it resembles the SETI@home project [10]. This tool can handle a comprehensive list of applications that are used for encryption. The transfer of the file to DNA™ from the other tool (Rainbow Tables) is done manually. The future versions of DNA™ will have features that make this process automatic. There are two major components in the DNA™ system and a third one comes into play in certain circumstances. Based on a Server-Client paradigm, the main engine that accepts files for processing, creating and distributing tasks to the clients, supports a user interface and maintains communication with all the clients is called the DNA™ Supervisor. The client in this system that does the task of code breaking is called the DNA™ Worker. There is a limit to the number of DNA™ Workers that a DNA™ Supervisor can manage effectively. In order to manage a large number of such clients, the DNA™ Master Controller is used. This component can control multiple DNA™ Supervisors which then control the DNA™ Workers. The architecture of the DNA™ system is shown in Figure 3.1

This project was developed by a team at FSU in collaboration with Eric Thompson, Founder and CTO of AccessData Corporation®.

Some of the major objectives defined for this thesis were:

- Develop a system that would interface with two digital decryption tools, DNA™ and AccessData Rainbow Tables™ in.

- Develop a DNA silo that has 24 by 7 job submission capability and is accessible by agents for processing of their files.
• The portal to be built needs to be secure, reliable and user friendly.

• The solution developed should be fairly easy to integrate with other digital forensic tools after reasonable modifications

For this thesis the DNA Silo and the online interface to be built is collectively termed DNAOnline and is described in detail in the rest of this thesis.
CHAPTER 4

System Design

A multi-component approach has been used in the design of this interface. All components have been developed as portable applications. The primary reason for doing so was to make the process of integration with a digital decryption tool seamless. Different digital forensics tools have unique system requirements and usually have certain requirements as to the type of operating system they can run on. Hence, having these as portable components is clearly an advantage.

As the system has been built to serve as an online interface to digital forensics tools and specifically digital decryption tools in this thesis, there has to be a web component that forms a major component of this design. The web component is an ensemble of Java\textsuperscript{TM} based modules running on the Apache Tomcat v5.5.20 web server [11]. The following section describes this component in detail. These modules along with a stand-alone Java\textsuperscript{TM} application, known as the Session Server are hosted on a Fedora Core 5 Kernel v2.6.18 (Linux) operating system.

As the AccessData decryption tools used in the DNAOnline system are compatible to run on the Windows\textsuperscript{R} XP/2000 based platform, a component has been created to support the collaboration of the web component with these tools. This is a custom-built Java\textsuperscript{TM} application (described later in Chapter 5) that is used to interface with the tools as well as to move appropriate content to and from the Windows\textsuperscript{R} platform on which the decryption tools run.

When interfacing with other digital decryption tools which run on a different operating platform, some of these modules and applications developed, would need to be modified. This modification would not be a difficult process though as these components are portable and are themselves modularised. A generic version of this interface design is described in
more detail in [12]. For now, we concentrate on the design in the context of interfacing and collaborating with the AccessData decryption tools. The architecture makes the system design highly flexible and portable.

4.1 The Web Component

Forming the major component in the design of DNAOnline, are the web applications that run on the web server. These applications consist of modules that feature Server-side processing of requests from web clients and dynamically generated Hyper Text Markup Language (HTML).

Three Java™-based technology components were used to develop these modules. The first one, Javaserver Pages (JSP)™ technology [13], is used for generating dynamic web content based on the requests of the web client, initiated by the user. An example of such an action is in the case of user login. The request is sent over by the web client as a user authentication event and the corresponding JSPs managing the event process the request.

The second type of technology known as Java Servlets™, is used for extending the functionality of the web server, as well as for generating on-the-fly content [14]. An example of the use of this technology is when the cases (described in Section 4.3.2) of a user are displayed.

As with any online interface that supports multiple user accessing the system at the same time, the notion of user sessions is at the core of such an interface. The session management modules are developed through a technology known as JavaBeans [15]. These are reusable software programs and are at the forefront of efficient Server-side processing, state management and aid in making the interface reliable. These modules once created can be easily invoked and managed as instances by the JSP components using the jsp:useBean, jsp:setProperty and jsp:getProperty tags. The following illustrates their syntax as defined by Sun Microsystems Inc. [16],[17], [18].

```xml
<jsp:useBean
    id="beanInstanceName"
    scope="page\request\session\application"
    { class="package.class" | type="package.class" |}
```
The separation of each of the instances of the JavaBeans running in a different “sandbox”, for each user session is managed by the Apache Tomcat web server.

The use of the Tomcat web server is advantageous as it “is the servlet container that is used in the official Reference Implementation for the Java Servlet and JavaServer Pages technologies“ [11]. Running a web server securely is extremely important. Hence, some standard procedures used by the industry to harden the web server have been implemented here. The web server is made to execute as a daemon with permissions of a non-privileged user. In order to have the web server bind with the reserved HTTP and HTTPS ports 80 and 443 respectively, available for privileged users(such as root) on the Linux machine, and still run with the permissions of a non-privileged user, the jsvc [19] tool was used. Using this tool, the web server process is initially started with privileged user permissions so as to be able to bind with the aforementioned ports. The permissions are then downgraded to be that of the non-privileged user. The output of the top command shows the web server
running as process \textit{jsvc} as the user \textit{tomcat} that has similar permissions as the user \textit{nobody}, commonly used in most Unix-based systems as a non-privileged user.

