R·I·T

Rochester Institute of Technology

Golisano College of Computing and Information Sciences

Networking, Security, and Systems Administration Department

4050.581.1 – Computer Systems Forensics

iRecover – File Recovery Tool

Prepared By

ANDREW BELL, HASSAN ALSAFFAR, and LEONARDO RUBIO

Friday May 3, 2013

TABLE OF CONTENTS

❖ <u>Executive Summary:</u>

The purpose of this document is to outline the iRecover tool. This tool was developed to aid forensics investigators by combining other commonly used command-line forensics tools into a simple graphical user interface. Our team noticed that to accomplish a simple task such as carving files from images, an array of tools were required. iRecover seeks to improve this part of a forensics investigation automating as much as possible for the investigator and providing a simple to use graphical user interface. While developing this tool, we considered many options, many of which we had the time and resources to implement, such as customizable options when carving files from an image. Admittedly, our tool has much room for improvement, and we are open to providing future updates based on user feedback and recommendations.

❖ <u>Current Condition Overview:</u>

Our tool iRecover provides a GUI interface using the Perl/Tk programming library for forensics examiners to use when retrieving deleted files from an image/ file partition under forensic investigation. Investigators can use this tool to selectively retrieve only certain kinds of files (text, images, video, audio, documents, etc.) for a quick overview of information or for recovery using the "icat" Sleuth Kit tool to actually go into the image and select the files of interest to display.

- **Supported File Systems:**

As of this first version of iRecover, our script offers support for the FAT and NTFS file systems. With the FAT lineup of file systems, we recommend running our script against relatively recent versions of FAT (FAT16 or FAT32 for example). And yes, we understand how deliciously ironic it is that our tool is called iRecover and yet it does not allow processing HFS or HFS+ file systems. In the future, we hope to be able to provide support for a wide variety of other contemporary file systems in use (such as EXT2/3 and HFS).

- **SUPPORTED FILETYPES:**

These are based off of the output of the sorter script command that is includes as part of the Sleuth Kit set of forensics tools

Sorter groups file types together based on a common, shared category description (jpeg and gif files under "images" for instance). The list of sorter categories currently defined and supported in our scripts is based on off already pre-defined categories included in Sorter by default. They are:

- Images.
- Text.
- Data.
- Video.
- Audio.
- Compress.
- Crypto.
- Disk.
- Documents.
- Exec.
- System.
- Unknown.

For more specific details on what kinds of file types fall into each category see the sorter man page.

The GUI comes with a handy viewing feature for displaying key information on the selected file retrieved which includes filename, location, and certain metadata attributes such as file size and time of last modification. The last attribute is interesting when recovering deleted files as it helps give the investigator the time of deletion for the file being recovered. The user can also choose to retrieve only a subset of the discovered items for a given file type. Rather than retrieving all of them which could waste some space on disk especially if some of the uninteresting files are excessively large in terms of file size.

The standard procedure for operating our tool is to first determine if you would like to retrieve a specific category of files (only 1 category as of version 1.0) holding certain file types or all of the files on the image. After making a selection, the investigator should then select an image to process. At any point in this selection, reverting fields back to their defaults is easy – just press the "clear" button. Otherwise, clicking the "find" button will go into the image and utilize a wide variety of sleuth kit tools (mmls, sorter, fls, istat) to gather information and metadata on all the files that match the desired file type being searched for.

After finding the files, the tool will then output the results in the Listbox. This Listbox will display the findings according to the column headings that are defined - these include the file's name, its size, its file path (relative to the image being analyzed), a brief description of the file, and the time of last modification. It is possible to sort, resize, and move around these columns by clicking on the column headers and borders between each column. When you are ready to recover files, simply click on the files of interest (default is ALL files are selected) and then hit the "Recover" button. At any time, should you desire to delete the contents of the table, simply click the "Clear Entries" button or select another image to analyze. This will automatically remove the current listings from the listbox table.

Once the recovery process starts, iRecover will proceed to create a directory to store the recovered files in. Its location will be placed within the current directory the script is run from. iRecover will then carve out the files which the user has selected into the newly created directory using the "icat" command - the carved file will retain the same name that appears for it on the image. Currently for version 1.0, this recovery directory will follow the naming format “image-outputFiles” and every time the recovery button is clicked for a single image, more files will be placed within that directory.

