410247: Laboratory Practice IV

410244(C) Cyber Security and Digital Forensics

Group 1:

Assignment No 1:

Problem Statement: Write a program for Tracking Emails and Investigating Email Crimes. i.e. Write a program to analyze e–mail header.

Objectives: To track and investigate email crimes.

Theory: Emails play a very important role in business communications and have emerged as one of the most important applications on internet. They are a convenient mode for sending messages as well as documents, not only from computers but also from other electronic gadgets such as mobile phones and tablets. The negative side of emails is that criminals may leak important information about their company. Hence, the role of emails in digital forensics has been increased in recent years. In digital forensics, emails are considered as crucial evidences and Email Header Analysis has become important to collect evidence during forensic process.

An investigator has the following goals while performing email forensics –

- To identify the main criminal
- To collect necessary evidences
- To presenting the findings
- To build the case

Email forensics is the study of source and content of email as evidence to identify the actual sender and recipient of a message along with some other information such as date/time of transmission and intention of sender. It involves investigating metadata, port scanning as well as keyword searching.

Some of the common techniques which can be used for email forensic investigation are:

- Header Analysis
- Server investigation
- Network Device Investigation
- Sender Mailer Fingerprints
- Software Embedded Identifiers

To understand email header fields in Gmail, we take a message of a sender as an example.

```
Delivered-To: paul.friedman@gmail.com
Received: by 10.12.174.216 with SMTP id n34csp2326299qvd;
        Wed, 1 Feb 2017 00:39:09 -0800 (PST)
X-Received: by 10.28.27.14 with SMTP id b14mr1702258wmb.82.1485938349292;
        Wed, 01 Feb 2017 00:39:09 -0800 (PST)
Return-Path: <reply@activetrail.com>
Received: from i2.a01.ms18.atmailsvr.net (i2.a01.ms18.atmailsvr.net.
[91.199.29.18])
        by mx.google.com with ESMTPS id
5si23398790wrr.176.2017.02.01.00.39.08
        for <paul.friedman@gmail.com>
        (version=TLS1_2 cipher=ECDHE-RSA-AES128-GCM-SHA256 bits=128/128);
        Wed, 01 Feb 2017 00:39:09 -0800 (PST)
Received-SPF: pass (google.com: domain of reply@activetrail.com designates
91.199.29.18 as permitted sender) client-ip=91.199.29.18;
Authentication-Results: mx.google.com;
       dkim=pass header.i=@activetrail.com;
       spf=pass (google.com: domain of reply@activetrail.com designates
91.199.29.18 as permitted sender) smtp.mailfrom=reply@activetrail.com;
       dmarc=fail (p=NONE sp=NONE dis=NONE) header.from=gingersoftware.com
X-IADB-IP: 91.199.29.18
X-IADB-IP-REVERSE: 18.29.199.91
DKIM-Signature: v=1; a=rsa-sha256; c=relaxed/relaxed; q=dns/txt;
d=activetrail.com; s=at; h=X-BBounce:X-IADB-URL:Sender:Submitter:X-
Feedback-ID:From:To:Date:Subject:MIME-Version:Content-type:Content-
Transfer-Encoding; bh=GytDyTyaDleCfGk0d7bL4F2bXbTuWsb/xtpIVyVaCRw=;
b=sgh6nUFjt5FC7rBC2BwXFulNuG+k14R7bBsstb4erjtZfTn4z/NPHNhVb4Ax1yXoOgX+
Il6n5SCcXTCkwQdmaxpxt/BzPjWVziBdzU1WichHhPabVFeKctyp6pCjv4+d2FVIiEuxqi
v5dBTcJjXBVpOwU0mqgRceh3pqcvd5Rj4=
```

By analyzing the key parameters in an email header, you can get an idea of how the message traveled from the source to the destination. For instance, you can use the originating IP to find the original sender. For this, you need to examine the first Received parameter in the email header. The first IP address here is the originating IP which is also sometimes presented in the fields X- Originating-IP or Original-IP. Borrowing the same header example:

Received: by 10.12.174.216 with SMTP id n34csp2326299qvd; Wed, 1 Feb 2017 00:39:09 -0800 (PST)

By analyzing the key parameters in an email header, you can get an idea of how the message traveled from the source to the destination. For instance, you can use the originating IP to find the original sender. For this, you need to examine the first Received parameter in the email header. The first IP address here is the originating IP which is also sometimes presented in the fields X- Originating-IP or Original-IP. Borrowing the same header example:

The highlighted portion is the original sender of the message. You can use a free reputation service like SenderBase by Cisco to get the reputation rating of the IP. This can help ascertain whether the email is spam or a phishing attack. Although, bear in mind that if the IP address is private, then it may not fetch any result. Message-ID is another important field

that can be checked during email header analysis for spoofing. You can see it below in the highlighted region.