\texttt{#top -n 1 -u tomcat | grep jsvc}

\texttt{<process id> tomcat 18 0 242m 57m 11m S 0.0 5.7 0:11.53 jsvc}

An important task in the design and implementation process was to ensure that all communication between the web server and its web clients was secure and protected. The Tomcat web server supports the standard SSL/TLS protocol \cite{20}. An SSL certificate generated for the web server enables the web clients to set up a secure SSL connection and then communicate securely. Hence, sensitive content such as user authentication credentials and files being transferred will be encrypted.

Figure 4.1 shows a sample screen shot of one of the pages that is displayed by the web component.

### 4.2 User Authentication

Given the nature of this interface, it is essential that only authorized users be allowed access. After providing appropriate authentication credentials, users are allowed to use the features available in the interface. This includes the option of submitting files to be processed by the digital decryption tools. Hence, access control mechanisms had to implemented in the design in order to prevent the service being accessed by and probably misused by unauthorized individuals. The web server hosting the interface would first receive the user authentication credentials. Being exposed to the external network makes the web server vulnerable and consequently there are limitations in how well it can be reasonably secured. Hence, important and sensitive information is kept off it. This information is passed on to a trusted host on the internal network as soon as it is available at the web server.

For user authentication, a mechanism was needed that would implement a widely used and strong user authentication scheme. The authentication mechanism would have to be similar to a Single Sign-on solution. After gaining access to the interface, a user will have access to the services offered including having their files processed by the decryption tools employed at the backend. The “trusted third party network authentication protocol designed to provide strong authentication using secret key cryptography” \cite{21}, known as
Kerberos V5 [22], has been implemented. Although the protocol is quite complex and has multiple components that act together, it can be simplified and explained as such. The major components are the Authentication (AS), Ticket Granting Server (TGS) and finally the Service Server (SS). We can also define an entity called the Subject that is requesting the authentication. In this case the Subject is the web client that acts on behalf of the user generating the authentication request. The Subject tries to authenticate itself to the AS by providing the user credentials. If the authentication request is successful, then the Subject can request for a token, known as a ticket. This ticket is used by the Subject to access the service(s) offered by the SS.

In the design of the interface the AS, TGS, and SS would be on the same machine. The decryption tools in this design are running on a trusted host on the internal network.
As files to be processed by these tools are to be transferred on to this host, having the authentication mechanism on the same host makes sense. The processing of files by these tools can be thought of as a service and the users can access that service by authenticating themselves to the AS and then getting a ticket. As Kerberos V5 is compatible with almost all operating platforms, it can be run on the same host as the decryption tool.

In the DNAOnline system the host running the decryption tools and implementing Kerberos V5 has been taken to be a Windows® 2003 Server. The Active Directory feature available in this operating system has been set up to manage user accounts. The authentication mechanism for this feature is implemented via the Kerberos V5 protocol. The AS component when configured makes this host act as the AS for authorized users/Subjects. The web component is the Subject that acts as the Kerberos client, issuing remote user authentication requests. There is no requirement of keeping user credentials in persistent storage on the web server. The Kerberos Login module known as Krb5LoginModule is Sun’s implementation for the Kerberos V5 protocol. The web component uses a similar client configuration entry as shown below to implement the Kerberos V5 client module.

```java
KerberosAuthHandler {
    com.sun.security.auth.module.Krb5LoginModule required useTicketCache=FALSE;
};
```

The useTicketCache option controls whether the Ticket Granting Ticket (TGT) [22] is to be obtained from the ticket cache available in the operating system.

## 4.3 Features and Supporting Components

As mentioned earlier in Chapter 4 the DNAOnline system is a multi-component solution. This section describes some of the major components and the corresponding features that DNAOnline supports.

### 4.3.1 User Account Management

One of the most important features that DNAOnline provides is the support for multi-user access to AccessData’s digital decryption tools. DNAOnline does not provide users with direct access to the user interface of these tools. The DNAOnline interface instead manages
the jobs being submitted by users and processed by these tools. The tools do not have support for multi-user access and the DNAOnline interface can be thought of as a layer of abstraction above these tools. This feature aims to significantly reduce the time it takes for agents or corporate security personnel, to have their encrypted files processed and getting back a result. Most installations running these tools without a remote interface such as DNAOnline have an administrator or a team of administrators, depending on the size of the setup, managing the processing of encrypted files. The DNAOnline interface is built with the intent of improving this process of Job management for users and site administrators.

There are two components that have been developed for this feature. The first component takes care of the user account creation and the management of user accounts and user credentials. This component, known as Administration Manager, is created as a web application that is running on the web server. The second component handles user authentication requests after a user account is created.

