The Design and Development of an Undercover Multipurpose Anti-Spoofing Kit (UnMask)

Sudhir Aggarwal, Jasbinder Bali, Zhenhai Duan, Leo Kermes,
Wayne Liu, Shahank Sahai, and Zhenghui Zhu
Florida State University, Tallahassee, FL 32306
{sudhir, bali, duan, kermes, liu, sahai, zzhu}@cs.fsu.edu

Abstract

This paper describes the design and development of a software system to support law enforcement in investigating and prosecuting email based crimes. It focuses on phishing scams which use emails to trick users into revealing personal data. The system described in this paper, called the Undercover Multipurpose Anti-Spoofing Kit (UnMask), will enable investigators to reduce the time and effort needed for digital forensic investigations of email-based crimes. A novel aspect of UnMask is its use of a database to not only store information related to the email and its constituent parts (such as IP addresses, links, domain names), but also to organize a workflow to automatically launch UNIX tools to collect additional information from the Internet. The retrieved information is in turn added to the database. Reports can then be automatically generated according to the needs of the forensic investigator, including correlations across multiple email data stored in the database. UnMask is a working system. To the best of our knowledge, UnMask is the first comprehensive system that can automatically analyze emails and generate forensic reports that can be used for subsequent investigation and prosecution.

1. Introduction

This paper describes the design and development of a software system to support law enforcement in investigating and prosecuting email based crimes. It focuses on *phishing scams* [1], which is the use of email to drive users to spoofed websites using technical exploits and social engineering to trick users into revealing personal data (e.g., passwords, social security numbers and credit cards numbers). Once these data are (illegally) captured, they are typically used to commit a number of more serious cybercrimes, such as fraud, identity theft and hacking (unauthorized access and theft of services). The system described in this paper, called the *Undercover Multipurpose Anti-Spoofing Kit (UnMASK),* will enable investigators to reduce the time and effort needed for digital forensic investigations of phishing e-crimes and can also be used for forensic investigations of other crimes that use emails as a vector, such as threats and harassment.

Investigating incidents of phishing and the related problem of identity theft tend to be labor-intensive tasks that produce lots of dead-ends and few tangible results [2, 3, 4]. Automated methods of following leads would reduce the effort, training, and resources dedicated to the investigation of such email crimes. UnMask is a user-friendly system for parsing email header and body to produce an actionable evidentiary trail that law enforcement investigators can use to develop viable leads for the cases they are investigating.

An important feature of the UnMask project is that a database is a central aspect of not only keeping track of the initial phishing emails under investigation, but also the mechanism to store subsequent information searched after deconstruction (parsing with a view to determining important components related to the investigation) of the email. Thus, for example, we automatically launch UNIX tools such as whois, dig and traceroute to determine further information about IP addresses in the header and the body of the email and store this retrieved information back into the database. Once the complete information related to an email is obtained, UnMask can generate reports that provide details about the email's trajectory, a summary of the content, factual and forged IP addresses, pointers, linkages, discrepancies, etc. To the best of our knowledge, UnMask is the first comprehensive system that can automatically analyze emails and generate forensic reports that can be used for subsequent investigation and prosecution.

Since each email that is considered is stored in the database, UnMask can effectively be used to answer queries related to multiple emails, such as discovering if similar source addresses were used during a particular period of time across the set of stored emails. The functionality of correlating multiple emails is particularly helpful and crucial for law

enforcement because investigators often need to process a batch of emails, seized via warrant, subpoena, or court order, that were sent to and received by the owner of a specific email address throughout a definite period of time.

UnMask is a working system, complete except for hardening of the code. Our next step is to deliver it to law enforcement for experimental use. We have been working with both the National White Collar Crime Center and the Florida Department of Law Enforcement in building UnMask. We envision that, once validated, UnMask will augment investigation reports composed by human investigators and be viewed as reliable in terms of impartiality and consistency, thus meeting a legal standard for admissibility of evidence [5]. These reports then can be trusted and used in future legal proceedings, such as requesting search warrants or supporting court subpoenas to further the investigations or entering as case evidences to conclude the prosecutions.

This paper is organized as follows. In section 2, we discuss some background related to our development effort and some related work. In section 3, we present the high-level architecture of the UnMask system. Section 4 presents the detailed implementation of two key aspects of UnMask: (1) the use of a PostgreSQL database and a novel use of triggers to create a workflow manager; and (2) the automation of the use of UNIX tools to automatically retrieve additional desired information from the Internet and store it into the database. In section 5, we discuss how reports are automatically generated and give examples of such use in an investigation. Section 6 concludes with a brief discussion on future work.

2. Related work

Many phishing attacks involve impersonating web sites and emails; yet the majority of countermeasures focus only on the former, and not the latter. Browser plug-in tools such as SpoofGuard [8], SpoofStick [9], and Trustbar [10] often either fail to detect a phishing web site or fail to convey their detections in a more convincing way [11]. The efficacy of more elaborate user interface enhancements such as Dynamic Security Skins [12], PassMark [13], and Web Wallet [14] is also limited, as they either rely on users to make the final judgment based on some visual differences, or, require broad knowledge about the legitimacy of a great number of sites which is unlikely to be feasible.