Reply-To: Newsletter@gingersoftware.com
From: Ginger (noreply@gingersoftware.com)
To: "paul.friedman@gmail.com" (paul.friedman@gmail.com)
Message-ID: (c000e5f41f8f4137a30cab4g6eddcd1e@gingersoftware.com)

- Date: Wed, 01 Feb 2017 10:39:06 +0200
- Subject: Thank you for registering with Ginger!
- MIME-Version: 1.0

Since Message-ID is added by the mail server that processes the email, it can't be altered. Also, message systems often use a date/time stamp, along with the sender's domain name. So, if the domain name in the message ID doesn't match the domain name mentioned in the From: field, then it can suggest a possibility of spoofing. In the example above, the From: field shows the domain name gingersoftware.com, which is same as the Message ID. So, it's safe to assume that no spoofing took place here. Before you can analyze an email header, you first need to obtain it. Here‘s how to do that in Gmail:

- Open Gmail.
- Find the message you want to analyze.
- Click the three vertical dots in the upper-right of the message.
- Click —Show Original.

Here, you‘ll see a brief breakdown of the information found in the header. If you scroll down, you can also see the full text of the email header, which will end just before the body content of the message. From there, you can copy the email header into an email header analysis tool to learn more details.
How to Analyze Email Headers in Outlook?

Here‘s how to obtain your email header in Outlook:

1. Open Microsoft Outlook.
2. Click on the message you want to analyze.
3. Click on the dropdown arrow in the upper-right of the message.
4. Click —View message details.

Here, you‘ll see the full text of the email header.
From there, you can copy the email header into an email header analysis tool to learn more details.

Email Header Analyzer Tools
Once you have a copy of the email header, you can analyze it using one of the following email header analysis tools.

Almost all of these tools are free, and function in the same way, so I won‘t go into detail describing the minor differences. With all of them, you‘ll copy and paste your email header, click a button, and review the information after it is parsed.

1. G Suite Toolbox Messageheader.
If you‘re already using Gmail, you might as well try G Suite‘s Messageheader tool.

2. Mx Toolbox.
Mx Toolbox has a great standalone email header analyzer, as well as detailed information on email headers for the uninitiated.

3. What Is My IP?
We‘ve already credited them for their helpful email examples, but What Is My IP also has a convenient email header analysis tool.

4. Mailheader.org.
There‘s also Mailheader.org, where you can review mail header samples in addition to the header you‘ve selected.

5. Gaijin.
In case you needed more options, you could also try Gaijin.
If the above email header analyzer tools return an error message or if there‘s a bit of information you were unable to find in their analysis, consider manually reviewing the email headers yourself.

Conclusion: Thus we can track and Investigate Email Crimes.

Assignment No 2:

Problem Statement: Implement a program to generate and verify CAPTCHA image.

Objectives: To implement a program to generate and verify CAPTCHA image.

Theory: To generate captchas using Python following are the steps:

Install The Captcha Module

So just like any other program, the very first step is to install the CAPTCHA library. In order to do that open your command prompt and run the following command:

pip install captcha

Steps to Create Captcha Generator in Python

We would try to generate both images as well as audio captchas in this tutorial. Hence, when you are done installing the library, you need to import the ImageCaptcha and AudioCaptcha functions from captcha.image and captcha.audio sub-libraries respectively

1. From captcha.image import Image Captcha
2. From captcha.audio Audio Captcha

Generating Image Captcha in Python :

Let‘s start by creating an Image captcha. We will be taking input about the text that needs to display on the screen from the user and then generate the image captcha for the data.
To create the captcha, we need to create an Imagecaptcha object and then generate the captcha for the data using the generate function. Look at the code below:

1. Img=ImageCaptcha(width = 280, height = 90)text =
2. Input(“Enter the Text for Captcha:”)
3. Cap_data = img.generate(text)

The image is generated but to save the image we need to use the write function using the code below.
Img.write(text,’Sample_Cap1.png’)

C++ code:

We are using rand() to generate CAPTCHA randomly.