As mentioned in 4.2 all users accounts are created and managed using the Active Directory service that Windows® Server 2003 provides. A user account in DNAOnline consists of user authentication credentials, user information and a designated storage space on the Windows® 2003 host. The Administration Manager provides an administration interface that allows the individual(s) designated as site administrators during the setup of DNAOnline to create and manage user accounts. There is a complementary component known as the Administration Helper on the Windows® 2003 server that initiates system commands to create and manage user accounts based on the requests made by the Administration Manager component. The Administration Helper component is a Java™ based component. There are currently three features that Administration Manager provides to site administrators. The first feature is the creation of a new user account. The site administrator provides information about the user and their credentials and the account is created. The second feature is the ability to reset user passwords as required. The third feature is the ability to delete an existing user account. The Administration Manager checks whether the user account being deleted has any Jobs that are currently being processed. The site administrator is notified of such an event during this process.

The second component, Authentication Manager, is built to handle user authentication requests. As described in Section 4.2, this component uses the Kerberos V5 login module (Krb5LoginModule) to act as the Kerberos V5 client on behalf of users. The configuration
of this component has the Windows® 2003 server as the Authentication Server, where all authentication requests are sent [23]. The Java Naming and Directory Interface (JNDI) [24] is used in this configuration to access the Active Directory setup. JNDI supports interfacing with naming and directory services available on most operating platforms thus extending the portability of this design.

4.3.2 Case Manager

Investigators organize their work into Cases, which typically deal with a single or a series of incidents. There could be multiple encrypted files related to a Case and the feature of job management on a per Case basis is usually not supported by many decryption tools. In the design of DNAOnline, for each user account, a case is a logical entity created as a directory in the storage space allocated for an account. There could be multiple such logical entities in a user account as required. The Case Manager component has been developed for managing the creation, management and removal of cases as requested by users. The requests sent over by the user can occur after a user successfully authenticates and logs in. Hence, this component is session-scoped and is active and presented to the user after login. The Case Manager has features that manage the access control to cases. The gathering of auxiliary information during the creation of a case is also managed by this component. This auxiliary information could be wordlists and/or dictionaries from uploaded auxiliary documents or output from an on-the-fly web crawler. The user can provide a set of URLs to crawl and text from those sites are pulled to assemble a wordlist that could be used to create dictionaries. This capability is deemed to provide useful input in the passphrase recovery research (See Chapter 8), as heuristic-based attacks using such custom dictionaries would be more likely to produce a successful result. Brute-force attacks, considering the huge key space in the passphrase domain, would most often not be effective. The current use of these custom dictionaries in processing of jobs, is achieved by using the DNA™ decryption tool’s user interface to import these dictionaries in its own pool of dictionaries. Additional capabilities in future versions of DNAOnline, would support automatic imports of these dictionaries.

4.3.3 Case Explorer

One of the primary objectives of DNAOnline is to provide the user real time information about the status of a job. As jobs are placed in cases, the management of jobs inside a case
is handled by a component known as the *Case Explorer*. When the users request for case specific information such as status of the jobs or want to upload a new job for processing, this component comes into play. The status of a job in a case corresponds to one of the four bins that have been defined for each case. They are:

- **Submitted**: Jobs are initially received for processing in this bin and have to be submitted to the decryption tool. In the case of DNAOnline, these tools are DNA\textsuperscript{TM} and AccessData Rainbow Tables \textsuperscript{TM}.

- **Processing**: As soon as a job is submitted to the decryption tool for processing, it is placed in this bin. This acts as an indication to the user that the job is currently being processed but not yet finished.

- **Succeeded**: The result of a successful decryption done by the decryption tool is placed in this bin. The user can retrieve the result(s).

- **Failed**: If the decryption tool is not able to generate either a key or a password for a job, it is termed “failed”. The job is placed in this bin to indicate the same to the user.

Figure 4.2 illustrates the user account organization in this design.

In order for processing (by the decryption tool) to be started on the submitted jobs, a component on the Windows \textsuperscript{®} 2003 Server has been created, known as the *File Management module*. This component scans the *Submitted* bins inside the *cases* of all users and submits the job to the decryption tool. This component performs job to case mapping for all users. This component manages the moving of jobs to the *Processing* bin or *Failed* bin and the results to the *Succeeded* bin.
Figure 4.2: User Account Organized as cases.
CHAPTER 5

The Session Server

The effective management of user sessions is essential for any web enabled application. Implementing this feature is a complex process which involves taking into account the restrictions and issues involved in the design. Many web applications utilize client-side state keeping mechanisms (cookies) and/or client-side scripting to handle this issue. Although effective, there could be exploits associated with them that makes applications more vulnerable to being compromised. Hence, a component known as the Session Server has been created to manage user sessions without the use of such client-side processing.

