Covert Channel Forensics on the Internet: Issues, Approaches, and Experiences

Ashish Patel, M. Shah, Rajarathnam Chandramouli, and Koduvayur P. Subbalakshmi
(Corresponding author: K. P. Subbalakshmi)

Department of Electrical and Computer Engineering, Stevens Institute of Technology
Hoboken, NJ 07030, USA
E-mail: ksubbala@stevens.edu

(Received Oct. 21, 2005; revised and accepted Dec. 3, 2005 & Jan. 27, 2006)

Abstract

The exponential growth of the Internet (WWW in particular) has opened-up several avenues for covert channel communication. Steganographic communication is one such avenue. Hiding secret messages in digital data such as images using steganographic software tools is becoming easier. These digital images posted in public Web sites can then be downloaded at the receiver and the hidden messages may be extracted securely. To thwart covert channels on the Internet new types of search engines that can identify, detect and track these channels are necessary. Traditional search algorithms will fail to identify these channels. In this paper, we discuss various key issues involved in developing a stego (forensic) Web search engine. We also propose approaches to address some of these issues. Finally, we discuss a prototype forensic search engine that we developed called STEALTH and discuss in detail its architecture. Some experimental results are also reported.

Keywords: forensics, multimedia, security, steganalysis, steganography, Web search

1 Introduction

A covert channel is any communication channel that can be exploited by a process to transfer information in a manner that violates the system security policy [19]. One of the earliest works on this was done by Lampson [10] who defines a covert channel as a channel that is not intended for information transfer. The theoretical dangers of covert channels were first addressed in the National Computer Security Center's (NCSC) - Trusted Computer System Evaluation Criteria (TCSEC) as early as 1983 and 1985.

Covert channel is a simple yet very effective mechanism for sending and receiving information between two parties on a network without triggering any firewall and intrusion detection system (IDS) on the network. Covert channels bypass traditional security mechanisms by hiding information in seemingly benign data (e.g., control fields in the TCP and IP headers [14], digital images etc.). Covert channels are of two types: (a) timing channel and (b) storage channel. Timing channels carry covert message by modulating the response time of a system. Messages are stored in spatial locations in storage channels. Steganography (stego) is an example of a storage channel. Here, a secret message is hidden in digital data such as digital images, graphics, audio files, and streaming video, using a secret key. The receiver of the message then extracts the hidden message using the same key. Further, availability of various freeware and shareware steganographic tools facilitate their ease of use. Throughout this paper we will assume that the message carriers are digital images even though the concepts discussed in this paper are applicable to other types of digital data as well.

Message hiding using steganography has its pros and cons. Digital watermarking is an application of steganography for copyright protection. On the other hand, secret communications using steganographic algorithms could be used for malicious purposes. Further, the Internet has facilitated the proliferation of steganographic content. It is easy for someone to post plain looking digital images that contain hidden messages on their Web page. One receiver or multiple receivers can then download these images and extract the hidden message. Clearly, identifying and detecting the Web pages that contain covert messages is of immense interest and importance to digital forensics experts.

Consider the following examples of the potential threat caused by covert channels:

- A worm such as MyDoom [11] can be hidden inside a digital image and passed through a target network. Upon extraction from the image, this worm can hunt for e-mail addresses. Note that a variant of MyDoom also succeeded in knocking a number of smaller search engines, including Lycos and AltaVista, off the Web completely.

- Terrorists may use steganography to conceal information in digital images and post them on a public Web page such as ebay.com. The receiving party can then download these images and extract the hidden information.

Current Web search engines such as Google, Yahoo, etc. and a wide variety of intrusion detection systems (IDS) are incapable of searching or identifying Web pages that contain steganographic messages. Web crawlers search and accumulate data from the publicly indexable Web pages. These crawlers are not efficient enough to search for the hidden Web or invisible Web. Moreover, most search engines are still text based. Also, metrics used by current search engines such as the popularity of Web links, relevance etc. may not work for forensic search. We note that current Web search engines are not capable of identifying and searching multimedia content containing hidden steganographic messages. This is because they do not incorporate steganalysis algorithms that can identify steganographic content, rely on long history of Web links while steganographic Web pages may only live for a short period of time for stealth reasons, etc.

Considering the immense growth of the Web, exhaustive search of the Internet for stego content is clearly infeasible. Some of the key questions we identify are the following:

- Where to look: How to identify Web links that could potentially contain covert messages?

- What are the fundamental design issues in developing a forensic search engine?

- How to look: How to exploit network traffic to detect stego Web pages?

- Exploiting side information: Message carrying Web sites may have not a public link. Then how to exploit side information such as http traffic request in the back bone to identify these hidden links?

- Time scale: Stego Web pages may be created, moved and destroyed randomly on a daily basis; short-lived Web pages.

- Scalability: A Web-page like e-bay could contain thousands of multimedia files, i.e., images, audio, etc. What are the efficient search techniques to employ?

- Steganalysis: How do we integrate steganalysis with forensic Web search?

In this paper, we attempt to address some of these questions. We discuss a forensic search architecture called STEALTH. The main goal of STEALTH is to identify Web pages with stego information. A prototype implementation of STEALTH is also presented. Some other closely related work can be found in [3, 5]. A detailed mathematical model for Web search steganalysis as an optimization problem is provided in [5]. Finally, we note that to our knowledge STEALTH is the first stego Web search engine.

The rest of this paper is organized as follows. Section 2 gives a brief overview of covert communication using steganography. Section 3 discusses some of the key design issues in developing a forensic Web search engine and some possible approaches. Architecture of the proposed stego search engine, STEALTH, is discussed in Section 4. Section 5 presents some conclusions and future work.