1. Create a function generateCaptcha that generate a CAPTCHA of length n Create an empty string captcha to store the generated captcha.

Use rand() function to add characters to the captcha.

2. Now get the user input.

3. Use compare() function to compare the generated captcha with user input.#include

<bits/stdc++.h> using

namespace std;

// function to check user input to generated CAPTCHA bool

check_Captcha(string &captcha, string &user_input){

return captcha.compare(user_input) == 0;

}

// function to generate CAPTCHA of length nstring generateCaptcha(int n){

time_t t; srand((unsigned)time(&t));

char*required_chars="abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUV
WXYZ0123456789"; string

captcha = "";

while(n--)

captcha.push_back(required_chars[rand()%62]);

return captcha;
}

int main(){

int n;

cout<<"Enter the required length of CAPTCHA: ";cin>>n; string

captcha = generateCaptcha(n);

cout<<"CAPTCHA: "<<captcha<<endl; string user_input;

cout<<"Enter the CAPTCHA: ";

```
        cin>>user_input;

        if (check_Captcha(captcha, user_input))

                cout<<"Valid CAPTCHA"<<endl;
        else
                cout<<"Invalid CAPTCHA"<<endl;

                return 0;
}
```

Output:

Enter the required length of CAPTCHA: 6CAPTCHA:

OsBVxh

Enter the CAPTCHA: OsBVxh

Valid CAPTCHA

Enter the required length of CAPTCHA: 8CAPTCHA:

R5y3cVuW

Enter the CAPTCHA: R5y3cVuW

Valid CAPTCHA

Enter the required length of CAPTCHA: 5CAPTCHA:

Y3EgK

Enter the CAPTCHA: y3egk

Invalid CAPTCHA

Conclusion: Thus we have implemented a program to generate and verify CAPTCHA image.

Assignment No: 4

Problem Statement: Write a computer forensic application program for recovering permanent deleted files and deleted partitions.

Objectives: To write a computer forensic application program for recovering permanent deleted files and deleted partitions.

Theory:

Destroyed Evidence

In a criminal or cyber-criminal case, the attempts to destroy the evidence are very common. Such attempts can be more or less successful depending upon the following conditions:

- Action is taken to destroy the evidence.
- Time Available to destroy the evidence.
- Type of storage device like magnetic hard drive, flash memory card, or SSD

Deleted Files

Deleting files is one of the easiest, convenient, and foremost ways to destroy the evidence. Whether it is using the —"Delete" button or —"Shift+Delete" button. The principle of file recovery of deleted files is based on the fact that Windows does not wipe the contents of the file when it's being deleted.

Instead, a file system record storing the exact location of the deleted file on the disk is being marked as —"deleted" and the disk space previously occupied by the deleted file is then labeled as available – but not overwritten with zeroes or other data.

- The deleted file can be retrieved by analyzing the contents of the recycle bin as they are temporarily stored there before being erased.
- If the deleted files have no trace in the recycle bin like in case of the —Shift+Delete‖ command, then, in that case, you can use commercial recovery tools to recover the deleted evidence. One such example commercial tool is Disk Internals Partition Recovery.
- Looking for characteristic signatures of known file types by analyzing the file system and/or scanning the entire hard drive, one can successfully recover :
    - Files that were deleted by the user.
    - Temporary copies of Office documents (including old versions and revisions of such documents).
    - Temporary files saved by many applications.
    - Renamed files.

Formatted Hard Drives:

Recovery of the data from the formatted hard drive depends upon a lot of parameters. Information from the formatted hard drive may be recoverable either using data carving technology or by using commercial data recovery tools.

There are two possible ways to format a hard drive: **Full Format and Quick Format**.

Full Format – As the name suggests, this initializes the disk by creating the new file system on the partition being formatted and also checks the disk for the bad sectors. Prior to Windows Vista, a full format operation did not zero the disk being formatted. Instead, Windows would simply scan the disk surface sector after sector. Unreliable sectors would be marked as ―bad‖. But in case of Vista and Windows 7, a full format operation will actually:

- Wipe the disk clean.
- Writing zeroes onto the disk.
- Reading the sectors back to ensure reliability.

Quick Format – This is never destructive except for the case of SSD. Disk format simply initializes the disk by creating the new file system on the partition being formatted. Information from disks cleared using a quick format method can be recovered by using one of the data recovery tools thatsupport data carving.

SSD Drives

SSD means Solid-State Drives represent a new storage technology.