As explained in Section 4.2, the user accounts have a designated storage space allocated on the Windows® 2003 host. Considering the ideal scenario, when a user successfully authenticates and logs in, a user session is started. At this time, the storage space associated with that user account needs to be mirrored onto the web server serving the client’s requests. In DNAOnline, The Common Internet File System (CIFS) [25] mount service is used to achieve this. The Tomcat Session which is a Java™ object instantiated by the inbuilt Servlet Container, is “used to maintain state information between requests for a particular user of a web application” [26]. The Tomcat Session remains active as long as the user session remains active. This can be achieved programmatically (e.g. `session.invalidate();`) when the user sends a request to log out. At this time the CIFS service to ummount is used to make sure that no files associated with the user account are available on the web server. This mechanism for mounting and ummounting of user accounts is processed by a multi-threaded Java™ application known as the Mount Controller.

All web applications are based on the idea of a request being generated by the web client which is processed and served as a response from the web server. The communication has to be initiated by the client. There are many triggers in a network in the real world that
may prevent the client to maintain communication with the web server and hence disrupt the Tomcat Session. These can be network traffic delays, network failure or the event of a user closing the web browser accidentally. In the absence of any client-side processing a component was required that would maintain user account consistency by handling such events. The support for multiple sessions for a user as well as session timeouts in the case of user inactivity was also required to be supported. As the web server, exposed to the external network, is usually the most vulnerable to being compromised, the protection of user credentials and files was required as a security feature. The same protection is required in the event of a web server crash.

A multi-threaded Java\textsuperscript{TM} program, called the \textit{Session Server} has been developed, as a solution for all of the above. The \textit{Session Server} spawns a new thread to correspond to a new Tomcat Session. When a user successfully logs in, the request for the user account to be mapped on to the web server is made by the \textit{Session Server} to the \textit{Mount Controller}. The \textit{Mount Controller} can be thought of as being in direct interaction with this component. Any new Tomcat Session, sets up a TCP/IP Socket connection and sends a 128-bit secure random number \cite{26} that it is associated with it and the username of the user as a “signature”. The \textit{Session Server}’s thread in response sends a 128-bit secure random number that it freshly generates for this purpose. This exchange of signatures occurs until the user session ends. This component maintains a Vector of the current active user sessions. This enables support for multiple sessions for a single user without inconsistency. All previous sessions except the latest one are invalidated but allowed to complete processing, such as web crawling to generate wordlists.

When a user session is active there are requests generated at the web client that are sent to the web server. It may so happen that a user may close the web browser accidentally, there could be issues with the network that may hamper data transfer or simply, the user may have left the console with an active session. These events are monitored using the concept of “activity beats”, which is a message generated whenever there is user activity (e.g. clicking on a button etc.). During periods of user inactivity, the \textit{Session Server} checks for the time at which the last “activity” was recorded. If this time exceeds the threshold set, the corresponding user session is invalidated. There is also a trip-wire system in place that generates requests for all user accounts currently active to be unmounted by the \textit{Mount Controller}. This is to prevent access to confidential user files in the event of a web server crash.
The following pseudocode shows the steps taken by each active thread of the Session Server to check for user inactivity and/or the event of a web server crash.

**Vector V: Current active user sessions**

```java
#set the socket timeout value
socket.setTimeout(timeout value)
while (user session is active) {
    try {
        communicate with the web application;
    } catch (socket timeout exception) {
        #Invalidate user session
        V.invalidate(user session)
        # Unmount the user account
        MountController.unmount(user account)
        Perform any cleanup actions such as closing open file handles etc.
    }
}
```

The outline of the design of the DNAOnline system showing all the components including the Session Server is shown in Figure 5.1
Figure 5.1: System Design showing all Components.
CHAPTER 6

Integration with the AccessData decryption tools

The mechanisms involved in the integration of the DNA\textsuperscript{TM} silo and the online interface to work collectively as DNAOnline are elucidated in the following. The interaction between the DNA silo and the online interface needs to be setup such that jobs submitted by users are processed efficiently. As shown in Figure 5.1, the online interface is hosted on the web server. The DNA silo comprises of modules such as the Session Server on the Linux side and some modules and two decryption tools, namely DNA\textsuperscript{TM} and AccessData Rainbow Tables\textsuperscript{TM}, on the Windows side.

6.1 File Management in the DNA silo

In order to have a fully functional DNAOnline system, the file handling features associated with the two decryption tools available needed to be understood. The AccessData Rainbow Tables\textsuperscript{TM} application has the ability to scan a directory (including its sub-directories) for files. This directory is the Input directory, as any file placed in such a directory is processed by this tool as input. This tool can generate two types of output based on the type of input file given to it. If the file is encrypted using an application that this tool is able to attack using a precomputed brute-force method, then such a file is processed and the decrypted contents of that file is moved to a Output/Success directory. In the case of the file not being encrypted or encrypted using an application that it does not support, the file is placed in the Fail directory. Now, it needs to be mentioned that the input and output directories can be configured and an example is shown in Figure 6.1. The Input directory here is IN_JOBS, the Success directory is OUT_SUCCESS, and the Fail directory is OUT_FAILED.