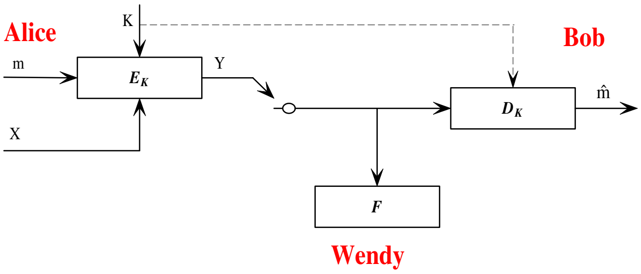


Figure 1: Prisoner's problem model for covert communication

2 Overview of Steganography and Steganalysis

The Prisoner's problem [15], an abstract model of covert communication using steganography is shown in Figure 1. Here, Alice (sender) and Bob (receiver) are prisoners in two different cells. They hatch a plan to escape by covertly communicating with each other using a storage channel (e.g., embedding a covert message in a plain looking digital image). It is assumed that Alice and Bob share a secret key (K) *a priori* that they use for encoding (E_K) and decoding (D_K) a covert message (m). X denotes the cover/host signal that carries the covert message. Each output (stego signal) $Y \in \Re$ from Alice's encoder is first examined by a passive adversary/warden, Wendy. Wendy runs a steganalysis algorithm (F) on the encoder's outputs to determine if it contains a covert message. If the output of the steganalysis algorithm is $F(Y) = 1$ then Wendy detects a covert channel and therefore punishes Alice and Bob. However, if $F(Y) = -1$ she decides that there is no covert message and allows Y to be received by Bob. Then Bob decodes a message ($\hat{m}$) from Y.

Steganalysis is a relatively new branch of research. While steganography deals with techniques for hiding information (such as fingerprinting), the goal of steganalysis is to detect and/or estimate potentially hidden information from observed data with little or no knowledge about the steganography algorithm and/or its parameters. It is fair to say that steganalysis is both an art and a science. The art of steganalysis plays a major role in the selection of features or characteristics a typical stego message might exhibit while the science helps in reliably testing the selected features for the presence of hidden information. While it is possible to design a reasonably good steganalysis technique for a specific steganography algorithm, the long term goal must be to develop a steganalysis framework that can work effectively at least for a class of steganography methods, if not for all. Clearly, this poses a number of challenges and questions.

We classify steganalysis into two categories [4]:

- Passive steganalysis: Detect presence/absence of hidden message in a stego signal, identify the stego embedding algorithm.

- Active steganalysis: Estimate the embedded message length, estimate location(s) of the hidden message, estimate the secret key used in embedding, estimate some parameters of the stego embedding algorithm, extract the hidden message.

During the steganalysis process the steganalysis detector may exploit *spatial diversity* and *temporal diversity* information:

- Spatial diversity information based steganalysis: Steganalysis methods can look for information in the spatial domain that repeats itself in various forms in different spatial locations (e.g., different blocks within an image or, in different images).

- Temporal diversity information based steganalysis: Steganography information that appears repeatedly over time can also aid steganalysis.

3 Issues and Approaches in Designing a Stego Search Engine

In this section we identify some of the key issues in designing a stego Web search engine and propose approaches to alleviate some of these problems.

3.1 Key Issues

The Web is becoming more and more complex while its size continues to grow at remarkable rate with an estimated 7 million new pages being created every day [13]. Hence, there are several important issues that need to be addressed while designing a stego search engine. We list some of the key issues below.

- Coverage: The rapid growth and large volume of the Web poses unparalleled challenges in scale for current general-purpose crawlers and search engines. Even though the coverage area for stego search is confined to images and other multimedia contents, scale is still a big problem. Web sites like e-bay, Bazee (e.g. [2]), etc. could contain millions of digital images, and searching this collection for covert stego messages is a computational challenge. It is believed that 7 billion new Web pages are added to the Internet every second [13]. Of this, Google covers 8.1 billion pages [8] which is by far one of the largest totals of any search engines today. But, this is only a fraction of the total publicly index-able Web pages. It is impossible for current general-purpose Web crawlers to search the entire Web due to its scale and other design difficulties. This problem is further exacerbated in stego search since the goal is to identify covert message channels that may be difficulty to identify in the first place.

- Search metrics: Current search engines use different search metrics applied to public links. These search metrics do not apply to identify links with covert messages as these Web sites may not have public links and also these engines are not integrated with steganalysis tools. For example, most search engines like Google [8] combine page rank with text-matching techniques to find relevant and efficient search results. Most search engines give significant importance to link popularity. The more number of sites pointing to a particular site the higher its popularity. Obviously, Web sites that carry hidden message may not have links to other Web sites for security reasons.

- Dynamic Web: The dynamic nature of the Web is an important issue. The Web contains billions of documents with Web sites, which are created, modified, moved, and destroyed on a daily basis, perhaps even randomly. Therefore conducting a trend analysis on certain Web sites would be difficult due to their random life times. Building a probabilistic model for randomness of these dynamic Web pages is a big challenge. This is especially true for stego Web sites because as soon as the receiver downloads and extracts the hidden message the sender may destroy the Web site for security reasons.

- Time scale: Covert channels are short-lived. There are difficulties in designing forensic search engines for finding such short-lived covert communication channels. With the advent of high speed Internet and other computing facilities, potential parties can easily communicate covertly within few moments. Digital images can be uploaded and downloaded by communicating parties in no amount of time. In such a scenario, the question is how to identify and trace such short-lived channels and how often to scan for them?

- The hidden Web: Detecting covert communication on some password protected forum and other service provider sites is a critical issue. There are many sources of information on the Internet, which are not accessible or "visible" to standard Web search engines. Either the files are not publicly available or because the pages are not stored as individual files, instead they are created on the fly or dynamically as the information is requested. This invisible portion of the Web is estimated to be many times larger than the portion, which is indexed by the general search engines such as Google or Yahoo. The types of material that are hidden includes: databases, sites requiring registration, archives (e.g., newspapers and magazines), dynamically created Web pages and interactive tools. A major problem for stego and other search engines is the design of a crawler to extracting contents from the hidden or invisible Web sites. Some message carrying Web sites may not be publicly linked so that only the intended parties can upload their images or covert messages in the hidden Web pages. Hence, forensic search engines techniques are needed to identify their existence.

3.2 Design Approaches

As stated earlier one of the major challenges in detecting stego content on the Internet is the sheer volume of data — Web pages, Web requests, dynamic Web etc. Therefore, developing good heuristics to identify URLs of interest in order to limit the search effort is important. These heuristics will allow the engine to easily track Web pages of interest efficiently. A generic procedure for the search framework that we use is as follows:

1) Save the URLs from HTTP requests in log files.