In future updates if so desired, we hope to be able to provide the user the option to create or specify directories to save file carving results to (we feel this may help to better group and categorize files for later analysis in a similar vein of action as the Foremost tool)

This tool determines the identity of the files contained on the image through the use of the header bytes (or file signatures) contained at the beginning and end of certain kinds of files. Many of the Sleuth Kit tools we've employed in our script look for these "magic numbers" as they are also known as to properly identify the file (regardless of if the user tried to mask the true file type by modifying the file extension). Since our script is highly dependent on these tools, it is a requirement that our script be run on a UNIX based workstation where these tools are already pre-installed and included in the directories listed for the workstation's PATH variable. Some Linux distros and Live CDs come with all the Sleuth Kit tools pre-installed and loaded. Examples of these types of distros include Backtrack 5R3 and Kali Linux (among many others). We have tested and debugged our script thoroughly on the Backtrack 5R3 distro, so we highly recommend using our tool for investigations on this distro.

Our GUI is built on top of the Perl/Tk framework and programming library so it is necessary that this programming library is present on your Linux station. Normally, Perl is a given for any Linux machine nowadays but Tk may require a download and compilation from the CPAN library if it is not already present. Additionally, there are add-on widgets methods/modules that act as extensions to Tk and are not included in the regular Tk library by default. Necessary extensions that need to be included at run-time are the Tk::MListbox and Tk::ProgressBar modules in order to display the Listbox and Progress Bar displayed when the Recovery operation is performed. All of these can be obtained through use of CPAN and a strong Internet connection.

We have tested our script and recommend that it is executed from the command line. No options to the command line are necessary - all options and decisions can be set within the parameters and options of our GUI. Since Sleuth Kit tools depend on the file system metadata layer, our script is NOT suitable for performing memory forensics. Seek the usage of other tools (foremost, Volatility, etc.) for performing that area of forensic recovery.

❖ <u>Tool selection:</u>

Selecting the tool was an evolving process. The first step was to decide on what we thought needed the most improvement. We reminisced on previous labs and tools we had used to determine what could benefit most from improvements or functionality. Some early candidates were Sleuthkit, TCT, Strings, and Mactime. We eventually decided to move forward with the foremost tool, but the evolving didn't stop there.

During our brainstorm session, our initial plan was to create a front end for the foremost tool using a command line menu. Then we moved on to a full blown GUI for the same tool. From there, adding functionality seemed like an endless inevitability. Slowly, the foremost tool front end turned into a general file recovery front end with many tools being executed in the background as a chain of effects to produce useful results.

The project took shape, it became the file viewing, analyzing, and carving tool that we sought out. It works for FAT16, FAT32, NTFS, and EXT file systems and will display recoverable files (including deleted files). The columns in the GUI show the file name, file size, file path, last modified time, and description of files in an image (see Figure 1).