The other decryption tool available was DNA\textsuperscript{TM} version 3.2. This tool, unlike the first one, does not have the notion of input and output directories. A file to be decrypted needs
Figure 6.1: The AccessData Rainbow Tables interface and configuration window.

to be submitted by the silo administrator manually using the user interface. The future versions of DNA™ are being developed to support such a feature.

In order to utilize the file handling features of AccessData Rainbow Tables™, and to allow for a seamless integration with future versions of DNA™ with similar file handling features, a mechanism for the exchange of files between the online interface and the DNA silo was established. The online interface, after receiving a job to be processed, is placed in the Submitted bin, as described in Section 4.3.3. There are three more bins (refer to Section 4.3.3) that collectively represent a case in a user’s account. In the implementation of DNAOnline, a case is a directory with these bins as sub-directories of it.

The File Management module as mentioned in Section 4.3.3, was developed to pick up files available in the Submitted directories of users and place them in the input directory of AccessData Rainbow Tables™ for processing. The management of the output generated by either of the decryption tools is also handled by this module. In its current implementation this module first scans the Submitted directories of all users for jobs. All available jobs are
moved to the input directory of AccessData Rainbow Tables™. This process also involves updating the corresponding Processing bin of the user’s case(s), to indicate the job is going to be processed shortly. When all the Submitted bins have been scanned, this module looks at the two output directories Success and Fail, of AccessData Rainbow Tables™. Files that are in the Success directory are moved to the Succeeded bin associated with a user’s case. Similarly, the files in the Fail directory are moved to the Failed bin. The whole process is repeated after a short time interval. The following shows the implementation of this logic.

#Four bins in a case
Submitted, Processing, Succeeded, and Failed
#Input directory for Rainbow Tables
IN_JOBS
#Output directories for Rainbow Tables and DNA
OUT_SUCCESS, OUT_FAILED

while(signal to stop execution not received)
{
    # Scanning for jobs in the Submitted bins of users' cases
    for each user U[i], i:1 -> N, do {
        for each case C[j], j: 1-> M, do {
            if file(s) exist in C[j].Submitted directory then {
                Move file(s) to IN_JOBS directory;
                Update C[j].Processing bin for job status;
            }
        }
    }

    # Scanning the output directories
    if file(s) exist in the OUT_SUCCESS directory then
        Move file(s) to the Succeeded bin of the user U in the case C
    if file(s) exist in the OUT_FAILED directory then
        Move file(s) to the Failed bin of the user U in the case C

}
# repeat process after an interval

wait(time_specified);

As mentioned before, DNA\textsuperscript{TM} does not have file handling features. Hence, the silo administrator needs to monitor the directory in which the AccessData Rainbow Tables\textsuperscript{TM} places files which are encrypted using an application not supported by it. Such jobs need to be manually submitted to DNA\textsuperscript{TM} and based on a successful or failed decryption attempt, the silo administrator needs to place the job in the corresponding output directory for the \textit{File Management module} to scan. This makes the current implementation of DNAOnline semi-automatic. Figure B.8 shows a user’s case with the four bins, with one job being processed. Figure 6.2 shows in detail how a job submitted by a user is processed by DNAOnline.
Figure 6.2: Processing of a Job in DNAOnline.
CHAPTER 7

Requirements and Constraints

Some of the major requirements and the security constraints associated with the design of this interface are described here. These have been taken into account during the design and implementation of DNAOnline.

7.1 Interfacing with the Digital Decryption Tools

One of the major requirements for the system to be designed has been the ability for the web application developed to interface fairly easily with the digital decryption tool(s) being used. The design needs to have features that make the process of interfacing with a digital decryption tool a fairly simple one. There has to be enough modularity in the design to allow for changes in the requirements.

Most digital forensic tools require files as input. For law enforcement these files may be present on a suspect’s hard drive and need analysis. Corporate security professionals use these tools to analyze data present in log files and other pertinent data to perform audits and general analysis of their systems. Hence, this requirement of a generic interface for digital decryption tools can be extended for digital forensic tools as well. The design of such a system would provide support for collaborative digital forensics.

7.2 Job Management Process

Current setups in most organizations usually have a system administration team that manages the installation, configuration and management of the digital forensic tools. They also have to manage the processing of files submitted by users. If the number of files submitted by users is large, a significant portion of the time of system administrators is spent in processing jobs for users, updating them on the status of the jobs, notifying them on the
outcome of the job as well as communicating results. The users usually have to rely on the system administration team to have “access” to a setup that has a set of dedicated machines running digital forensics tools. A requirement for the interface to be designed is to improve the process of user’s job management. This requirement also aims to significantly reduce the overhead of managing user jobs along with managing the site on system administrators.

The update on the progress of a file being analyzed by a digital forensic tool is displayed on its graphical user interface. This interface is usually not accessible remotely by users whose files are being processed. Also, these tools usually do not provide any user to file mapping mechanism. It needs to be noted that the user account under which the tool is running on the machine is different from the user who submits a file to be processed, and should not be confused. Hence, as an added requirement to effective job management, a feature has to be incorporated in the design that would make the task of user to file management automatic and convenient. The system administrator should no longer have to manage such a task.