2) Some Web sites (e.g., .gov for government and .mil for military) can be safely eliminated before the search begins by assuming that the security of these sites are not compromised. Test URLs to see if they are safe sites from safe domains by using filters. If the filter outputs suggests that the URL is safe then it does not need to be sent to the steganalysis module for further processing.

3) Non-safe URLs are sent to the steganalysis module periodically. This period of time is called the sweep time, and can be set by the administrator. Clearly, the choice of sweep time affects the search accuracy vs. efficiency trade-off. A large sweep time results in a larger volume of collected URLs and may produce more accurate results after filtering.

   URLs that are not already in the database are added with the current timestamp. If the URL already exists and the timestamp is less than the sweep time then the timestamp is not updated; otherwise it is. In every sweep cycle (say 24 hours) the URLs that have timestamps less than a day old are sent to the steganalysis module.

Some additional heuristics to identify, collect information and search URLs for stego information that we have implemented and tested in STEALTH are the following:

- Using the number and frequency of http requests: Sites that are popular will have a lot of requests. Generally, it would be unlikely that these would be used for malicious purposes, so we can consider them as “known” sites. A URL would achieve this status if it accumulates a certain number of Web requests over a period longer than the sweep cycle (say a week). If a URL is known, then it will not be tested for a certain amount of time (say a month) after that. However, popular sites may be taken down after some time necessitating a stego check later on. These known sites will have an entry in the log with fields for the URL name and the timestamp when the URL was included in the log file.

- Spike in http requests: This criterion also has to do with tracking the number of http requests. A malicious user may put up a page with a secret stego message and then take it down after a day or sooner, after the receiver downloads the hidden message. The Web activity of these kinds sites are bursty — no activity for a while, then a spike in activity when the message is received, and then no activity again. Therefore, when a page is updated we keep track of how many requests to access that page was seen in the last 24 hours (or sweep time). A spike in the maximum page access statistics in a day could indicate suspicious activity such as covert communication. This site will be marked for further processing and tracking. All the multimedia data (image etc.) from this Web site will be automatically downloaded and sent to the steganalysis module.

- Monitoring Web server log files: When a user accesses a Web site, all transactions between the users browser and the Web server software are logged in ASCII format in server log files. These files provides some useful side information from a data analysis point of view, since it indicates how users are arriving at the sites, what kind of browser software, source/destination IP address is being used, the last time the page was modified, the type of content that exists in the page, the size of the page, and its server type and/or image type (jpg, tiff, etc.). This information can be utilized at a later point for forensic analysis to detect the source of the covert communication.

- Back links and implicit covert links: Back links are the links going out to other pages from a given Web site. The total number of back links and the association of those back links with suspicious sites will provide useful information to map the links associated with the covert communication. For example, a given Web site URL may not be explicitly suspicious but one or some of the back links that are associated with this link may be in the set of suspicious sites. This means that the given Web site is an implicitly suspicious link.

- Use of side-information: Web sites of groups with radical, politically or religiously opposed views, religious cult’s Web sites and other information are useful for identifying candidate Web sites. Other side-

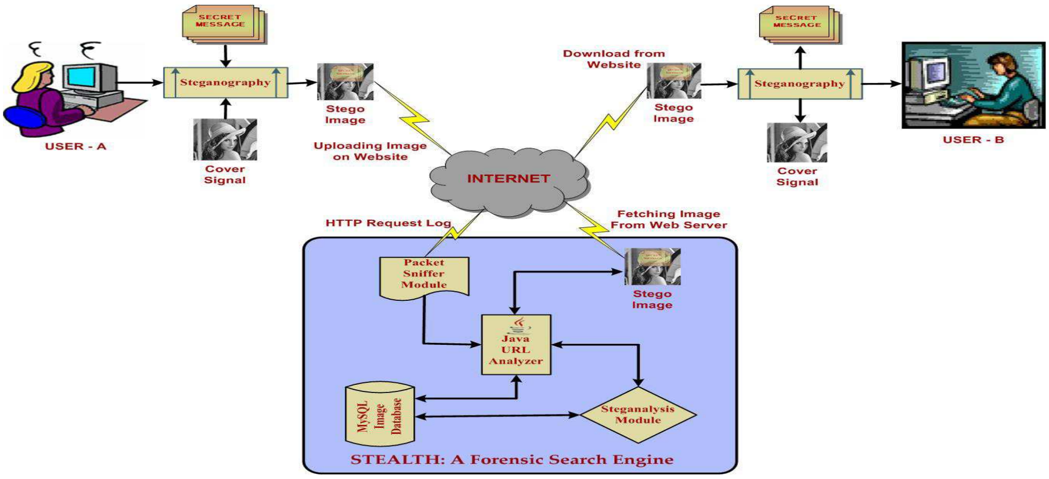


Figure 2: The STEALTH architecture

information provided by external intelligent information such as e-mail tracing, tapping phone conversation, etc. could also be used to select certain Web sites for close monitoring.

- Word analysis in URL hyperlink: Analyzing URLs for key words such as bomb, anarchy, terror, hack, etc. also helps to identify candidate Web sites for further search. These pages would be analyzed outside of the steganographic setting.

4 STEALTH Architecture and Prototype Implementation

Figure 2 shows the architecture of the proposed stego Web search engine, STEALTH. A basic scenario of user A (secret message sender) uploading stego image on a Web site and another user B (secret message receiver) downloading that image for message extraction can been seen in this figure. The STEALTH engine consists of four main modules: (a) network packet sniffer module; (b) Java URL analyzer; (c) MySQL image database; and (d) steganalysis module, as illustrated. The information flow between these four modules to detect a stego channel can be seen in Figure 2. The functionalities of these modules are described next.

4.1 Network Packet Sniffer Module

We use Ethereal [7] packet sniffer for tracking http requests on the backbone network. Ethereal is a freeware software released under GNU general public license. It allows live data to be captured and read in Ethernet, FDDI, PPP, Token-Ring, IEEE 802.11, classical IP over ATM, loop back interfaces, etc. We filter the captured packets only for the http requests (http GET requests). Ethereal is installed on one of the main server PC in our (MSyNC) research laboratory. This PC monitors and captures incoming and outgoing traffic log between the lab network and the Web.