- They operate much faster than traditional drives.

- They employ a completely different way of storing information internally, which makes it much easier to destroy information and much more difficult to recover it.

The culprit in SSD is **TRIM Command**. According to a survey, TRIM enables SSD completely wiped all the deleted information in less than 3 minutes. This means that the TRIM command effectively zeros all the information as soon as it is marked as deleted by the operating system. Moreover, TRIM command effects can‘t be prevented even by using Write-Blocking devices.

Traditional Methods are not useful when we try to recover deleted data from the SSD or even any information from the SSD formatted with either Full format or Quick format. This means the traditional methods can be used for data recovery in SSD only when the TRIM command is not issued or at least one of the components does not support TRIM. The components include:

- **Version of Operating System:** Windows Vista and Windows 7 support TRIM Command, on the other hand, Windows XP and earlier versions typically don‘t support TRIM Command.

- **Communication Interface:** SATA and eSATA support TRIM, while external enclosures connected via USB, LAN or FireWire don‘t.

- **File System:** Windows supports TRIM on NTFS volumes but not on FAT-formatted disks. Linux, on the other hand, supports TRIM on all types of volumes including those formatted with FAT.

Conclusion: Thus we have studied to write a computer forensic application program for recovering permanent deleted files and deleted partitions.

Assignment No: 05

Problem Statement: Write a program for log capturing and event correlation.

Objectives: To write a program for log capturing and event correlation.

Theory: What is log capturing? The logs generated by any system are stored in files which are called as log files. We can retrieve this information using various system commands which is called "Log Capturing".

Linux systems have a very flexible and powerful logging system, which enables you to record almost anything you can imagine and then manipulate the logs to retrieve the information you require. Linux uses a set of configuration files, directories, programs, commands and daemons to create, store and recycle these log messages.

Need of log capturing:

Linux system administrators often need to look at log files. Knowing where the system keeps its log files and how to make use of related commands can therefore help save valuable time during troubleshooting. It helps to monitor the system performance.

From security point of view the events logged in the log file are mainly:

1. Authentication attempts – both successful and failed.

2. All bad requests – which includes attempts for SQL injections and various hacking efforts. Helps to monitor and check resource allocation and usage

Log capturing in Linux:

At the heart of the logging mechanism is the rsyslog daemon. This service is responsible for listening to log messages from different parts of a Linux system and routing the message to an appropriate log file in the /var/log directory. The rsyslog daemon gets its configuration information from the rsyslog.conf file. The rsyslog.conf file is found in the /etc directory. This instruction comes from a series of two-part lines within the file. The two part instruction is made up of a selector and an action. The two parts are separated by white space.

1) auth or authpriv: Messages coming from authorization and security related events.

2) kern: Any message coming from the Linux kernel.

3) mail: Messages generated by the mail subsystem.

4) cron: Cron daemon related messages.

5) daemon: Messages coming from daemons.

6) news: Messages coming from network news subsystem.

7) lpr: Printing related log messages.

8) user: Log messages coming from user programs.

9) local0 to local7: Reserved for local useLinux

Commands Used:

top - top provides an ongoing look at processor activity in real time. It displays a listing of the most CPU-intensive tasks on the system, and can provide an interactive interface for manipulating processes.

who – shows who is logged in.

last – shows listing of last logged in users, last searches back through the file /var/log/wtmp anddisplays a list of all users logged in since that file was created.

lastlog - reports the most recent login of all users or of a given user, lastlog formats and prints the contents of the last login log /var/log/lastlog file.

last reboot - to find out when was the system last rebooted.

strace - In the simplest case strace runs the specified command until it exits. It intercepts and records the system calls which are called by a process and the signals which are received by a process. The name of each system call, its arguments and its return value are printed on standard error or to the file specified with the -o option.

cat - concatenate files and print on the standard output.

system() - The C library function int system(const char *command) passes the command name or program name specified by command to the host environment to be executed by the command processor and returns after the command has been completed

Conclusion: Thus we have studied log capturing and event correlation.

Assignment No: 7

Problem Statement : Study of Honeypot.