Efficient job management when incorporated into the design would allow users to have access to an online interface that is being hosted on a web server. Some of the components on this interface would somehow communicate with the digital forensic tool, for example a digital decryption tool, to process the jobs submitted by users. Users will no longer have to wait for the system administrators to communicate with them to know the status of jobs as well as any results available. The online interface would provide direct access to the digital forensic tool by acting as a proxy for them. This is aimed towards significantly reducing the wait time for users. The system administrators can concentrate on maintaining the setup and manage issues such as load balancing, server performance, etc. An overview of this requirement, taking a distributed digital decryption tool is shown in Figure 7.1

7.3 Extensibility and Portability

New digital forensic tools are developed and existing tools are updated to have new features. These are used by organizations that see a need for such tools. These upgrades on existing digital forensic software as well as new tools may come with a unique user interface and new features. The interface needs to be extensible and adapt to such changes without any major changes in design. In the context of the DNA™ tool, a new feature of generating dictionaries from wordlists that are submitted by the user along with a job needs to be supported by a feature on the online interface that supports the upload of such auxiliary information. The
integration of a web crawler that then generates custom dictionaries on the fly are features that should not need a major change in design. Making the interface extensible makes it easier to adapt to change.

It is often the case that different digital forensic tools have their own unique system requirements that include specific operating system requirements. The interface hence needs to be portable. The design should be built using standard software components and on standard platforms. Although porting to a different operating platform and interfacing with a different tools would require some effort, the design itself need not be changed. Changing the behavior of certain components should be sufficient.

Taking the example of the DNAOnline application, the DNA™ Supervisor runs only on
7.4 Security Constraints

Most organizations secure their setup of digital forensic tools by securing access to these tools to authorized individuals. Security is a major concern as these files might contain confidential data that needs to be protected. Data of evidentiary nature needs to be handled under strict guidelines. The requirement of an interface to these tools, as mentioned in Section 7.2, without the requirement of a system administrator in the middle has a direct impact on the security concerns. Although, this is a much better way to do things, certain security constraints need to be taken into account during the design and implementation phase. The next subsections describe the types of security constraints associated with the design of the interface.

7.4.1 Security of the Digital Forensic Tool Host

Access to the digital forensic tools being used by an organization is usually restricted to authorized users only. Having an online interface implies that there would be certain components on the web server that would have a communication channel with these tools. This is necessary in order to process jobs for users, provide status information on the progress of the job and sending back any results available. A web server is always vulnerable to new exploits and attacks and can be compromised. Even though users of the online interface would have to successfully authenticate before getting access to any services, the design of the system needs to secure access to the forensic tools at hand.

The main components to be considered in this discussion are the machines that host the web server and the digital forensic tool(s). The web server hosting the web application can be thought of as an application host. Taking the example of a digital forensic tool, its corresponding host could be termed as a decryption tool host.

In order to secure the decryption tool host, it needs to be invisible to the outside world, placed in an internal network protected by a firewall. The only machine that would be able to communicate with this host would be the application host. All requests coming in from web clients would be routed to the application host. The requests that are to be processed by
the decryption tool host would be generated by the application host and hence routed to it. Any external access request to the decryption tool host would not be generated as this host is isolated to a local subnet. Figure 7.2 shows a basic network configuration which conforms to the constraints.

7.4.2 Web Server Security

The web server that hosts the online interface would be at the center of it all, processing web client requests and interacting with the components that interface directly with the digital forensic tool. Place on the perimeter network, web servers are known to be vulnerable to attacks. Hence, steps need to be taken to use state-of-the-art technology and follow industry-compliant guidelines to run it securely in a hostile environment. Care needs to be taken to have user authentication credentials off of the web server. Vulnerabilities associated with client-side processing such as the use of JavaScript and other such technologies need to be avoided as most highly secure environments disable the use of such technologies.
CHAPTER 8
Passphrase Research

Passphrase breaking is a novel and challenging area of research. Most of the digital decryption tools out there have primarily concentrated on passwords rather than passphrases. Even the ones that have addressed the passphrase breaking problem have not considered the challenges in this area and come up with the ways to overcome those challenges. The following describes some of the challenges in passphrase breaking and the initial research that has been done in this area. The features in the interface that is hopeful to be of aid in this research are discussed.

8.1 Challenges in Passphrase Breaking

Encryption in some form or the other is used by system and application software for various purposes. A common yet very important use is access control. The decision to allow access to the services provided by the software to a user is dependent on the authentication and/or identification information provided. There are also cases where an individual uses a key or cryptovariable, as it is sometimes called, to obfuscate the information they want to protect. This is achieved by the use of one of numerous application softwares available either via Open Source or proprietary licenses. These software typically use encryption algorithms in their implementation. The most common way of generating a key that is unique for that user for that particular software is done by requesting user credentials, which are usually passwords.