The schematic of test bed set-up is shown in Figure 3. This figure shows the different nodes connected to the Internet via switches. A typical HTTP request from a node in the lab is captured by a a server as shown. Captured log files demonstrate the activity report of the network traffic with detailed header field information for each transaction. This log is analyzed later for detecting covert communication that might have occurred in subnetwork. Note that, this concept can be extended to monitor Internet traffic on a large scale by installing packet capturing tool on Internet routers and then re-assembling those log files at some server for later analysis.

4.2 MySQL Image Database

MySQL, MySQL Administrator, PHP, and phpMyAdmin are system metrics used for database setup. MySQL is the database software we use for this work. The current schema for the database tables is illustrated below.

- URL requests: This database table stores all URL requests with source/destination IP address, time stamp and total number of http requests and index of associated Web-path. If same links are visited more than once, duplication is handle by updating last modified time and updating total number of visits.

- Web-path: Whenever a URL is passed to the URL-Analyzer it runs a check to find out if the URL is already in the database. If the URL is not found

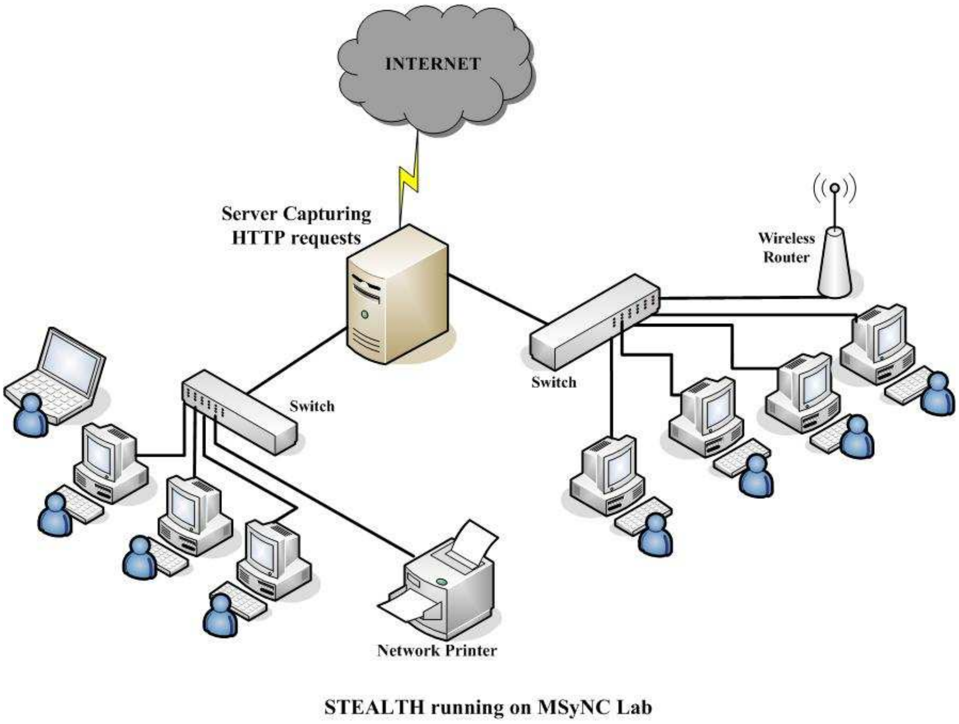


Figure 3: Stego search experimental test-bed set-up

in the database an unique index is assigned to that URL and all the Web paths related to that URL will use same index. Hence, using that index it is easy to collect all Web paths belonging to that URL. This can then be used to compute the depth of the URL. Hence, filtering is made easier for various Web paths of same URL.

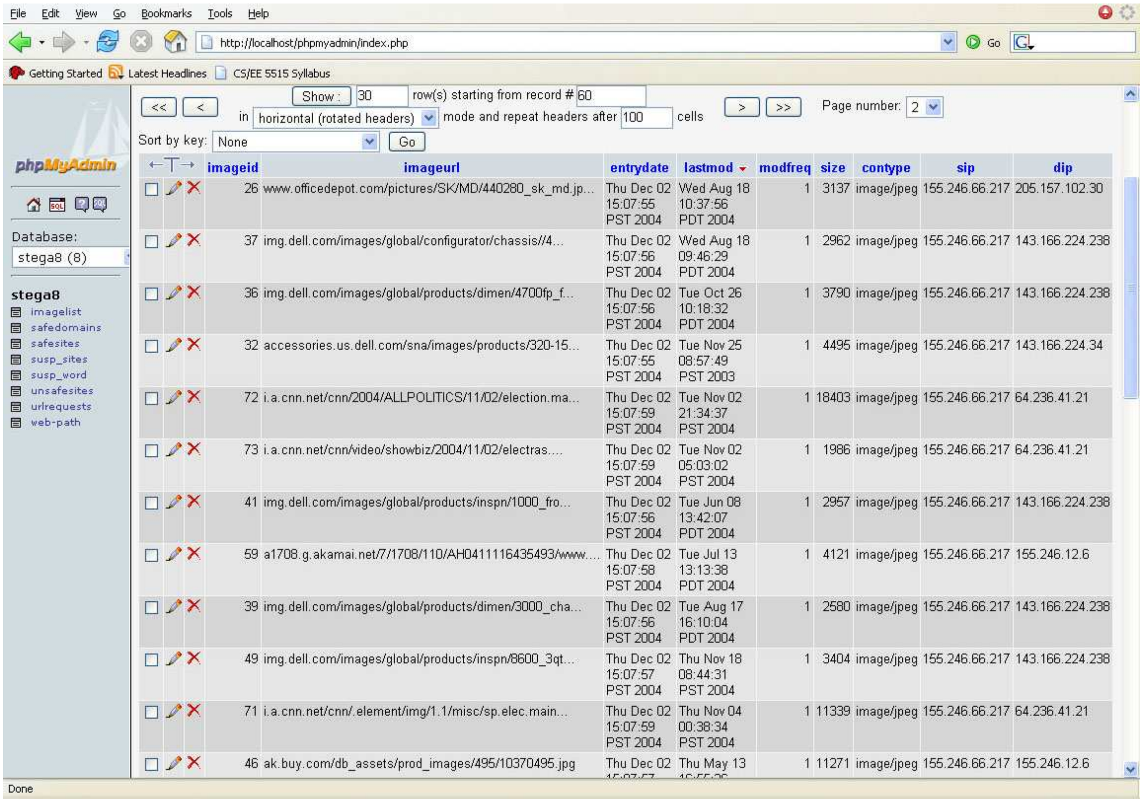
- Safe domains: This is a table containing a list of domains which we assume are safe and do not need to be analyzed by steganalysis module. Currently, we assume government domains .mil and .gov, are safe. But, this list can be easily extended in the current prototype implementation. We first check whether the URL's domain is listed in safe domain table. If the domain is listed here then the URL will be stored in safe sites table. And that URL will not be used for future analysis. This helps to eliminate some of the sites for searching and alleviates the scale problem to some extent.