Objectives: To study of Honeypot**.**

Theory:

Honeypots:
A honeypot is an "an information system resource whose value lies in unauthorized or illicit use of that resources"

"A server that is configured to detect an intruder by mirroring a real production system. It appears as an ordinary server doing work, but all the data and transactions are phony. Located either in or outside the firewall, the honeypot is used to learn about an intruder's techniques as well as determine vulnerabilities in the real system"

Honeypot History:
The first publically available honeypot was Fred Cohen's Deception ToolKit in 1998 which was "intended to make it appear to attackers as if the system running DTK [had] a large number of widely known vulnerabilities" More honeypots became both publically and commercially available throughout the late nineties. As worms began to proliferate beginning in 2000, honeypots proved imperative in capturing and analyzing worms. In 2004, virtual honeypots were introduced which allow multiple honeypots to run on a single server.

Types of Honeypots:
There are two broad categories of honeypots available today, high-interaction and low-interaction. These categories are defined based on the services, or interaction level, provided by the honeypot to potential hackers. **High-interaction honeypots** let the hacker interact with the system as they would any regular operating system, with the goal of capturing the maximum amount of information on the attacker's techniques. Any command or application an end-user would expect to be installed is available and generally, there is little to no restriction placed on what the hacker can do once he/she comprises the system. On the contrary, **low-interaction honeypots** present the hacker emulated services with a limited subset of the functionality they would expect from a server, with the intent of detecting sources of unauthorized activity.

General Honeypot Advantages and Disadvantages:
Honeypots provide several advantages over other security solutions, including network intrusion detection systems:

- Fewer false positives since no legitimate traffic uses honeypot
- Collect smaller, higher-value, datasets since they only log illegitimate activity

- Work in encrypted environments
- Do not require known attack signatures, unlike IDS

Honeypots are not perfect, though:

- Can be used by attacker to attack other systems
- Only monitor interactions made directly with the honeypot - the honeypot cannot detect attacks against other systems
- Can potentially be detected by the attacker

Tradition security solutions, such as intrusion detection systems, may not be enough in light of more complicated attacks. Honeypots provide a mechanism for detecting novel attack vectors, even in encrypted environments. Advances such as virtualization have made honeypots even more effective. Honeypots have drawbacks, though, so it is important to understand how honeypots operate in order to maximize their effectiveness.

Honeynets and Honeyfarms:

Honeynets and honeyfarms are the names given to groups of honeypots. Honeyfarms tend to be more centralized. Grouping honeypots provide many synergies that help to mitigate many of the deficiencies of traditional honeypots. For instance, honeypots often restrict outbound traffic in order to avoid attacking non-honeypot nodes. However, this restriction allows honeypots to be identified by an attacker. He et al. use honeyfarms as redirection points for outbound traffic from each individual honeypot. These redirection nodes also behave like real victims Figure 1 shows the redirection of outbound traffic from a honeypot to another node in the honeyfarm.

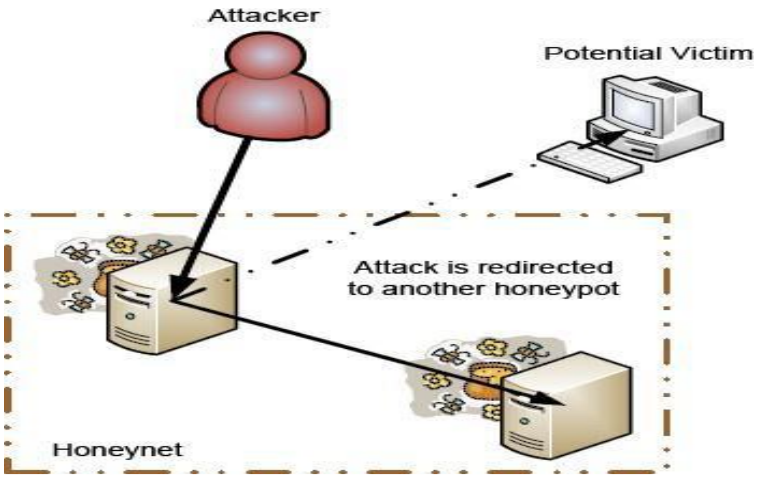

Figure 1. Redirecting an outbound attack in a honeypot

Shadow Honeypots

Shadow honeypots are combination of honeypots and anomaly detection systems (ADS), which are another alternative to rule-based intrusion detection systems.

Shadow honeypots first segment anomalous traffic from regular traffic. The anomalous traffic is sent to a shadow honeypot which is an instance of a legitimate service as shown in Figure 2. If an attack is detected by the shadow honeypot, any changes in state in the honeypot are discarded. If not, the transaction and changes are correctly handled. While shadow honeypots require more overhead, they are advantageous in that they can detect attacks contingent upon the state of the service.