SiteWatch [15] is a two-pronged anti-phishing solution that checks both emails and potential phishing web sites. Whenever it finds a suspicious URL in an email, it forks a separate process to compare the potential phishing page against the real one and assesses the visual similarities between them in terms of key regions, page layouts, and overall styles. Similarly, but taking a more offense-centric approach, Phoney [16] uses a set of fake information that is submitted to the possible phishing site to gauge the site's response. And, preemptively taking this offense-centric approach even further, in [17] Microsoft uses a pipeline of automated "monkey programs" running on virtual machines with varied patch levels to search for web sites that exploit browser vulnerabilities. Even when initial investigation failed to reveal the identity of the spammer/phisher, Microsoft reports how the perpetrator can still be pursued by filing a "John Doe" lawsuit [18] and following up with thorough third-party discovery.

Those preemptive and aggressive approaches can be useful in preventing phishing crimes but may not be very useful in computer crime investigations. Other approaches at the email level can also be effective for early prevention. For example it may be easier to detect the anomaly of an email and prevent the phishing target from responding compared to verifying the legitimacy of a web site. A great deal of work related to prevention has been done, mainly related to spam messages. Areas of research include learning and mining technologies in email classification [19, 20]; rules finding and patterns matching [21, 22]; and statistical and probabilistic determination [23]. The UnMask work, however, is focused on digital forensic investigations and evidence gathering. Thus, much of the current research focusing on spam filtration is orthogonal to our research goals.

Tools or websites, such as Sam Spade [24] or domaintools [25] share a similar goal to ours. These tools are used interactively to various degrees by the law enforcement community. These tools/websites, as well as UnMask, provide network-query functionality that lets users probe domain names, IP addresses, etc. Sam Spade, for example, lets user crawl websites to pull out a list of email-addresses/links. These tools also let users analyze email headers to determine whether the email message was sent from a valid address or forwarded via an open relay to cover the sender's tracks. However, these tools expect some reasonable networking expertise from law enforcement. More importantly, they do not sufficiently automate the work nor do they provide a database for further analysis.

SPARTA's Phisherman project [26] is more closely related to UnMask, in that both employ a database as central repository. However, the Phisherman project is a global effort to simply collect and archive data related particularly to phishing scams

and disseminate this data to its subscribers. In contrast, our goal is to help users in a more direct way, i.e., provide them an automated tool to process emails. In the future, there may be a utility in integrating our work with the Phisherman project services. Our capability of correlating multiple emails across multiple users or retrieving past cases can be easily scaled for global applications and some integration of UnMask with Phisherman may be an appropriate direction, after experience with the deployment of UnMask at law enforcement agencies.

Organizations such as the Anti-Phishing Working Group (APWG) [1] provide public up-to-date statistics and maintain an extensive collection of phishing email messages. Through special arrangement with APWG, we have a subset of their phishing email collection and are using these emails in our test of UnMask. The anti-phishing research group at Indiana University [6] conducts surveys and research to understand behavioral aspects related to phishing. Studies of phishing scams [7] illustrate some of the idiosyncrasies of the current email system.

3. Overview of UnMask

In this section we present a high-level overview of the UnMask system. We will present a more detailed implementation of UnMask in Section 4.

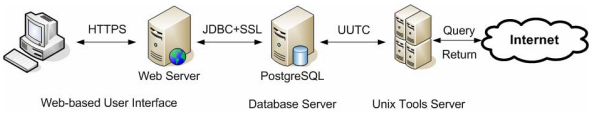

Figure 1: Overview of the UnMask System

As a system to automate and facilitate email investigations for law enforcement, UnMask must support the functionalities to adequately parse a submitted email and automatically collect detailed forensic information about the email. Figure 1 presents the basic architecture of the UnMask system. As shown in the figure, UnMask consists of three key components: a web-based user interface, a database system, and a UNIX Tools system. Users interact with UnMask via the web-based user interface to perform forensic tasks such as submitting an email message for analysis or generating email investigation reports.

The database system glues all components of UnMask together. In particular, it parses and stores the submitted email messages, and interacts with the UNIX Tools system to gather detailed forensic information about the email messages. The UNIX Tools system component provides the basic forensic toolkit to implement Internet email investigation. In the rest of this section, we describe in further detail the individual components and their interactions.

3.1 UnMask web-based user interface

UnMask users interact with the system via a web-based user interface. UnMask allows users to perform a variety of automatic email investigation tasks. Users can submit a specific email message for analysis; they can also cross-examine or correlate emails with certain properties by querying the system. By default, when a user submits an email message, the UnMask system conducts email header and body forensic analyses similar to the current common practice of law-enforcement investigators. For example, UnMask helps identify the ISP and its contact information associated with the email sender. UnMask greatly reduces the complexity in email forensics by automating this email investigation process. When the analysis of the submitted message is done, the user can obtain a forensic report of the message via the same user interface. Alternatively, the user can also specify an email address to which the report should be delivered. The web-based interface supports organizing investigations into cases and facilitates maintaining meta-information such as jurisdiction and investigator information.