- Suspicious words in URL: In a database schema there is "susp_word" table, which contains the following list of suspicious words: bomb, hack, al qaeda, forum, terrorism, and warez. Web URL crawler will look for these keywords in the hyperlink. If any of the the suspicious words are found in the URL then it will be stored in "susp_list" table.

- imagelist table: All URLs related to images are stored in this table. All the images downloaded from the sites will be stored in the directory on the local drive. These images are used by the steganalysis module for detecting steganographic contents. If a stego message detected by the steganalysis algorithms(s) then all the associated properties for those images will be updated to unsafesites table. *imagelist* table contains the following fields: image URL, image entry date, last modified date, modified frequency since entry, image size, content type, source IP, and destination IP. A sample visual representation of the imagelist table with test entries is shown in Figure 4.

- Suspicious sites: After performing various analysis as described previously the URLs are stored in this table. Figure 5 shows an example table with suspicious sites and the reason the sites were triggered as suspicious. Clearly, the triggering procedure could lead to some errors (false positives and false negatives). Figure 5 shows some false entries for Google, Yahoo and Stevens homepage because of their higher http request frequency when compared to other overall frequency history. However, we can minimize such false positive or false negative alarm easily by modifying safe domain table and adjusting sweep cycle time in the URL analyzer.

- Unsafe sites: The steganalysis module analyzes all the images in the local drive folder. If any suspicious image is found then the related information is stored in unsafesites table. These URLs will then be monitored continuously by the URLAnalyzer to check for



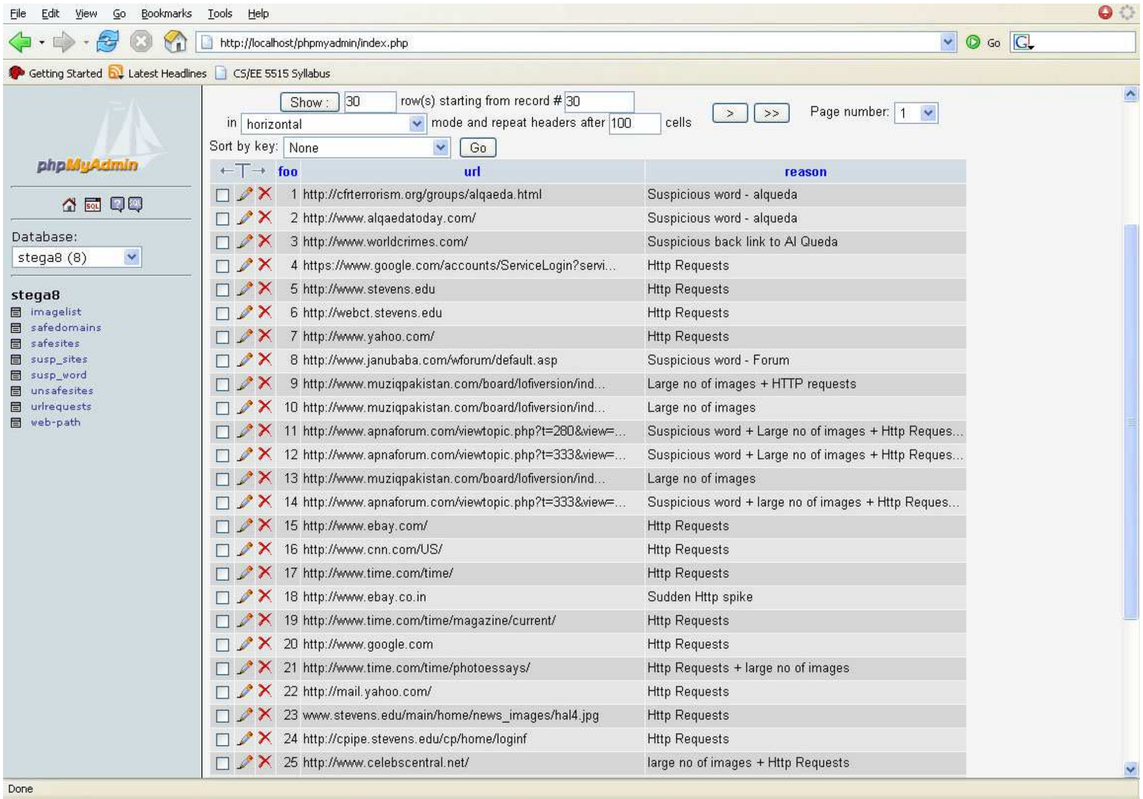
Database: stega8 (8)

Show : 30 row(s) starting from record # 60 in horizontal (rotated headers) mode and repeat headers after 100 cells Page number: 2