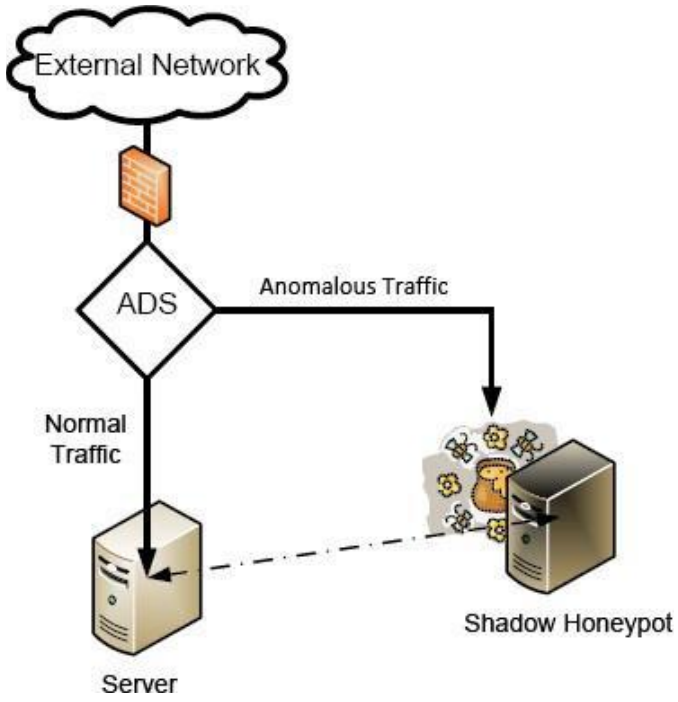

Figure 2. Segmenting traffic in a shadow honeypot system

Distributed Honeypots

One disadvantage of honeypots is that must take up a large portion of the address space in order to be efficient and useful (since attackers and malware must target the honeypots). Yang et al. provide a distributed framework for grid computing in which legitimate hosts redirect suspicious users to a single honeypot . An alternative is used by Honey@home in which each client is responsible for a single unused IP address. The client traffic is redirected anonymously through the Tor network to a collection of central honeypots .

Honeyfarms, honeynets, and distributed honeypots all address the need to monitor a large set of network addresses in order for a honeypot to be effective. As discussed in Section 2.1,

grouping honeypots can also add functionality to honeypots by allowing for operations such as simulated outbound traffic. Honeynets, shadow honeypots, and distributed honeynets are just a few of the advances occurring in the field of honeypots. We encourage you to explore journals and online to read about the latest advances.

Existing Honeypot Products

In this section, we provide a very brief survey of the Honeyd, HoneyBOT, and Specter honeypots. For each, we describe what differentiates the solution and provide reference information. We then offer advice for selecting amongst the solutions.

Honeyd

Honeyd is a honeypot for linux/unix developed by security researcher Niels Provos. Honeyd was ground- breaking in that it could create multiple virtual hosts on the network (as opposed to just using a single physical host). The honeypot can emulate various operating systems (which differ in how they respond to certain messages) and services. Since Honeyd emulates operating systems at the TCP/IP stack level, it can fool even sophistic network analysis tools such as nmap. Upon attack, Honeyd can passively attempt to identify the remote host. The Honeyd project is located at http://www.honeyd.org/

HoneyBOT

HoneyBOT is a Windows medium-interaction honeypot by Atomic Software Solutions (http://www.atomicsoftwaresolutions.com/honeybot.php). It originally began as an attempt to detect by the Code Red and Nimda worms in 2001 and has been released for free public use since 2005. HoneyBOT allows attackers to upload files to a quarantined area in order to detect trojans and rootkits. HoneyBOT's user interface is shown in Figure 3.

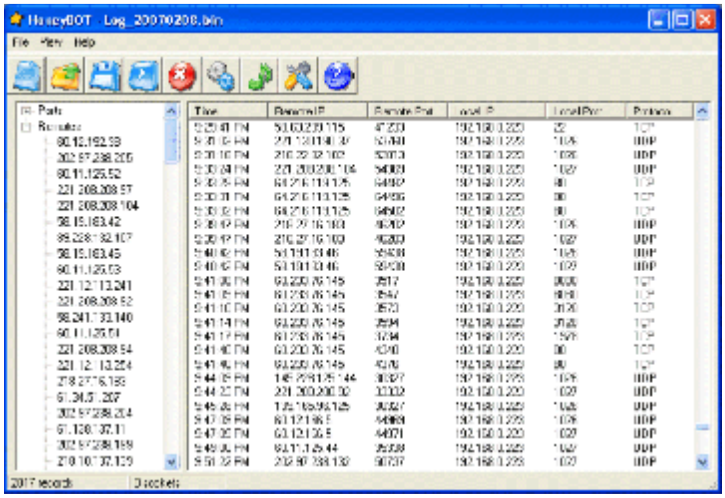

Figure 3. The main HoneyBOT user interface. Taken from http://www.atomicsoftwaresolutions.com/screenshot.php.