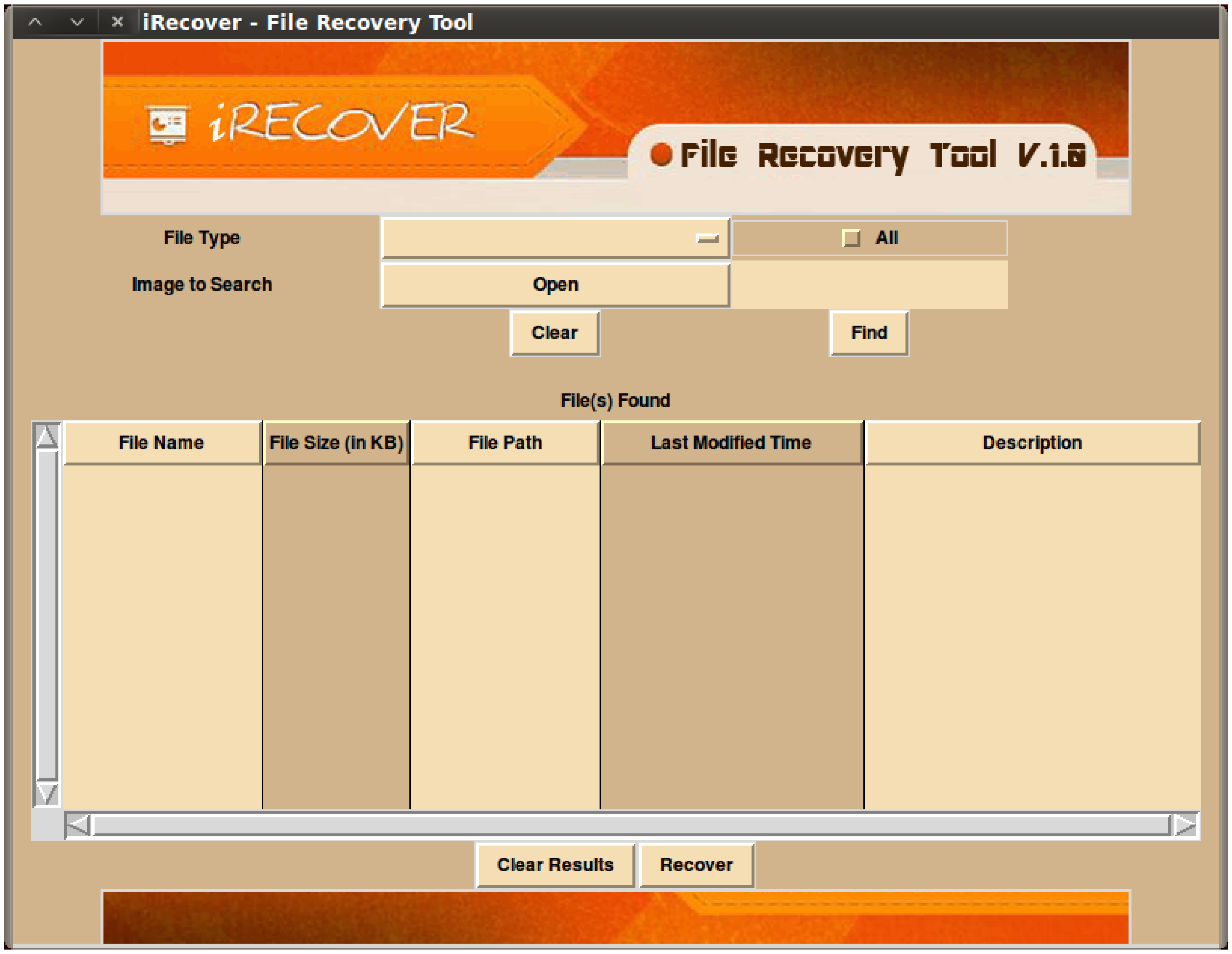

Figure 1. iRecover tool after coming into fruition.

❖ <u>**Tool development:**</u>

This tool was developed using PERL in conjunction with the Tk module. The development of this tool took place in 2 parts for division of labor and compartmentalization. Initially, there was a command line query driven script that performed system calls to the various sleuthkit tools. This script printed and organized the results but there was no GUI component to it. Then we had a GUI script that just laid out the initial template and functionality of iRecover.

We set out to combine these 2 separate scripts to inject the results derived from the first command line script with the nice columns and buttons interface of the second script. We began working towards this and for the most part it was a smooth confluence – when all testing and debugging finished, we managed to successfully port over the command line script into the GUI script forming just one single script for our final product. This was a desirable end result as having a GUI interface over command line makes using the tool more intuitive and lends greater ease of use. Most of the difficulties developing our tool came when testing out different file systems with the script and polishing it, as is further elaborated in the following section.

❖ <u>**Interesting Issues Encountered:**</u>

One of the issues we encountered in scripting and debugging our tool was in regards to how the various sleuth kit tools presented the output of their findings when dealing with various types of file systems. For a given sleuth kit tool, FAT file systems in particular had their own formatted output and NTFS file systems had different output when the same tool was run against it. This complicated matters as it made it very difficult for us to use one grep command that could apply to any of istat, fsstat, etc.'s output – we needed to try and find and workaround for this by using conditional statements checking for the file system type that was being analyzed and tailoring our parsing methods to deal with how the tool reports the output. It would certainly have helped greatly if the Sleuth Kit tools were consistent in how they obtained and presented results for the various types of file systems in use today. Though, we also acknowledge the fact that not all file systems are the same and each of the sleuth kit tools themselves would need to be modified in order to meet this goal.

❖ <u>**Conclusion:**</u>

We have outlined the full process and development life cycle of iRecover and its current condition. Its functionality and limits are not set in stone as this tool is still a work in progress. That being said, this tool is capable of becoming an integral part of a forensic toolkit. It provides a simple and quick way to carve files out of an image without having to use various tools separately – everything is utilized together in one package. The tool could be improved upon by adding more features and compatibility to expand upon its functions. The GUI can also use some sharpening and data delivery scalability. Overall, we're happy with the functionality and usability of our tool and believe it has potential.

We enjoyed developing this tool and feel we accomplished what we set out to do. There are numerous system calls going on in the background of this tool. As such, it makes file recovery a more streamlined and straightforward process for a forensic investigator.

We are open to continuing the development of this tool. With some feedback and criticism this tool could take shape as a viable and professional support to the sleuthkit suite. We look forward to receiving bug reports so that any problems can be patched and the tool's integrity can be improved.

References

- Lidie, Stephen and Walsh, Nancy. *Mastering Perl/TK.* Sebastopol, CA: O'Reilly & Associates, Inc, 2002. Print.
- Various Online Perl/Tk Discussion Boards and Help Forums