The Mobile Forensic Platform

By

Frank Adelstein

From the proceedings of

The Digital Forensic Research Conference

DFRWS 2002 USA

Syracuse, NY (Aug 6th - 9th)

DFRWS is dedicated to the sharing of knowledge and ideas about digital forensics research. Ever since it organized the first open workshop devoted to digital forensics in 2001, DFRWS continues to bring academics and practitioners together in an informal environment.

As a non-profit, volunteer organization, DFRWS sponsors technical working groups, annual conferences and challenges to help drive the direction of research and development.

http:/dfrws.org

MFP: The Mobile Forensic Platform

Abstract

Digital forensics experts perform investigations of machines for "triage" to see if there is a problem, as well as to gather evidence and run analyses. When the machines to be examined are geographically distributed, an investigator could benefit greatly if he could conduct the investigation, or even its initial stages, remotely.

The Mobile Forensic Platform (MFP) is a tool for performing remote network forensics. With it, investigators can gather evidence on a remote running system, maintain a copy of the original evidence (protected by a cryptographic hash), and run various analyses on the data to determine the next steps in the investigation (e.g., seize the machine, run tests, look elsewhere). The MFP maintains audit logs on all tasks it performs.

ATC-NY (formerly Odyssey Research Associates) has designed the framework for, and implemented a prototype of, the MFP. We have modeled the investigative process to define the tasks investigators perform. The framework defines the interface for each modular component in the process. The prototype serves as a proof-of-concept to demonstrate how the different components of the system will function together. We implemented the MFP on a laptop computer with a platform-independent web interface serving as the GUI. The investigator runs a web browser on his desktop computer, connecting remotely to the MFP's web server to perform the investigative tasks.

The preliminary version of the MFP supports a small set of analysis tools that illustrate the potential of the MFP. In particular, we created 3 sample logfile analysis tools. They detect if the data in a log file has been tampered, and also served to define how the analysis tools fit into the overall MFP framework. We describe each of the tools.

1 Introduction

After a crime, investigators must perform forensic investigations on any computers that might contain evidence. Investigators could be military personnel, law enforcement officials, company employees doing internal investigations, or external consultants. Because skilled investigators are few and machines may be physically far apart, there is a need for remote network-based forensics that includes gathering and analyzing evidence. It is typically impractical for an investigator to download the necessary information across the net. Firewalls, resource constraints (i.e., bandwidth), and security policies may prevent that—and even if they don't, an investigator may simply lack the resources to store it all. The goal of the MFP is to ease the burden on the investigator who needs to examine various, geographically dispersed sites.

2 The Need for Remote Forensics

A typical forensic investigator now performs the investigation by hand–reading mail and data files, checking for system activities through different log files, and verifying the consistency of the data through the time stamps associated with files on the file system. Protections such as firewalls often force the investigator to perform these tasks on-site.

The difficulties of performing a local analysis can limit the investigation. First, forensic software must be running on the local machine, and may have to be installed. Second, running such software locally risks damaging or contaminating data. Third, if the machine has been compromised, the investigation may produce suspect results—or worse, may alert the attacker. And finally, even if the forensic software does not alter the data (e.g., we boot an O/S from a floppy or CD-ROM), it still makes the machine unavailable, and may disrupt mission critical services.

Working through a remote login also poses problems. The network may not be configured to allow such remote connections (e.g., because of a firewall or because the network contains classified data). The network connection may not be fast enough to support the transfer of a large amount of data; and transmitting data across a public network (the Internet) risks divulging too much internal information.

What is needed is an approach that combines the advantages of both methods, accessing data locally on a secure platform with known tools, and keeping raw data within the network (to provide data security and to limit communications overhead).

3 How the MFP Works

The Mobile Forensic Platform (MFP) is a stand-alone machine, connected to the same local subnet as the machine to be investigated, allowing for high-speed transfer of data. The investigator, or an administrator, connects to the MFP through a secured connection (either through the Internet or a phone line), encrypting all data transferred between the machines.

We envision two ways in which the MFP can be used. A large organization with multiple geographically distributed sites could keep an MFP on-site, ready to be deployed. Before it was ever needed, it would be configured for that site (e.g., dynamic vs. static IP addresses, server names, services that are run, etc.). A system administrator could then deploy the MFP at the direction of the investigator, who need not present. This would involve essentially little more than plugging it into the network and a power outlet, and turning it on.

The second way the MFP could be used is by configuring it at the time it is deployed. The configuration should be straightforward and again could be done under the supervision of the investigator, via a phone call. In both situations, the investigator can begin gathering and analyzing evidence without having to go on-site.

The MFP tries to limit the amount of trust an organization must put into the investigator. In reality, he *will* have virtually unrestricted access to the machines that are under investigation. However, the MFP maintains its own audit log of every transaction performed, from every data file downloaded to every analysis or browsing of the data files.

4 The Investigative Process

We talked to a number of investigators and created a five-step description of the process they use, shown in Figure 1.