In addition to submitting and analyzing individual messages, UnMask users can also cross-examine or correlate multiple messages by querying the system. For example, users may wish to obtain forensic information of all messages related to the same investigation case or all messages from the same sender within a certain period of time. Via the web interface, users may access the system from the internal networks of law-enforcement agencies, or through the public Internet. Given that users may access UnMask via the public Internet, it is required that the connection between the web-based user interface and the UnMask database system be highly secure. The web interface that we use is adapted from a project that was designed as a highly secure environment for investigators to upload password files to a code-breaking system [27].

3.2 UnMask database system

At the heart of the UnMask system is a database system that glues all the components of the system together. We chose to use the PostgresSQL [28] database because of several features of use to us, including triggers and stored procedures, and the fact that it is freely available and well supported. The database system implements three key functionalities to automate and facilitate the email forensic efforts of

law-enforcement investigators. First, when a message is submitted via the web interface, the message is fully parsed to obtain all the atomic elements of the message such as email addresses, mail server domain names, mail server IP addresses, URLs contained in the message body, and so on [29]. These atomic elements, along with the raw submitted message, are stored in the database.

Second, the database system instructs the UNIX Tools system (see Section 3.3) to launch the proper forensic tools to collect further information associated with the message. The interaction between the database and the UNIX Tools system is initiated through an innovative use of the "trigger mechanism" of PostgreSQL, in conjunction with a simple protocol we have implemented (called UUTC) between the PostgreSQL database and the UNIX Tools system. The database system can thus, for example, automatically signal the UNIX Tools system to gather the location and contact information of the ISPs in charge of the mail servers along the message delivery path to aid the email investigation. Such forensic information obtained by the UNIX Tools is in turn stored back into the database system.

Third, the database system also provides mechanisms for supporting cross-examination and correlation of email messages. For example, users may wish to obtain forensic information of all messages related to the same investigation case or all messages from the same sender within a certain period of time. Such cross-examination of email messages, however, is constrained by proper jurisdiction so that one investigator does not get access to information of another investigator unless properly authorized to do so.

3.3 UnMask UNIX tools system

The UnMask UNIX Tools system provides the basic forensic toolkit for email investigation. Based on the current common practice of law enforcement in email investigations, the UNIX Tools system provides the following basic functionalities: 1) mapping between domain names and IP addresses; 2) identifying the DNS and mail servers associated with a domain; 3) identifying the contact information of the person(s) or organization responsible for maintaining an IP address or domain; 4) verifying the validity of email addresses; and 5) reachability of and routes to an IP address or domain.

The UNIX Tools system runs as a background daemon on a UNIX machine waiting for service requests from the database system. Upon receiving a request from the database system, the UNIX Tools system will perform the corresponding action(s) and return the results to the database system, where the results may be further parsed and stored. For better performance, multiple UNIX Tools machines may be deployed in an UnMask system.

4. Detailed software architecture

In this section, we describe the functionality of two important aspects of the UnMask system: the database server component, including the parsing of the emails; and the gathering of information from the Internet by the UNIX Tools component. Figure 2 below shows the interaction of the Website, the PostgreSQL database and the UNIX Tools system.

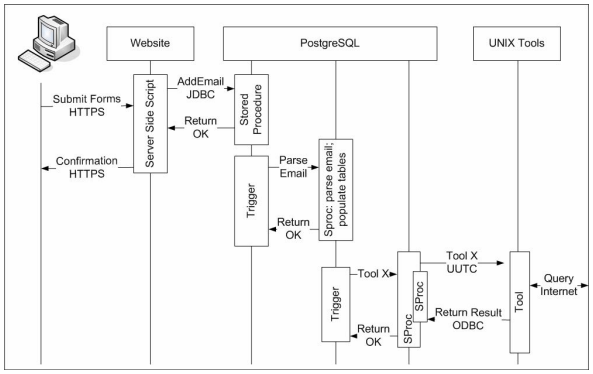

Figure 2: UnMask Architecture

4.1 The database server

In implementing the UnMask system we chose to use the PostgreSQL 8.2 database. Our requirements for a database were the ability to: (1) store all email related data after parsing it to an appropriate level of granularity and (2) mechanisms to invoke a toolkit of various UNIX tools like traceroute, dig, whois, etc. to retrieve additional information related to the email from the Internet.

4.1.1 Why PostgreSQL

We chose PostgreSQL over other relational database management systems because it is free / open source and it has excellent support for many features including the following:

Native Interfaces for Procedural Languages: PostgreSQL allows user defined functions to be written in programming languages besides the native PL/pgSQL. Currently supported languages include Perl and C. We extensively use Perl in our database programming.

