File Hound: A Forensics Tool for First Responders

By

Wm. Blair Gillam and Marc Rogers

From the proceedings of

The Digital Forensic Research Conference

DFRWS 2005 USA

New Orleans, LA (Aug 17th - 19th)

File Hound: A Forensics Tool for First Responders

Wm. Blair Gillam
Purdue University
wgillam@purdue.edu

Marc Rogers
Purdue University
mkr@cerias.purdue.edu

ABSTRACT

Since the National Institute of Justice (NIJ) released their Electronic Crime Needs Assessment for State and Local Law Enforcement study results in 2001, several critical strides have been made in improving the tools and training that state and local law enforcement organizations have access to. One area that has not received much attention is the computer crime first responder. This paper focuses on the development and current results from File Hound, a "field analysis" software program for law enforcement first responders that is currently used by over 14 law enforcement agencies around the State of Indiana. It has been successfully used in several cases ranging from child pornography to fraud.

Keywords
Computer Forensics, Digital Evidence, Tool Development

1. INTRODUCTION

The Electronic Crime Needs Assessment for State and Local Law Enforcement Research Report [1] released in 2001 highlighted the needs of state and local law enforcement agencies for combating computer crimes. Ten critical needs were defined including uniform training and certification courses, cooperation with the high-tech industry, special research and publications, and investigative and forensic tools. Seventy-five percent of agencies involved in the study reported that they did "not possess the necessary equipment or tools to effectively detect and identify computer or electronic intrusion crimes" [1]. The study also found that many investigative and forensic tools were out of budget for most local and state agencies. However, when these tools were available, they were frequently out-of-date and incapable of being used in a forensic investigation. A real life example of this occurred when a police agency in Indiana could not purchase updates for a commercial product because the vendor was no longer registered with the state as a supplier for state agencies. The police agency was forced to use a very outdated version which resulted in a potential process/procedural "gap" that a defense attorney could have exploited to question the integrity and admissibility of any derived evidence.

While major strides have been made over the past four years to eliminate these ten critical needs, similar needs and problems still exist today. There are roughly 150 different tools used by law enforcement agencies to assist in computer crime investigations [2]. Of those 150 tools, many fall into two different branches—commercial software and open source software.

Commercial computer forensics packages are attractive for managing cases and performing forensic investigations. However, for some agencies, these commercial solutions are financially unattractive. These agencies have a choice between purchasing the tools and providing resource capabilities that are easier to justify (e.g., traffic enforcement, drug enforcement).

There is also a level of complexity that must be dealt with. Many of the commercial tools require specific training on the functions and methods of usage. In addition to selling the software, most vendors offer classes on their products, but here again at an additional cost to the agency.

On the other end of the spectrum are the Open Source applications. Open Source software is very affordable (i.e. free), but for a first responder, there can be a rather large associated learning curve with open source software. Much of this stems from users who are familiar with a Windows GUI environment and have little or no experience with operating systems other than Windows.

These tools are excellent for case management and investigations in a laboratory but investigations do not always occur in the lab. Time-sensitive investigations can occur out in the field. This has led to a new classification of investigators--the first responders. These officers are the first on the scene and have basic training dealing with searching and handling digital evidence. File Hound was developed to assist first responders in conducting a quick field analysis to satisfy 4th Amendment (protection against unreasonable search and seizure) and issued warrant requirements. It should be noted that File Hound was not designed to compete with any of the more function-rich open source or commercial software packages.

2. FILE HOUND

File Hound (FH) is a "field analysis" software package for law enforcement first responders developed at Purdue University during the summer of 2004. It is currently free to law enforcement agencies and limited support is provided. Due to time limitations and the operating system of the target audience, all prototype and release versions were created using the Visual Basic .NET programming language. Any computer capable of running Microsoft Windows 2000 or XP with the Microsoft .NET Framework is capable of running File Hound. It was also assumed that File Hound would be used with some type of hardware write block device.

The File Hound project centered on four basic functions[1]:

1. *Search for images*. The software had to be able to search a hard drive for image files. Since a filename search may not be thorough enough for a forensics investigation, the search must focus on file headers to determine a file's true identity.

2. *Identify relevant images*. Since several hundred or thousand files may be found during a search, the software had to present an interface for an examiner to browse through the images found and select those relevant to an investigation. This interface should be simple but yield the results in an intuitive form.

3. *Generate a report of the results*. The software had to generate a report of the results. At a bare minimum, the report must include the full logical path of the file.

4. *Require minimal user training*. A user should be able to fully utilize the software with minimal user training. A powerful but intuitive user interface was determined to be the best means to accomplish this goal.