Sort by key: None

imageid	imageurl	entrydate	lastmod	modfreq	size	contype	sip	dip
26	www.officedepot.com/pictures/SK/MD/440280_sk_md.jp...	Thu Dec 02 15:07:55 PST 2004	Wed Aug 18 10:37:56 PDT 2004	1	3137	image/jpeg	155.246.66.217	205.157.102.30
37	img.dell.com/images/global/configurator/chassis//4...	Thu Dec 02 15:07:56 PST 2004	Wed Aug 18 09:46:29 PDT 2004	1	2962	image/jpeg	155.246.66.217	143.166.224.238
36	img.dell.com/images/global/products/dimen/4700fp_f...	Thu Dec 02 15:07:56 PST 2004	Tue Oct 26 10:18:32 PDT 2004	1	3790	image/jpeg	155.246.66.217	143.166.224.238
32	accessories.us.dell.com/sna/images/products/320-15...	Thu Dec 02 15:07:55 PST 2004	Tue Nov 25 08:57:49 PST 2003	1	4495	image/jpeg	155.246.66.217	143.166.224.34
72	i.a.cnn.net/cnn/2004/ALLPOLITICS/11/02/election.ma...	Thu Dec 02 15:07:59 PST 2004	Tue Nov 02 21:34:37 PST 2004	1	18403	image/jpeg	155.246.66.217	64.236.41.21
73	i.a.cnn.net/cnn/video/showbiz/2004/11/02/electras....	Thu Dec 02 15:07:59 PST 2004	Tue Nov 02 05:03:02 PST 2004	1	1986	image/jpeg	155.246.66.217	64.236.41.21
41	img.dell.com/images/global/products/inspn/1000_fro...	Thu Dec 02 15:07:56 PST 2004	Tue Jun 08 13:42:07 PDT 2004	1	2957	image/jpeg	155.246.66.217	143.166.224.238
59	a1708.g.akamai.net/7/1708/110/AH0411116435493/www....	Thu Dec 02 15:07:58 PST 2004	Tue Jul 13 13:13:38 PDT 2004	1	4121	image/jpeg	155.246.66.217	155.246.12.6
39	img.dell.com/images/global/products/dimen/3000_cha...	Thu Dec 02 15:07:56 PST 2004	Tue Aug 17 16:10:04 PDT 2004	1	2580	image/jpeg	155.246.66.217	143.166.224.238
49	img.dell.com/images/global/products/inspn/8600_3qt...	Thu Dec 02 15:07:57 PST 2004	Thu Nov 18 08:44:31 PST 2004	1	3404	image/jpeg	155.246.66.217	143.166.224.238
71	i.a.cnn.net/cnn/.element/img/1.1/misc/sp.elec.main...	Thu Dec 02 15:07:59 PST 2004	Thu Nov 04 00:38:34 PST 2004	1	11339	image/jpeg	155.246.66.217	64.236.41.21
46	ak.buy.com/db_assets/prod_images/495/10370495.jpg	Thu Dec 02	Thu May 13	1	11271	image/jpeg	155.246.66.217	155.246.12.6

Figure 4: An example of imagelist table obtained from experimentation with the prototype



Database: stega8 (8)

Show : 30 row(s) starting from record # 30 in horizontal mode and repeat headers after 100 cells Page number: 1

Sort by key: None

foo	url	reason
1	http://cfrterrorism.org/groups/alqaeda.html	Suspicious word - alqueda
2	http://www.alqaedatoday.com/	Suspicious word - alqueda
3	http://www.worldcrimes.com/	Suspicious back link to Al Queda
4	https://www.google.com/accounts/ServiceLogin?servi...	Http Requests
5	http://www.stevens.edu	Http Requests
6	http://webct.stevens.edu	Http Requests
7	http://www.yahoo.com/	Http Requests
8	http://www.janubaba.com/wforum/default.asp	Suspicious word - Forum
9	http://www.muziqpakistan.com/board/lofiversion/ind...	Large no of images + HTTP requests
10	http://www.muziqpakistan.com/board/lofiversion/ind...	Large no of images
11	http://www.apnaforum.com/viewtopic.php?t=280&view=...	Suspicious word + Large no of images + Http Reques...
12	http://www.apnaforum.com/viewtopic.php?t=333&view=...	Suspicious word + Large no of images + Http Reques...
13	http://www.muziqpakistan.com/board/lofiversion/ind...	Large no of images
14	http://www.apnaforum.com/viewtopic.php?t=333&view=...	Suspicious word + large no of images + Http Reques...
15	http://www.ebay.com/	Http Requests
16	http://www.cnn.com/US/	Http Requests
17	http://www.time.com/time/	Http Requests
18	http://www.ebay.co.in	Sudden Http spike
19	http://www.time.com/time/magazine/current/	Http Requests
20	http://www.google.com	Http Requests
21	http://www.time.com/time/photoessays/	Http Requests + large no of images
22	http://mail.yahoo.com/	Http Requests
23	www.stevens.edu/main/home/news_images/hal4.jpg	Http Requests
24	http://cpipe.stevens.edu/cp/home/loginf	Http Requests
25	http://www.celebscentral.net/	large no of images + Http Requests

Figure 5: An example of suspicious sites table obtained from experimentation with the prototype

any further modifications and activity in these Web pages.

4.3 URLAnalyzer

URLAnalyzer is a software program written in the Java language which processes and analyzes the log files data. URLAnalyzer automatically scans for information like host URL, Webpath, time, source IP and destination IP. This information is then stored in the database for later use by the steganalysis module.

A high level pseudocode for the proposed Web crawler for URLanalyzer is shown in Algorithm 1. This algorithm essentially takes the log file as input. It then starts to collect the URLs from the log files along with the associated information on the Web path, source IP and destination IP.

Algorithm 1 Pseudocode for URL analyzer

```
Require: $logfile$
  $Readlogfile \rightarrow URLAnalyzer$
  while $EndofFile \neq 0$ do
    $databse \leftarrow URL + Webpath$
    $database \leftarrow SourceIP$
    $database \leftarrow DestinationIP$
    if $URL \in Safedomain$ then
      $update \leftarrow safesites$
    else
      $update \leftarrow URLrequest$
      $update \leftarrow Webpaths$
    end if
    if $URLhassuspiciouswords$ then
      $update \leftarrow suspiciouslist$
      $update \leftarrow Imagelist$
    end if
    $appyanalysis \rightarrow suspicioussites$
    $applysteganalysis \rightarrow imagelist$
    $update \leftarrow unsafesites$
  end while
```

The steps implemented by the URL analysis module is shown in the flow diagram in Figure 6. The process starts from scanning the log file URLanalyzer.java which collects the Web path, source IP and destination IP. URLanalyzer first checks whether the URL is safe or not using the safe domain table. If the captured site domain is in the safe sites domain then the URL is stored in the safe sites table. All other URLs are stored in URLRequest table. If the URLRequest already exists in the database, then the last modified date and total number of http requests fields will be updated in the URLRequest table. Multiple links with the same URL are filtered and associated Web paths are stored in separate Web-path table. Hyperlinks with suspicious words will be stored in susp_list table. Then the different filters such as the total number of requests, suspicious back links, suspicious words in the URL, etc. are applied to this list. Web site which have higher chances of suspicious contents will be stored in unsafesites table based on the filter outputs. URLs for images are stored in imagelist table and the associated images are stored in a folder on the local hard drive and used as inputs to steganalysis tools.