Triggers and Stored Procedures: Triggers are a mechanism to invoke (user-defined) functional

processing initiated by database commands such as inserting a record. PostgreSQL has excellent version control of stored procedures (functions) and the procedures can be written in many languages.

ACID Transactional Capabilities: PostgreSQL has excellent support for atomic transactions. One can do a "begin transaction" and then issue commands such as "create table", and "alter table" with the assurance that concurrent transactions will work properly. The correct support for handling DDL (Data Description Language) statements within a transaction was very important to us.

4.1.2 Email parsers

In UnMask, we iteratively parse an email into finer and finer granularity. We have organized our parsing related code as several (wrapper) parsers described below. These parsers are used to deconstruct the raw email, analyze email header fields and email body, and extract specific components from the email such as IP addresses or machine domain names from subparts of the email. These parsers are written in Perl and are based on freely available email and HTML parsing packages from the CPAN website [30]. The email is deconstructed in several stages.

Raw Email Parser: This parser is used to first deconstruct the raw email into various header fields and the body according to the message formats defined in [29].

Email Address Parser: This parser is used to extract email addresses found in header fields. Examples of such header fields are: "*From*", "*To*", "*Cc*", "*Bcc*", and "*Sender*."

Received Field Parser: This parser is used to deconstruct "*Received*" fields in a header to extract mail relay server information. There may be several "*Received*" fields because each mail relay server will add its own information in the header.

Body Parser: This parser is used to analyze HTML code and plain ASCII text in the email body and extract email addresses and links. In the future we are also planning to analyze other scripting systems such as embedded Java code and Macromedia Flash.

4.2 UnMask database design

Our database is designed so that tables that contain the raw email and deconstructed components of the email are "write once." This helps in maintaining an evidentiary trail for subsequent prosecution. Header fields that allow multiple instances (such as several *Received* or *Resent* fields) are maintained as separate tables. All database inserts and retrievals take place using functions to avoid SQL injection [31]. When inserts occur they can initiate other database activities through the use of triggers. Activities can include parsing fields of records in tables, initiating a connection to the UNIX Tools Server and entering new records into tables. Specific tables are used to store data that is returned from actions of the UNIX Tools Server.

The connection between the database and the UNIX Tools server is through a new protocol that we designed and implemented called the UnMask UNIX Tools Connection (UUTC) protocol. This protocol opens a socket connection when needed to a daemon process (the Unix Tools server) and allows parameters needed for invoking specific tools to be sent across the connection, and permits return information to be properly put back into the database (using a separate ODBC connection).

Using Figure 3, we give a simple example of the sequence of events that occur in the database when an investigator creates a case and uploads an email. Our goal is to illustrate how various records are stored in the database and how triggers cause appropriate events to happen. The law enforcement investigator is able to create cases through our user interface. Whenever a new case is created, law enforcement information (such as investigator name, jurisdiction, etc.) is stored in the *tbl_users* table (not shown in Figure 3). When an email file (in eml format) is uploaded through the user interface, the entire email text is first stored as a field in the *tbl_email* table along with other fields such as a unique ID created by the system called *unmask_id*. As soon as a record gets inserted in the *tbl_email* table, trigger 1 (*trg_email*) is fired that in turn invokes a function *sp_email* written in PL/Perl which parses the email and stores appropriate components in various level 2 tables such as *tbl_l_header* and *tbl_uri*.

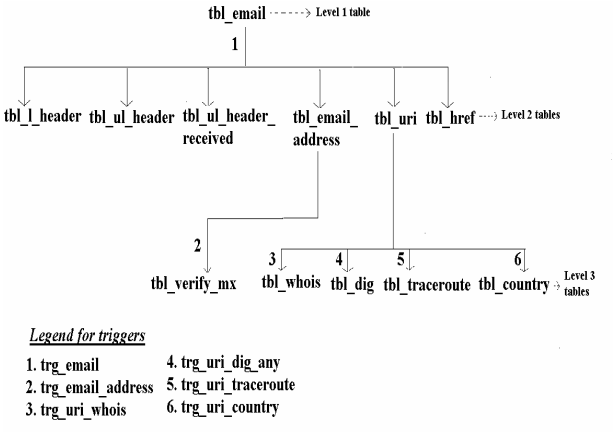

Figure 3: Tables, Triggers and Dataflow

Level 2 tables can similarly trigger further actions. An insert in *tbl_email_address*, for example, invokes trigger 2 (*trg_email_address*) which in turn calls a function that would open a connection with the UNIX Tools server using the UUTC protocol. Results are returned and stored in level 3 tables such as *tbl_verify_mx*.