After the goals/functions for File Hound were laid out, the application was divided into three separate modules based on function—searching, identification, and reporting.

2.1 Searching

File Hound searches for files using a recursive directory search. A pattern-based search can be performed in File Hound. Using asterisks as wildcards, simple or complex filters can be created. However, in many cases, the filenames are changed to innocent looking names or file

[1] A possible fifth function involves making the software capable with all hardware write blockers. Guidance Software's FastBloc is the only hardware write blocker that File Hound has been tested with.

extensions are changed to hide the images. To overcome this obstacle, File Hound utilizes an image header-based search.

A very simple way to check the file format of a possible image file is to use the Bitmap class (in .NET) to obtain the raw format of the file. See figure 1 for an example of using the Bitmap class to obtain the raw format of a file in C#.

Figure 1. TestImage class in C#.

```
public class TestImage
{
    Bitmap m_testbitmap;
    public string Format
    {
        get
        {
            ImageFormat bmpFormat = m_testbitmap.RawFormat;
            if (bmpFormat.Equals(ImageFormat.Jpeg)) strFormat = "JPEG";
            return strFormat;
        }
    }
}
```

The RawFormat method in the ImageFormat class in .NET is limited to 10 possible image formats. Table 1 lists these formats.

Table 1. Image formats from the ImageFormat class.

BMP	JPEG
EMF	Memory BMP
EXIF	PNG
GIF	TIFF
ICON	WMF

The first version of File Hound relied solely on the ImageFormat class to determine whether a file was an image or not. In subsequent versions, a similar means of checking for images is used; however, it has been enhanced to look for other image types like .art files.

The current version utilizes an algorithm that tests the raw format of a file during the search process. If a file is determined to be an image, the algorithm also retrieves the file size, MAC times, and an MD-5 (or SHA-512) hash of the file. These attributes are stored in an embeddable database which File Hound uses to manage all the information it finds. (If EXIF [7] information is present, it is also stored in the database.) This embeddable database is comprised of a compact, single-file engine (approximately 500 kB in size) that uses a high-speed SQL-92 compliant query processor. Information contained in this database can be saved however, when File Hound is closed or another search occurs, the database is reverted back to a "clean" state in preparation for new search results.

When the initial search has completed, File Hound displays a list of all the filenames found including their full logical path, MAC times, and hash. Investigators can use this information to add "interesting" files to their report.

2.2 Identifying

Since identifying which files are relevant to an investigation can be a daunting task when scrolling through a list of filenames, File Hound has an identification screen. The original version of File Hound only displayed one thumbnail at a time. Based on feedback from our users and taking a cue from the Windows operating system, File Hound now uses a list view to display multiple thumbnails of the images found during a search. This provides a mechanism that allows investigators to quickly and efficiently examine hundreds and even thousands of images.

2.3 Reporting

Finding images is only a piece of the overall solution. In many cases, a hard copy of an image would be beneficial for an investigation. File Hound offers the ability to generate reports based on images found. A standard report includes the case number, name of the examiner, and a thumbnail of each image along with the full logical path, MAC times, and MD-5 or SHA-512 hash. These reports can either be printed or saved for later viewing. Another reporting feature is the ability to print out a full page image.

An optional case log report is currently in development. This simple document will provide

play-by-play documentation of all the steps an investigator has taken inside the File Hound program.

2.4 Training

Requiring minimal user training was the final goal for File Hound. Care was taken to craft an interface that was easy to understand and intuitive for first-time users. The first File Hound training session consisted of a 15 minute step-by-step presentation on File Hound. All participants were taught how to use File Hound in conjunction with a hardware write blocker to preserve the integrity of potential evidence. The attendees were then given a mock case to work on. Due to File Hound's usability, all the attendees appeared to pick up on the program quickly.

2.5 Testing

Testing File Hound proved to be a unique challenge. There are currently no set standards for "forensic software". An effort was made to create real life tests for File Hound. An examination laptop was setup complete with a hardware write blocker and File Hound. Several lab workstations had images of money and the Purdue campus "planted" in random locations on the hard drives. The hard drives were each retrieved and searched using File Hound. In these tests, File Hound found every image that was planted.