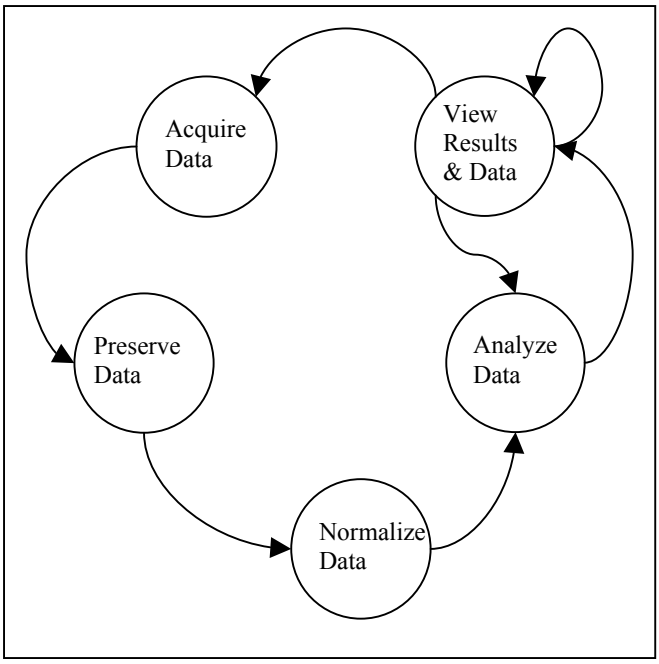

Figure 1 Investigative Process

The investigative process consists of:

1. acquiring evidence,
2. preserving evidence,
3. normalizing evidence,
4. running analyses, and
5. examining the results.

The only non-obvious steps are preserving and normalizing the evidence. Digital evidence is preserved by taking a cryptographically secure hash, such as MD5, of the raw data, such as MD5. An investigator can show that the data has not been changed at a later date by taking a hash of the data and comparing it to the original hash. Note that this implies that the original hash must be protected. It tends to be much easier to preserve the integrity of 100 bytes or so of data compared too tens or hundreds of megabytes of data.

Data is normalized by converting it to a common format. For example, all timestamps and machine names must be in the same format (local time vs. UTC vs. Epoch for timestamps, and IP numbers vs. FQDN for machine names).

The entire process repeats as needed. We built the MFP to follow this process.

5 MFP Support for the Investigative Process

The MFP has a modular design, to simplify coding and testing as well as to allow flexibility and extensibility. Conceptually, the MFP has six layers, as shown in Figure 2. The input comes from a lower layer, is processed, and then is passed up to the next higher

layer. The top layer represents the user interface, and the remaining five map into the steps of the investigative process shown in Figure 1.

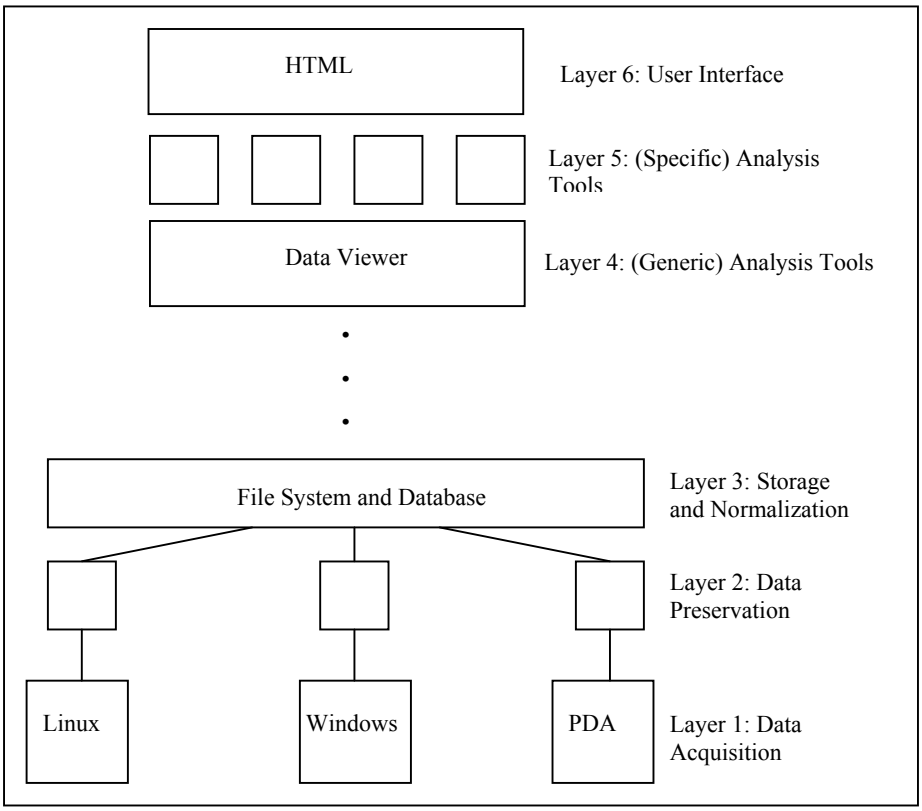

Figure 2 MFP Architecture

5.1 Acquiring

The lowest layer of the MFP handles the gathering data from the various sources. The details of communicating with the device under investigation and obtaining its data are confined to this layer. By limiting the "device dependent" code, the upper levels of the MFP can work independent of how the data was obtained, and new drivers can be added transparently, to support new platforms.