4.3 UNIX tools server

The UNIX Tools server is a daemon that runs programs (tools) invoked by the database. The daemon and the database communicate through the UUTC protocol. Programs can be wrappers around utilities such as whois, dig, and traceroute, or more complex programs designed for specific tasks such as verifying an email address. Some of the tools that we have developed are shown in Table 1, which also shows the parameters required by each tool. Parameter *unmask_id* is a digital ID generated by our database. Parameters *domain* and *local_name* are the domain name and local user name respectively of an email address. *Source* indicates the source of the email address in the parsed email. *Dns_server* is an optional parameter sent to the tool server. If this is provided, the tool server will run the dig command for the specified DNS server else it will use a default value. Parameter *host* refers to a numerical IP address or a canonical host name.

Tool Name	Parameters	Function
tool1	*unmask_id domain local_name source [dns_server]*	Run "*dig*" to find the mail servers of the domain, and then ESMTP VRFY, HELO to verify the email address at one of the mail servers.
tool2	*unmask_id host*	Run "*traceroute*" to find reachability and routes to an IP address or canonical host name.
tool3	*unmask_id domain*	Run "*whois*" to find registration data for a domain.
tool4	*unmask_id domain source [dns_server]*	Run "*dig*" to get full DNS information
tool5 (uses a package called *IPGEO* [32])	*unmask_id host*	Find geographical location (currently only country) of an IP address or a canonical host name

Table 1: UnMask Tools

The flexibility of our design is that it allows other tools to be easily developed and incorporated into the system. Note for example that because of security concerns, ESMTP VRFY usually provides no information. This aspect of Tool1 could easily be dropped but since it is automated and might in some cases be useful, we might as well obtain its results. More complex investigative tools can be incorporated as they are designed.

5. Reports and analysis in UnMask

The user interface for UnMask supports a case management system for uploading of email files for analysis as well as generation of reports based on information stored in the database. UnMask uses a password-based user access control. In order to submit an email, the user first logs into the system. The user can submit an email file (in eml format) by browsing for it locally and uploading it as part of a new or an existing case. After the email is deconstructed and processed as discussed in Section 4, the user is able to view the generated reports. Figure 4 illustrates the case management screen of the Unmask user interface. The three cases that user *liu* is investigating are listed, and further information on each can be accessed by clicking on the case name.

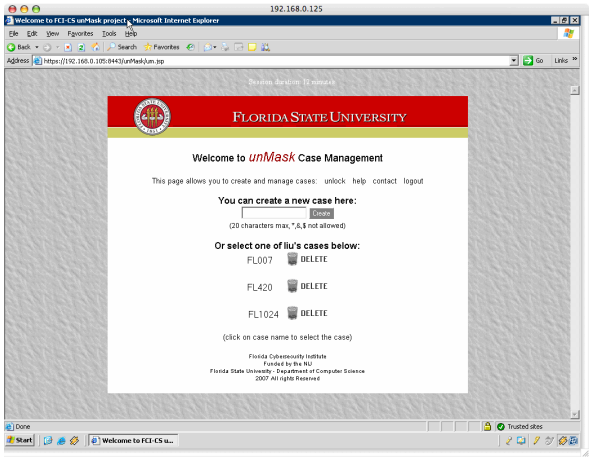

Figure 4: UnMask User Interface

The implementation of the user interface is through an interactive web-based infrastructure rendered using dynamic web pages written in Sun Microsystems' Java Server Pages (JSP) technology. The application logic in a JSP page uses the Java Database Connectivity (JDBC) API to create dynamically-generated HTML output from the contents of the database by providing a call-level API for SQL-based database access. Also within a JSP page we have HTML code that displays static text and graphics. When the page is displayed in a user's browser, it contains both static HTML content and

dynamic information retrieved from the database about his or her specific case.

5.1 Requirements for a report

Reports are designed to support law-enforcement in analyzing email components. For example, the sender email address in the raw email being analyzed may have been forged, or a URL in the rendered email may be redirecting the recipient to a website different from what is commonly inferred from its name. As part of requirements analysis, a brief survey was done to ascertain what investigators would ideally like to see in a report. Some of these desired features were determined to be:

For each email address found in the phishing email, determine the MX record for its domain, and also the results of executing ESMTP EXPN and ESMTP VRFY on the mail server. It must be clearly mentioned as to what field (i.e., "*From*", "*Cc*", "*Bcc*", etc.) the particular email address was found in.

Determine the IP address of the originating machine, and run the network utilities traceroute, dig, and whois on this address.

For each IP address/URL specified anywhere in the body of the raw email, again run the aforementioned network utilities.

Also, we implemented a simple select query which retrieves all email messages in the database which have the same *"From"* field email address as the investigated email. General message correlation functionality will be implemented in the next version of UnMask.