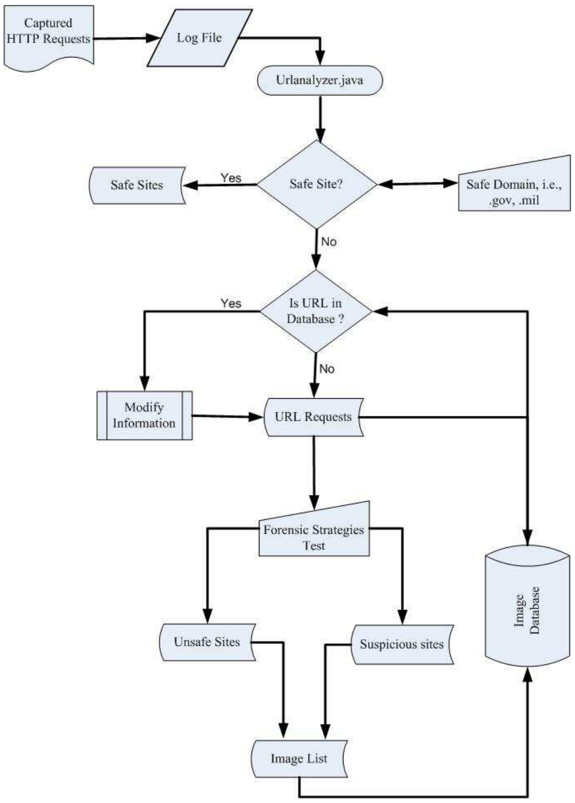


Figure 6: Flow diagram for URLAnalyzer

4.4 Steganalysis Module

While steganography deals with techniques for hiding information the goal of steganalysis is to detect and/or estimate potentially hidden information from observed data with little or no knowledge about the steganography algorithm and/or its parameters. The test-bed currently uses the spread spectrum steganalysis algorithm [18] for detecting stego messages in images. This algorithm is capable of detecting and estimating secrete key used in sequential steganography, the message length and other parameters of the hidden message.

The steganalysis module currently employs a hypothesis testing approach to detect abrupt changes in the statistics of an observed signal (stochastic process) using a sequential probability ratio test. Specifically, a cumulative sum (CUSUM) test [12] is used for detecting change points in the observed stochastic process. This statistical test takes as input one sample at a time and decides on whether a statistical change point has occurred (due to message beginning or ending). The decision error probabilities are traded-off for decision delay.

The steganalysis module observes a sequence of independent, identically distributed samples (random variables) $\{Y_k\}$ (encoder's output) with probability density $p_\theta(y)$ parameterized by θ. In general, θ can be a vector; however, in our current implementation we consider scalar values for θ. The value of θ changes due to message embedding after an unknown value of index k. Therefore let's say, before the unknown change time k_0, the parameter θ is equal to θ_0 and after the change we have $\theta_1 \neq \theta_0$. The steganalysis module then attempts to do the following: (a) detects change in parameter θ and (b) estimates change time k_0. Let's say H_0 is the hypothesis when there is no embedded message (no change) and H_1 corresponds to the hypothesis when message is embedded. That is,

$$\begin{aligned} & H_0 : \theta = \theta_0 \text{ when } k < k_0 \\ & \quad \text{(no embedded message from } k = 0 \text{ to } k_0 - 1) \\ & H_1 : \theta = \theta_1 \text{ when } k \geq k_0 \\ & \quad \text{(message starts at location } k = k_0) \end{aligned} \tag{1}$$

Then, several cases arise:

- θ_0 and θ_1 values are completely known. This case arises as a result of Kerchoff's principle where the assumption is that, the steganographic embedding algorithm is completely known and only the secret key is unknown.

- θ_0 and θ_1 are partially known. A (noisy) estimate of θ_0 and θ_1 may be obtained using a large training set obtained before and after embedding. The steganographic embedding algorithm itself may only be known as a black box (e.g., only the executable code of a steganographic software may be available.).

- θ_0 and θ_1 are completely unknown. This is true for applications such as covert communications where only the stego signal may be available to the steganalysis module with no further knowledge.

We concentrate only on the first two cases in the implementation.

We consider spread spectrum embedding and steganalysis in the current set-up. Let,

$$y_k = x_k + \alpha w_k, \; k = 1, 2, \cdots, N,$$

where $y_k \in \Re$ is the stego signal, $x_k \in \Re$ is the cover signal, $w_k \in \Re$ is the embedded message carrier, and $\alpha > 0$ is the message strength. Also, if $x_k \sim N(0, \sigma_0^2)$ (i.e., Gaussian distributed) and $w_k \sim N(0, \sigma_1^2)$ then $y(k) \sim N(0, \sigma_0^2 + \sigma_1^2)$ assuming x_k and w_k are independent. If the message is embedded from $k = k_0$ to k_1 then,

$$y_k \sim \begin{cases} N(0, \sigma_0^2) & k = 1 \text{ to } k_0 - 1, \\ N(0, \sigma_0^2 + \sigma_1^2) & k = k_0 \text{ to } k_1, \\ N(0, \sigma_0^2) & k = k_1 + 1 \text{ to } N. \end{cases}$$

We assume stationarity or piece-wise stationarity of the observations here. If this assumption is violated in real-life, then some sort of pre-processing may be necessary. In order to perform the hypothesis test (1) we define g_k, for $k = 1, 2, \cdots$, as follows:

$$g_k = \begin{cases} g_{k-1} + s_k & \text{if } g_{k-1} + s_k > 0, \\ 0 & \text{if } g_{k-1} + s_k \leq 0, \\ 0 & \text{if } k = 0, \end{cases}$$

where s_k is the log likelihood ratio. Suppose we have partial knowledge about the possible distribution of σ_1, then in this case we can use Wald's weighting function [20]:

$$\text{(Likelihood Ratio) } LR = \int \frac{p_{\sigma_1}(y/\sigma_1) \cdot p(\sigma_1)}{p_{\sigma_0}(y)} d\sigma_1.$$