Several agencies requested the ability to read the contents of raw image files (obtained by 'dd') with File Hound. File Hound cannot natively read these images. It can read images mounted with Mount Image Pro with somewhat successful results. Brian Carrier's Digital Forensics Tool Testing Image #8 (JPEG Search Test) [5], a raw partition image of an NTFS partition, was mounted using GetData's Mount Image Pro. File Hound was then opened and the newly mounted drive was selected and successfully searched. This test proved that File Hound could successfully read and search a volume mounted from a raw image. In theory, if an image of a hard drive is mounted and Windows can read it, File Hound can search it. Several valuable observations were noted with the test. File Hound found and displayed the jpeg file with a jpeg extension and the jpeg file with a non-JPEG extension. More importantly, File Hound did not return false positives with several invalid files that contained the 0xffd8 value but were not images.

Not all the results were positive as File Hound was unable to find the images hidden in Word documents, zip archives, and alternate data streams. File Hound was also not able to find the deleted jpeg files. The JPEG Search Test was not an all-inclusive test pattern; however, it has provided direction for future File Hound development.

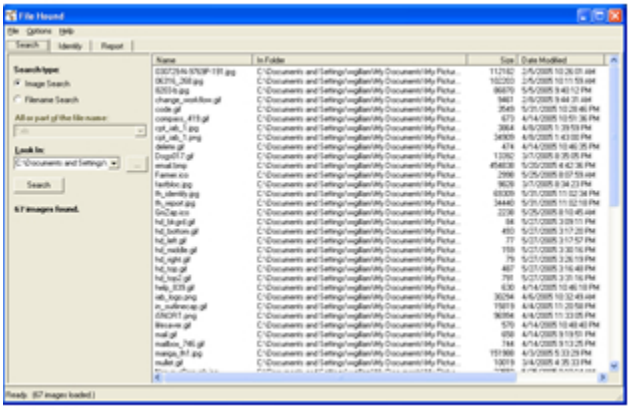
Another test for File Hound was to see how it would react to image bombs. Image bombs are image files that appear small in size but when viewed, expand into large images. Two different image bombs were used—one that expands to 500 MB and one that expands to 1 GB. Surprisingly, File Hound was able to handle both image bombs. The only issue that arose occurred when trying to view the larger 1 GB image bomb in thumbnail mode. File Hound knew it was an image but was unable to show a thumbnail preview.

3. EXAMPLE INVESTIGATION

This section provides a brief overview of an example investigation using File Hound. The investigation begins when the suspect's hard drive is connected and mounted to the investigator's laptop using a hardware write blocker. File Hound is started and the suspect's hard drive is selected from a drop down list. By clicking search without changing any options, an image search is initiated. Once the search has completed, the results are displayed in a tabular

format. Figure 2 shows the results from a completed search.

Figure 2. File Hound Search Results.



The total time needed for the initial search depends on the size of the files being searched. File Hound typically searches through a gigabyte of data in 15 minutes.

Next, image identification can occur using the identify tab. Figure 3 shows the identification window with thumbnails of all the images found during the search. The investigator can select any of the images for inclusion in the final report.

Figure 3. Thumbnail Identification Screen.

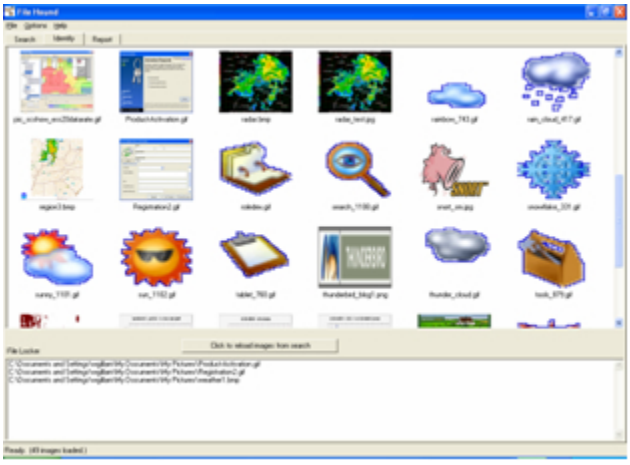


Figure 4 is the final report which displays the image thumbnail and the full logical path to the image.

Figure 4. Report Generated in File Hound.

4. CONCLUSION

File Hound was originally released to the law enforcement agencies attending the 2004 National White Collar Crime Center training conference held at Purdue University in August 2004. An overwhelming positive response was received. This hands-on training approach provided an excellent opportunity to receive immediate feedback on File Hound including any improvements and recommendations straight from the users who requested it.

In total, there were 13 distinct features requested. The features and improvements requested ranged from simple changes like canceling a search if the user has seen enough to incorporating the NIST hash database to look for known malicious tools and scripts. Several requests were made on improving the reporting function. In the original File Hound, the report was a text file containing the full logical path of the file. Several agencies requested the ability to print out a thumbnail of the image. They requested that detailed information about the image or file be included. This includes MAC times, MD-5, SHA-512, or CRC32 hash, and file ownership information.