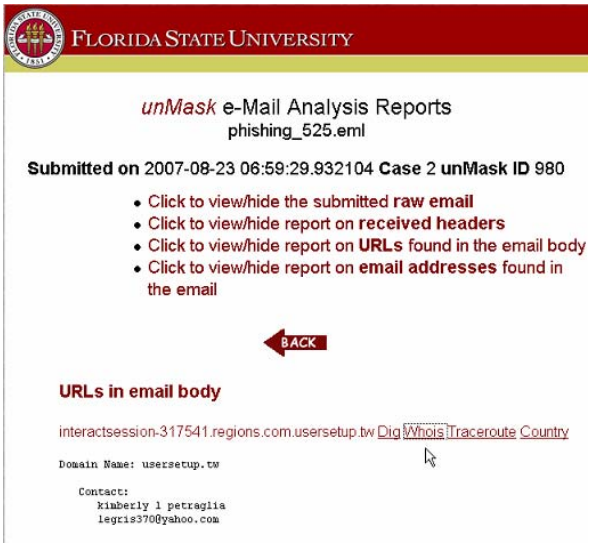

Figure 5: Segment of a Report

The reports that UnMask generates include all the above information organized in a structured fashion discussed in the next subsection. See Figure 5 for a portion of a report illustrating analysis of the email *phishing_525.eml*. The report contains links to sub-reports on received headers, URLs, and email addresses. The lower half of the figure shows a URL found in the body followed by links to further analyses (*Dig, Whois, Traceroute, Country*). The *Whois* link has been clicked and shows the registration information of the URL.

5.2 Report Structure

A report follows the structure of an email message. Starting with email header information the report shows the specific header fields isolated for clarity and coupled with information gathered by the UNIX tools. This additional information expands the investigator's understanding of that field. For example the trace fields "*Received:"* would appear with an analysis of the sending and receiving mail hosts (IP address, domain name, traceroute result, DNS and whois records, etc).

As mail hosts (represented by name or IP address), email addresses, website links, and other items appear in the report at different sections of the email, so does the information gathered about these items. This provides the investigator with as much information as possible and aids in the decision making process on what forensic leads to follow further.

5.3 UnMask reports: an example

To have a better understanding of the UnMask report structure and how it may be used by law enforcement, we present an example section of a report that provides detailed forensic information on the "*Received:*" fields in an email header. Each email message carries in its header a set of "*Received:*" fields (the set can be empty), which collectively describe the routes that the message takes from the sender to the recipient of the message at the mail relay server level. It is important to note that, in order to mislead the recipient (or investigator) of an email message about where the message originates, it is common for the (spam and phishing) message sender to forge the first few "*Received:*" fields. However, the set of "*Received:*" fields in a message still contains a portion of the true path that the message takes as part of the route shown in the set of "*Received:*" fields, so it provides valuable investigation information for law enforcement.

As discussed above, an UnMask report follows the structure of an email message. For each field, we provide additional forensic information gathered by the Unix Tools system. In particular, for each "*Received:*" field, we first extract the domain names and IP addresses of the mail relay servers appearing in the field. To aid the law enforcement investigation, we then determine the location and contact information of the organization (or person) that is responsible for a domain name (or IP address), the route to the mail relay server, and the IP address of a domain name (and vice versa), along with other information that we collect, by launching the corresponding Unix tools. Discrepancies discovered during the analysis of the "*Received:*" fields can easily be noted by an investigator.

The following snippet is an example "*Received:*" field from an email message that we received, which contains two domain names (`walking14.legessermon.com`, `mx.google.com`) and one IP address (`64.192.31.14`). For each of them, UnMask collects proper forensic information by launching the corresponding tools. The information shown was collected within one day after we received the message.

```
Received: from walking14.legessermon.com
(walking14.legessermon.com [64.192.31.14])
     by mx.google.com with ESMTP id
e18si15752160qbe.2007.05.30.10.46.13;
     Wed, 30 May 2007 10:46:24 -0700 (PDT)
```

Figure 6 shows a snapshot of the report section related to the domain name `walking14.legessermon.com` found in the example "*Received:*" field (the snapshot only captured a segment of it). In this section of the report, UnMask displays location and contact information of the organization that is responsible for the domain name (partially shown in the figure), the MX and DNS records for the corresponding domain, the route to the domain name, and the IP address of the domain name, among other things. The corresponding IP address returned by the dig tool for this domain was `64.192.31.2`, which is different from the one listed in the "*Received:*" field for the aforementioned domain name. Note however that no strong conclusion can be drawn from this discrepancy since the two IP addresses are on the same subnet.

UnMask collected similar information about the IP address `64.192.31.14` and the domain name `mx.google.com`. We do not discuss these results further due to space limitation. Note however that the location and contact information returned from probing the IP addresses tend to be more long-lived and reliable than the ones from probing the domain names. Domain names (especially for phishing sites) and their associated registration information tend to be short-lived. However, IP address allocation is normally delegated to ISPs nowadays and is quite stable. Three days later we re-ran the tools to generate another report on the message, containing the domain name `walking14.legessermon.com` and its associated registration/contact information. The resulting information turned out to be the same; however if it would have been different then further investigation may have been warranted.