If σ_1 is normally distributed with zero mean and variance equal to σ, then

$$\begin{aligned} S_j^k &= \lambda_{j,k} + \ln \Gamma\left(\frac{l_k + 1}{2}\right) - \frac{l_k + 1}{2} \cdot \\ & \qquad \ln\left(\lambda_{j,k} + \frac{1}{2\sigma^2}\right) - \ln(2\sqrt{2\pi}\sigma), \end{aligned}$$

where $\lambda_{j,k}$ and l_k are given by

$$\lambda_{j,k} = \frac{1}{2} \sum_{i=j}^{k} \frac{(y_i)^2}{\sigma_0^2}$$

and

$$l_k = \begin{cases} l_{k-1} + 1 & \text{if } g(k-1) > 0, \\ 1 & \text{otherwise.} \end{cases}$$

Γ denotes gamma function, namely, $\Gamma(n) = (n-1)!$. For detailed experimental results we refer to [18]. We are currently implementing other steganalysis algorithms to be included in this module.

5 Conclusion and Future work

Developing a stego (forensic) Web search engine raises several key issues that are not found in traditional Web search. It is found that using the network traffic information is useful in improving the efficiency and accuracy of a stego search engine. The prototype STEALTH stego Web search engine implements some of the filtering techniques discussed in this paper. Experimentation using this prototype in a laboratory environment has been observed to be feasible.

In the ongoing work we are taking a plug-and-play approach to incorporate a variety of steganalysis tools. This must improve the overall accuracy of the proposed search engine.

We are further developing the different modules within STEALTH. We expect to run extensive experiments on our university's main router. We are also investigating other potential covert channels such as e-mail, FTP, and peer-to-peer communication. Some mathematical techniques to optimally distributed the search engine resources is also being investigated.

In addition to filtering the Web traffic for stego information we also plan to develop and implement a Web crawler to gather steganographic contents on the Internet. Obviously, these crawlers must have the capability to detect steganographic activity on suspicious links and sites in both manual and automated mode. Note that there are many Web sites (e.g., [1, 6, 9, 16, 17, 21]) that keep track of the Web sites of a specified user's choice. These services can track and report changes in the Web sites and deliver the results to the user via e-mail, or, if the user prefers, they can log on to the Web sites and look at the results themselves. Clearly, the techniques used by these monitoring tools can also be exploited for detecting suspicious activities in certain Web sites.

Acknowledgement

The authors thank the reviewers for helpful comments. This work was supported by a U.S. National Science Foundation grant DAS 0242417.

References

[1] aignes, "Website watcher - save time, stay informed", http://aignes.com

[2] Bazee, http://www.bazee.com

[3] J. Bloom, "Smartsearch steganalysis", in *SPIE Conference on Security and Watermarking of Multimedia Contents*, pp. 167-172, 2003.

[4] R. Chandramouli, "A mathematical framework for active steganalysis", *Springer Multimedia Systems Journal*, vol. 9, pp. 303-311, 2003.

[5] R. Chandramouli, "Web search steganalysis: some challenges and approaches", in *Proceedings of IEEE ISCAS, Special Session on Information Hiding*, pp. 576-579, 2004.

[6] Changedetection, http://www.changedetection.com

[7] Ethereal, http://www.ethereal.com/

[8] Google, http://www.google.com

[9] Infominder, http://www.infominder.com

[10] B. W. Lampson, "A note on the confinement problem", *Communication of the ACM*, vol. 16, no. 10, pp. 613-615, 1973.

[11] Mydoom gags google, *Computer Fraud & Security*, no. 8, pp. 1-1, Aug. 2004.

[12] I. Nikiforov and M. Basseville, *Detection of Abrupt Changes*, Prentice Hall, 1998.

[13] D. Poremsky, *Google and other Search Engines*, Peachpit Press, 2004.

[14] C. H. Rowland, "Covert channels in tcp=ip suite", http://www.firstmonday.dk/issues/issue2 5/rowland/

[15] G. Simmons, "The prisoners problem and the subliminal channel", in *Advances in Cryptology*, pp. 51V67, 1984.

[16] Tracerlock, http://www.tracerlock.com

[17] Trackengine, "Right information, right people, right time", http://www.trackengine.com

[18] S. Trivedi and R. Chandramouli, "Active steganalysis of sequential steganography", in *SPIE Conference on Security and Watermarking of Multimedia Contents, Special Session on Steganalysis*, pp. 123-130, 2003.

[19] U. S. D. O. D. , (U.S. Department of Defense), "Trusted computer system evaluation criteria", http://csrc.nist.gov/publications/history/dod85.pdf, 1985.

[20] A. Wald, *Sequential Analysis*, Wiley, 1947.

[21] Watchthatpage, *Your Monitor for Changes on the Web*, http://www.watchthatpage.com

Ashish Patel is a Ph.D. student in the ECE Department at Stevens Institute of Technology. His research interests are in the areas of steganography, steganalysis and Web forensics.

Rajarathnam Chandramouli is an Associate Professor in the ECE Department at Stevens Institute of Technology. His research interests are in the areas of wireless networking and security, multimedia security, and applied probability theory.

Koduvayur P. Subbalakshmi is an Assistant Professor at the Electrical and Computer Engineering department at Stevens Institute of Technology, where she co-founded and co-directs the Multimedia Systems, Networking and Communications (MSyNC) Laboratory. Her research interests lie in the areas of: Information and Network Security, Wireless and Multimedia Networking and Coding. She chairs the Special Interest Group on Multimedia Security, IEEE Technical Committee on Multimedia Communications. She received the Stevens Presidents Research Recognition Award in 2003. She is the Guest Editor of the IEEE Journal on Selected Areas of Communication, Special Issue on Cross Layer Optimized Wireless Multimedia Communications.