Code breaking tools try to recover the encrypted data by decryption. An attempt is made to decrypt a file that may have been protected by one or a series of encryption algorithms. The protection mechanism is based on the user password, that was supplied by the user before the encryption process was started. These passwords are usually mapped computationally
to a key or cryptovariable which has a specific length in bits. This length is dependent upon the type of encryption application used. Some of the most popular ones are usually based on or default to a 40-bit keyspace. Decryption tools have an attack type that makes most decryption attempts successful even in the event of a complex password being used. This attack utilizes the current number crunching abilities of processors that can process something in the range of several Giga instructions per second. Based on a 40-bit keyspace, the time it takes to search and exhaust the $2^{40}$ possibilities for a key is usually quite short. If the decryption tool is based on a distributed architecture with a dedicated set of client machines trying decryption on their allocated area of the keyspace, the time required is reduced even more.

Key lengths greater than 40-bits have also been looked at extensively in research done in academia and industry. There have been quite a few “smart” attacks that have been developed that exploit the fact that a password chosen by a user may be variations of the words that are in the vocabulary of that user. These words would most likely be found in the dictionary of the language that the user speaks. One of the attack schemes used is known as the dictionary attack. As the name suggests, words from dictionaries of a language are used as passwords for the attempt to decrypt the file. These dictionary attacks also have generation schemes that do transformations on these words to generate strings that may be the password used to encrypt the file. These generation schemes are based on the requirements of systems that require complex passwords. Such requirements include the use of digits, having at least one uppercase character, certain symbols etc. Even with such requirements, dictionary attacks can generate passwords that cover the keyspace for all the variations as long as the password is a variation of a word in the dictionary of the language used by the user.

Some of the newer and popular encryption applications are implementing new algorithms that considerably slow down any brute force attacks on their encrypted files. The use of a 128-bit keyspace is also becoming more and more popular. An example of such an application is Pretty Good Privacy (PGP) that implements both the public-key and symmetric key encryption schemes in its solutions for e-mail, Hard drive, etc. encryption. The schemes used makes any decryption attempt by a decryption tool with a dedicated cluster of high-performance CPUs performing code-breaking, difficult. Some claim that for a seven character password (letters and numbers), the time a single processor would take to find the correct
password would be close to 7,000 years \[27\]. The new encryption applications have proven considerably difficult to attack using standard attacks. Another level of complexity that these applications have brought about is the use of a passphrase instead of a password. Although, there is no precise definition of a passphrase, it may described as a sequence of words or variations thereof. This sequence may or may not have any whitespace, hence increasing the randomness. A simple example of a passphrase, such as “Bob and Alice” may be a non-recoverable job for most current decryption tools.

\section*{8.2 Research Activities}

The research focus in the area of passphrase breaking has been on the development of techniques and algorithms that can support faster decryption of passphrases. There has also been a focus on the design and implementation of features in the Web interface that can support these techniques by providing information that may be pertinent to the file being decrypted and hence improve upon the current passphrase breaking mechanisms.

Passphrase generation from a set of words as input is a very important and complex component in the passphrase recovery process. When trying to attempt passphrase recovery on files that have been protected by the encryption applications such as PGP\textsuperscript{\textregistered}, the trial of random passphrases from the generation process will not produce good results. This is primarily due to the passphrase keyspace being significantly larger than passwords. Hence, some techniques need to be developed that do not depend on randomly generated passphrases and yet can cover a significant portion of the gamut of the passphrase keyspace. One of the contentions of this thesis is that, based on the previous research done on human memorable passwords \[5\] being vulnerable to dictionary attacks, passphrases generated by humans will be human-memorable as well. Hence, such passphrases would be susceptible to smart passphrase generation schemes plugged into a passphrase recovery process. Certain Natural Language Processing (NLP) paradigms are being looked at that can help in the passphrase generation process by covering the reduced key space of human-memorable passphrases.

The NLP tools can be used to generate phrases for a particular language. Taking the English language into context, there can be many classifications of phrases that can be generated. Noun Phrases (a phrase whose head is a noun or a pronoun) or Verb Phrases (a phrase whose head is a verb) or other such syntactically correct phrases can be generated based on a set of words as input. The availability of such wordlists is discussed later.

36
Sentence can consist of Noun and Verb Phrases and a basic structure is shown as a tree in Figure 8.1

![Sentence tree](image)

Figure 8.1: A Sentence (S) formed as a tree of Noun Phrase (NP) and Verb Phrase (VP) nodes. Verb node denoted by V and Noun denoted by N

As in the case of password recovery, modifications of passphrases generated need to be done and tried out. These variations would be in certain systematic manner (common variations include replacing an O by a 0). The variations might include shuffling of words in the phrase, adding and deleting words in the phrase, replacing words by their synonyms or antonyms and even concatenating phrases. Some other variations could include modification based on numbers, special character and symbols [28], and Leetspeak [29]. An alternate methodology could include a passphrase generation mechanism using a non-syntactically correct technique and the output could be fed into an NLP tool to screen for “correct” phrases. A combination of these techniques could also prove effective.