4.1 Research Platform

Several agencies using File Hound have reported using it in Child Pornography cases but there is also at least one fraud case in which it was used. Aside from serving law enforcement needs, File Hound also serves as an academic research platform. Several development versions of File Hound exist. One is being used to conduct research on the Skin Color Analysis Algorithm (SCAA) [4]. The SCAA is an algorithm for detecting flesh tones in an image or a group of images. Instead of focusing on accuracy, the algorithm focuses on the speed of detecting the presence of skin tones among thousands of images. The algorithm groups images together based on temporal data associated with each image, like the last modified date. A sampling of pixels across all images is taken. If a set threshold is reached, the image or group of images is flagged as potential evidence. SCAA also contains a second algorithm that looks at each pixel in an image to determine if it is within a predetermined flesh tone range. Future research must be performed to test the accuracy of SCAA. At the present time, diversity in both ethnicity and lighting appear to be the two biggest limiting factors of SCAA.

4.2 Future of File Hound

The File Hound project has been very successful and has achieved all of its original objectives. The latest version is stable and has incorporated many of the recommendations from its user base. The future of File Hound looks bright as new capabilities are built into the software. These capabilities include making a platform independent version, timeline viewer, integrating the National Software Reference Library (NSRL), multithreading the application, and providing compatibility with Linux and UNIX file systems.

1. *Platform independent*. The next generation version will be platform independent. From a development standpoint, this means leaving the safety of the .NET Framework. Current possible solutions include porting File Hound into Java or Python. Ultimately, having the flexibility to use File Hound on any platform is a valuable characteristic.

2. *Timeline Viewer*. The timeline viewer is a graphical representation of when files were created or last accessed in relation to one another. This is useful in reconstructing a timeline of what a suspect was doing or looking at, thus verifying or disintegrating an alibi.

3. *Hash comparisons*. Integrating hash comparisons into File Hound is also on the horizon. Investigators will have the option to use the NSRL [3] released by the National Institute of Standards and Technology to compared known good and bad files. Investigators will also have the option to use their own hash set (i.e. Innocent Images) to compare against files.

4. *Compatibility with other file systems*. In its present state, File Hound only supports Fat32 and NTFS file systems. Future plans include providing support for HFS+, Ext2, and Ext3.

There are several issues that still need to be addressed in subsequent versions of File Hound. These include identifying images inside Alternate Data Streams, zipped files, and encrypted image files. File Hound currently does not identify images that have been deleted[2].

These new capabilities will give File Hound more flexibility; however, File Hound will remain a forensics tool for computer crime first responders and we will be responsive to their investigative needs.

One participant in the NIJ's 2001 study said "basic electronic crime training for all officers is

[2] Surprisingly, the ability to examine deleted images was not given a high priority by the first responders when surveyed regarding the initial basic functional requirements.

vital." Another stated that "technical assistance is needed for front-line officers and street cops to deal with electronic crimes that are occurring more frequently." File Hound is one software package that is not only helping front-line officers deal with basic computer crimes, it is also helping to strengthen the bond between the law enforcement community and academia.

5. REFERENCES

[1] Baker, R., Beaupre, D. S., Cassaday, W., Icove, D. J., Stambaugh, H., and Williams, W. P. *Electronic Crime Needs Assessment for State and Local Law Enforcement*. Research Report NCJ 186276, National Institute of Justice, Washington, DC, 2001.

[2] National Institute of Standards and Technology, "Project Overview," Computer Forensics Tool Testing Program, http://www.cftt.nist.gov/project_overview.htm (May 8, 2005).

[3] National Institute of Standards and Technology, National Software Reference Library, http://www.nsrl.nist.gov/index.html (May 8, 2005).

[4] Gillam, W., and Hoeschele, M., "First Responder Flesh Tone Detection Algorithms for Images," (2005).

[5] Carrier, B., "*JPEG Search Test #1,"* Digital Forensics Tool Testing Images, downloaded from http://dftt.sourceforge.net/test8/index.html (May 6, 2005)

[6] Kruse, W.G., II, and Heiser, J.G., Computer Forensics – Incident Response Essentials, Addison-Wesley (2002).

[7] Alvarez, P., "*Using Extended File Information (EXIF) File Headers in Digital Evidence Analysis.*" International Journal of Digital Evidence, Winter 2004, Vol. 2, Issue 3 (2004).

.