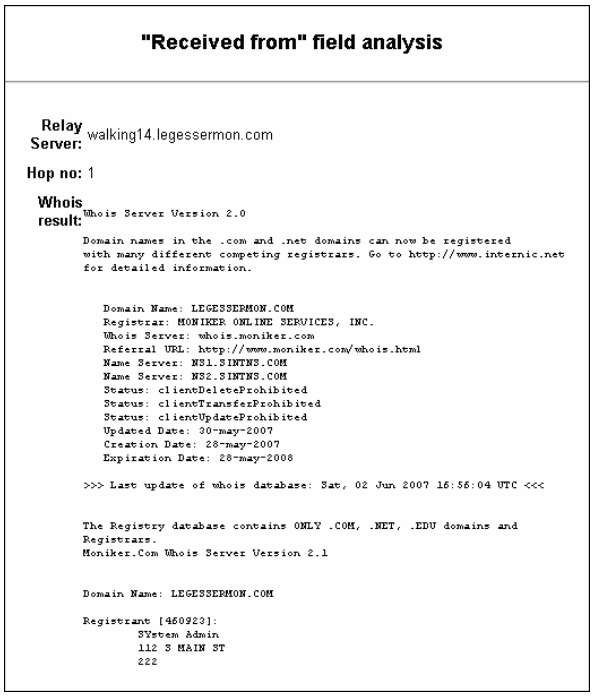

Figure 6: Part of header field report

5.4 Extending UnMask capabilities

While our current reports only present the information in the email analyzed and the additional information gathered by the UNIX tools, logical analysis of the data could be further incorporated into the report. Since PostgreSQL is a relational database, we can apply predicate logic to the relations in our database.

In our current use of the database we generate reports simply by collating the information gathered and present it in a meaningful format to the user, but we could do much more by applying formal logic and data mining. For example, we could data mine to identify companies which are enablers for phishers. These companies intentionally facilitate the misuse of resources (such as DNS registrations, site hosting, etc) by not enforcing policies and standard procedures, or worse. We could potentially gather vital statistics on

phishing scams identified with these companies. We could then query the database using predicate logic statements to cluster emails which are part of larger crimes. A more concrete example would be to structure a query that counts the number of unique URLs found in the bodies of all emails in the database of UnMask. Then we could count the number of URLs in this set registered at each Registrar (Go Daddy, eNom, Network Solutions, Tucows, etc.) in order to determine if any Registrar is associated with a significantly higher-level phishing activity. This information could be used to persuade Internet Corporation for Assigned Names and Numbers (ICANN) to investigate to see if there is cause for concern. As the system is used, the techniques used by investigators to interpret the data will evolve. The logic used can be folded back into the report, so the system will not only automate the task of human gathering of data, but also automate deductions about the data.

6. Conclusion

This paper has described the design and development of the UnMask system for supporting law enforcement in investigating phishing email crimes. The system allows a law enforcement investigator to upload a suspect email via a secure user interface and then get a report with detailed information about the email including data obtained from the internet through automated searches launched during the deconstruction (parsing) of the email. A novel aspect of UnMask is its use of a database to not only store information related to the email and its constituent parts (such as IP addresses, links, domain names), but also to organize a workflow to automatically launch UNIX tools to collect additional information from the Internet. The retrieved information is in turn added to the database. To the best of our knowledge, UnMask is the first comprehensive system that can automatically analyze emails and generate forensic reports that can be used for subsequent investigation and prosecution.

The version 1 of the UnMask is a working system, completed except for bullet proofing and hardening of the code. The functionality is completed and is as described in this paper. We next intend to have law enforcement use our system on an experimental basis. Using the feedback the investigators provide, we intend to add additional features and search tools, and increase the facility of investigators in determining the exact information they wish to gather.

For example, we plan to incorporate checking for black-listed sites to see if the sending host or MX server is listed as a rogue machine and to inspect websites with automated crawlers to search for investigative clues. Such searching for websites, however, is often not desired by law enforcement because they do not want to "tip off" possible rogue sites before having the necessary authority to shut them down. We also plan to incorporate a logic analysis module that will provide intelligent filtering and probability assessment of the retrieved data from the UNIX tools to make the investigator's job of detecting the important information (such as a suspected spoofed source address) easier.

7. Acknowledgements

This work was supported in part by the National Institute of Justice under grant 2005-MU-MU-K007 and grant 2006-DN-BX-K007. We wish to thank NW3C (Bob Hopper and Nick Newman) and FDLE (Mike Phillips and his group in the Computer Crime Center) for their invaluable feedback, help and support on this project.

8. References

[1] Anti-Phishing Working Group. http://www.antiphishing.org/

[2] Phishing and Federal Law Enforcement. Referenced 5/29/07, http://www.abanet.org/adminlaw/annual2004/Phishing/PhishingABAAug2004Rusch.ppt.

[3] President's Information Technology Advisory Committee (PITAC) (2005). Cybersecurity: A Crisis of Prioritization, Report to the President. Posted 2/28/2005, http://www.nitrd.gov/pitac/reports/20050301_cybersecurity/cybersecurity.pdf.