The testing process and the generation process for passphrases involves interaction between the two. As the testing process is slowed down considerably due to the kind of files on which such techniques will be tested, trying a passphrase more than once needs to be avoided as much as possible. The generation could be done by a single component which then distributes subtasks to other machines, or each machine can generate a subtask of phrases that does not overlap with those generated by other machines. The development of such techniques is hence more complex than in the password recovery problem. A possible solution to avoid the undesirable overhead of duplicate generation is the use of a large hashmap, representing the targeted passphrase key space, without the need for a lookup table. Other such methods to manage such large sets of passphrases that are easily accessible by distributed machines need to be developed.

The web interface described in this thesis, has been added with some features that would
generate output that could be useful for the passphrase generation processes developed. These features aim to create wordlists that are pertinent to the file for which passphrase recovery is being attempted. It may so happen that the passphrase given by the user to encrypt the file may be inspired by components of his/her biographical information, information present in the digital documents on their computer, books they might have read, movies watched, lyrics of songs, and web-content such as blogs relevant to that user. The interface that has been developed allows agents to upload such information in the form of auxiliary documents. In case the agents want to create word lists based on a particular set of web pages they believe might be related, a feature of specifying URL(s) that will be processed by a web crawler is also added. These features will hopefully be of aid in the area of passphrase research.
CHAPTER 9

Conclusions and Future Work

This thesis presents a novel interface for collaborative digital forensics, that is secure, reliable and user-friendly. Most importantly, this thesis aims to improve upon the processes involved in the use of digital forensic tools focusing on digital decryption tools at the moment. The design of such a system with the features described in this thesis should be helpful in improving the collaboration between systems and the information exchange involved in a digital forensics investigative process. For users of digital forensic tools, such as law enforcement agents and corporate security professionals the availability of a 24 by 7 portal to a dedicated setup of such tools should prove to be extremely useful. The design of this interface has been built based on the current industry standards, involving current technology. The modularity of the design and the extensible and portable characteristics of its components makes this interface easily adaptable to changes.

The foundational work done in passphrase research along with features supported by the interface is a good starting point for future work done in the area of passphrase recovery.

9.1 Future Work

The interface presented here is expected to be made more sophisticated, complete, more communicative to the user, and fully automated portal for submission and processing of jobs submitted by law enforcement and corporate entities. The interface needs to be integrated with the next versions (3.3 and 3.4) of DNA™. This integration needs to be explored with a somewhat different and more powerful architecture that would allow users to submit jobs to multiple locations. The interface needs to be tested for its interfacing capabilities with other digital forensics tools available. The future versions of this interface need to be developed following enterprise solutions that would allow some of the features in the interface to be
presented as a web service allowing users to have access to digital forensic tools using custom-built application in addition to a web interface. Progress on passphrase breaking through the implementation of generation techniques such as those described earlier needs to be accomplished.
APPENDIX A

DNAOnline version 1.0

DNAOnline version 1.0 is available for download and experimentation from SourceForge which contains the relevant packages and documentation. The automation of the process of moving a job not recognized by AccessData Rainbow Tables™ to the DNA™ tool without the need for an administrator should be available as one of the features in the next version on DNAOnline. The SourceForge page for DNAOnline version can be accessed at http://sourceforge.net/projects/dnaonline/.

The following describes what each package available for download is required for the complete installation and configuration of DNAOnline version 1.

1. README - This is a document that serves as the reference document for the complete installation and proper configuration of the DNAOnline interface.

2. dnaonline - This tarball contains the code that needs to be compiled for the web application that users access to submit jobs.

3. dnaadmin - This tarball contains the source code that needs to be compiled for the web application that administrators of DNAOnline have access to. The creation, management and deletion of users and their credentials is handled by this package.

4. dnascanner - This package serves as the File Management module as described in Section 4.3.3

5. dnaadminhelper - This package serves as the Administration Helper component as described in Section 4.3.1
APPENDIX B

Screenshots of the user interface

The following figures are the screens that constitute the user interface for DNAOnline version 1.0
Figure B.1: The DNAOnline login screen.
Figure B.2: The main menu displayed after user login.
Figure B.3: The first step in creating a new case.
Figure B.4: The second step in creating a new case (Auxiliary information).
Figure B.5: Biographical information file upload screen.
Figure B.6: Word list file upload screen.
Figure B.7: Screen to specify URLs to be web crawled.
Figure B.8: The cases as displayed by the *Case Manager*. 
REFERENCES


52
BIOGRAPHICAL SKETCH

Rajarshi Das

Rajarshi Das was born on August 4, 1981, in Ranchi, Jharkhand, India. He joined the Department of Computer Science at Florida State University (FSU), in Fall 2005 to pursue the Master of Science degree in Computer Science.

Rajarshi got his Bachelor of Engineering (BE) degree in Computer Science and Engineering from Vidyasagar University, West Bengal, India in 2004. He had a brief stint of nine months with Tata Consultancy Services (TCS), India as an Assistant Systems Engineer - Trainee before he joined FSU.

Rajarshi has co-authored and presented a paper at an IEEE research conference. He is a fan of the Java™ programming language and keeps himself current on the enterprise-wide specification from Sun Microsystems Inc.