Specter

Specter's authors describes Specter as a "honeypot-based intrusion detection system". However, the product is primarily a honeypot designed to lure attackers away from production systems and collect evidence against the attackers. Specter has a few interesting features not found in other solutions:

- Specter makes decoy data available for attackers to access and download. These data files leave marks on the attacker's computer as evidence

- Specter can emulate machines in different states: a badly configured system, a secured system, a failing system (with hardware or software failures), or an unpredictable system.

- Specter actively attempts to collect information about each attacker

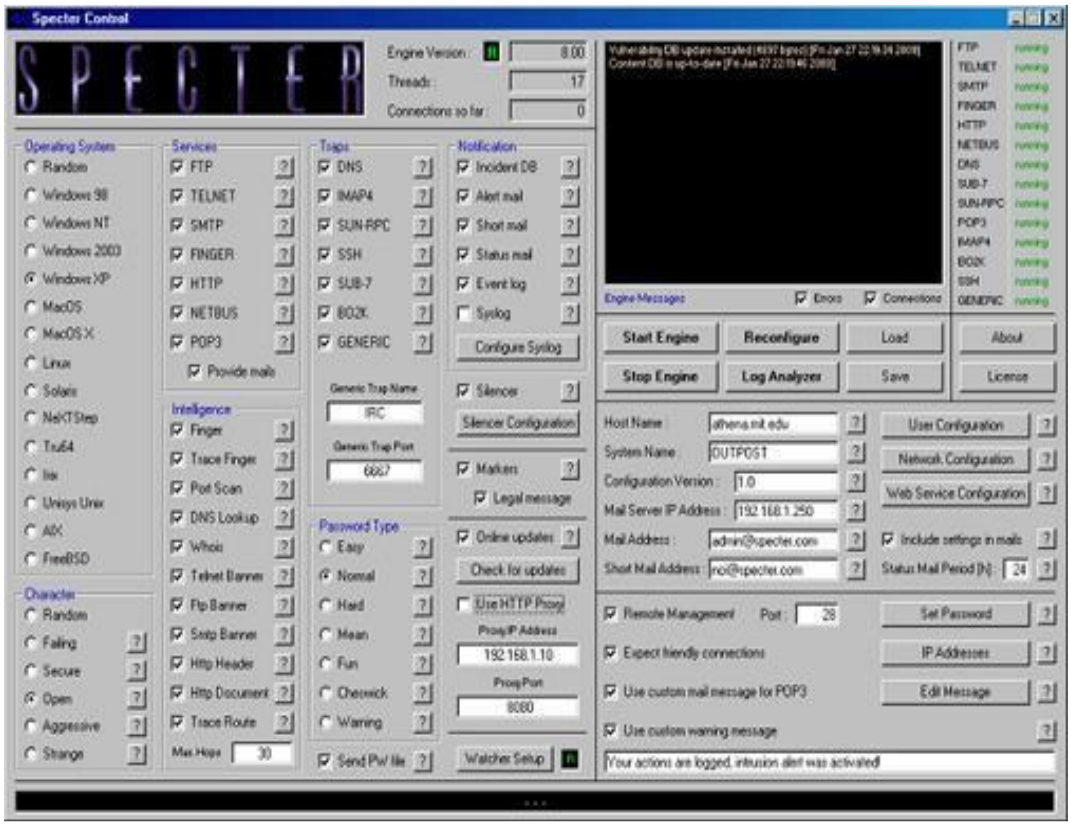

Figure 4. The Specter control center. Taken from http://www.specter.ch/details50.htm.

Conclusion: Hence we have successfully studied honeypot.

Group 2:

Mini-project:

Perform the following steps:

- Go to http://www.usdoj.gov/criminal/cybercrime/cyberstalking.htm.
- Read the 1999 report on cyber stalking.