[4] Law Enforcement Battles with Botnets. Referenced 5/29/07, http://government.zdnet.com/?p=2373.

[5] Daubert v. Merrell Dow Pharmaceuticals, Inc. *509 U.S. 579* (1993).

[6] The Anti-Phishing group at Indiana University. http://www.indiana.edu/~phishing/.

[7] Bob Breeden, Mike Cantey, Brett Cureton, Clifford Stokes, Peter Henry, Judie Mulholland, Wayne Sprague, and Jim Watson. The Phlorida Autopsy Report. *Digital Forensic Practice, Journal of*, 1(3):203-222, 2006.

[8] Y. Teraguchi N. Chou, R. Ledesma and J.C. Mitchell. Client-side defense against web-based identity theft. In *11th Annual Network and Distributed System Security Symposium (NDSS '04)*, San Diego, CA, USA, February 2004.

[9] SpoofStick. http://www.spoofstick.com.

[10] TrustBar. http://www.cs.biu.ac.il/~herzbea/Papers/ecommerce/spoofing.htm.

[11] Min Wu, Robert C. Miller, and Simson L. Garfinkel. Do security toolbars actually prevent phishing attacks? In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 601-610, New York, NY, USA, 2006. ACM Press.

[12] Rachna Dhamija and J. D. Tygar. The battle against phishing: Dynamic security skins. In *SOUPS '05: Proceedings of the 2005 symposium on Usable privacy and security*, pages 77-88, New York, NY, USA, 2005. ACM Press.

[13] PassMark. http://www.passmarksecurity.com.

[14] Min Wu, Robert C. Miller, and Greg Little. Web wallet: preventing phishing attacks by revealing user intentions. In *SOUPS '06: Proceedings of the second symposium on Usable privacy and security*, pages 102-113, New York, NY, USA, 2006. ACM Press.

[15] Wenyin Liu, Xiaotie Deng, Guanglin Huang, and A.Y. Fu. An antiphishing strategy based on visual similarity assessment. *Internet Computing, IEEE*, 10(2):58-65, March-April 2006.

[16] Madhusudhanan Chandrasekaran, Ramkumar Chinchani, and Shambhu Upadhyaya. PHONEY: Mimicking User Response to Detect Phishing Attacks. In *2006 International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM'06)*, pages 668-672, 2006.

[17] Yi-Min Wang, Doug Beck, Xuxian Jiang, Roussi Roussev, Chad Verbowski, Shuo Chen, and Samuel T. King. Automated web patrol with Strider HoneyMonkeys: Finding web sites that exploit browser vulnerabilities. In *Proceedings of the Network and Distributed System Security Symposium, NDSS 2006*, San Diego, CA, USA, 2006. The Internet Society.

[18] Aaron E. Kornblum. Searching For John Doe: Finding Spammers and Phishers. In *CEAS 2005 - Second Conference on Email and Anti-Spam*, July 2005.

[19] Joshua Goodman, Gordon V. Cormack, and David Heckerman, Spam and the ongoing battle for the inbox. *Commun. ACM*, 50(2), February, 2007.

[20] Aleksander Kolcz, Abdur Chowdhury, and Joshua Alspector. The Impact of Feature Selection on Signature-Driven Spam Detection. In *CEAS 2004 - First Conference on Email and Anti-Spam*, Mountain View, CA, USA, July 2004.

[21] William W. Cohen. Learning rules that classify email. In *Proceedings of 1996 AAAI Spring Symposium on Machine Learning in Information Access (MLIA '96)*, 1996.

[22] Isidore Rigoutsos and Tien Huynh. Chung-Kwei: a Pattern-discovery-based System for the Automatic Identification of Unsolicited Email Messages (SPAM). In *CEAS 2004 - First Conference on Email and Anti-Spam*, Mountain View, CA, USA, July 2004.

[23] Paul Graham. A Plan for Spam. http://www.paulgraham.com/spam.html, 2002.

[24] Sam Spade. http://www.pcworld.com/downloads/file/fid,4709-page,1/description.html

[25] DomainTools. http://www.domaintools.com.

[26] Phisherman, SPARTA, Inc. http://www.isso.sparta.com/documents/phisherman.pdf.

[27] Sudhir Aggarwal, Daniel Beech, Rajarshi Das, Breno de Medeiros, Eric Thompson. X-Online: An Online Interface for Digital Decryption Tools. *Proceedings of the 2nd Int. Workshop on Systematic Approaches to Digital Forensics Engineering (SADFE 2007)*, April 2007.

[28] PostgrelSQL, http://www.postgresql.org.

[29] P. Rensnick, "Internet Message Format", RFC 2822. April 2001.

[30] Comprehensive Perl Archive Network. http://www.cpan.org/.

[31] SQL Injection. http://en.wikipedia.org/wiki/SQL_Injection.

[32] IPGEO Tools. http://www.ipgeo.com.