The initial plan is for the MFP to use FTP, NFS, SMB, and SCP as methods to transfer files from a running system. To do so, of course, the MFP must have the proper credentials (e.g., user name and password). Note that this level obtains the data, but does not care *what* the data represents; at this level, all files are treated equally.

5.2 Preserving

Layer 2 preserves the data via a cryptographically secure hash function. The MFP automatically preserves all evidence it acquires by taking an MD5 hash of the file. The hash can be viewed at any time. This process is the same for all data, since during this step all data is represented as a file on the MFP.

5.3 Normalizing

Layer 3 normalizes the data. In order to maximize the usefulness of our analysis tools, we want them to work on multiple kinds of files (e.g., from different operating systems or running in different time zones). So the data in these files must be converted into some standard format. The normalizing process depends heavily upon the nature of the data. Data files that contain timestamps must represent time in the same way. Similarly, files that contain machine names or IP addresses must use the same representation for all data. Normalization will change the data, which is why we preserve the original in the previous layer. It is always possible to start from the original data file and rerun all subsequent steps to verify the procedure (as might be required in a court of law).

5.4 Running

The analysis tools reside in layers 4 and 5. Layer 4 contains the generic tools that can be used on all types of data. Currently, the only element in this layer is a data viewing tool. It allows the investigator to view the raw data and search for particular strings. Layer 5 contains analysis tools that are specifically designed to work with a certain type of data, for example a web server log file or a syslog log file.

5.5 Analyzing

While Figure 1 distinguishes running analyses and viewing the results as separate steps, the MFP combines these two activities. So layers 4 and 5 are also responsible for viewing the data.

5.6 User Interface

Layer 6, the user interface, allows the investigator to perform all tasks in the process by mediating all interactions between the MFP and other layers of the MFP. (The user does not interact with layers 2 and 3, which operate automatically, layer 6 mediates all interaction between the investigator and the MFP. The MFP uses a web-based interface to allow the maximum compatibility with different systems.

6 Initial Set of Analysis Tools

For our "proof of concept" we created three sample analysis tools, representative examples that help define the interface between the different tools in the MFP (specifically, the form of the input data and the type of data produced as output). We concentrated on performing analyses on Unix style syslog log files. The tools check for log file consistency by looking for out of sequence entries, the size of gaps between the entries, and missing periodic entries. We describe the three tools below.

6.1 Monotonic

The first tool, the monotonicity check, verifies that entries in the logfile are in (ascending) chronological order. This is a simple, straightforward check. We developed this script primarily to define how to manage input and output flows and to define the format for the I/O data. It also served as a simple test of the functions that read and parse

log files, which are used in the other tools. This script served as a first step towards the development of the other two more complex scripts.

6.2 Max-Gap

The purpose of max-gap analysis is to quickly spot anomalous gaps in the log files. It creates a histogram showing the distribution of "gap sizes" in the log file. For example, a typical server might be generating log file entries between 5-10 seconds, and once per second during busy periods. If an intruder were to delete, say, a 5 minute section of the log max-gap analysis would make that deletion stand out.

The histogram divides the data into 10 "buckets" equal-sized on a logarithmic scale that is dynamically determined, depending on the ranges of data values. We use the following equation to compute the range of values that go into each bucket.

Given:

min = minimum gap size (dynamically determined),

max = maximum value of largest bin (dynamically determined by largest gap size),

$bins_{max}$ = maximum number of histogram bins (set to 10),

k = minimum value of smallest bin (set to 1.0), and

gap_i = the gap size for the i^{th} data value,

the index of the bin, bin#, in which the data value gap_i belongs (1 – 10) is computed as:

$$bin\# = \left\lfloor \frac{\log(gap_i) - \log(\min)}{\log(\max) - \log(k)} * bins_{\max} \right\rfloor$$

The tool then produces HTML-based output indicating, for each of the 10 buckets, the minimum and maximum values, and the number of items in each bucket. We may allow additional configuration parameters (such as choice of the number of buckets) in a later version of the MFP.

Figure 3 shows a screen shot from a sample max-gap analysis. The histogram shows a clear bimodal behavior in which part of the time the machine was active and the gaps were spaced together closely, within 10-20 seconds, and the rest of the time the machine was idle with large gaps (over one hour). If an intruder were active for a few minutes and then erased his activity, the anomalous gap would likely stand out. This example is meant to demonstrate how the tool works, not to assert that the results of this analysis are typical. The log file had very few entries and the machine was not a server. Nonetheless, the investigator can get a visual "fingerprint" of the structure of the log files and may be able to detect anomalies.

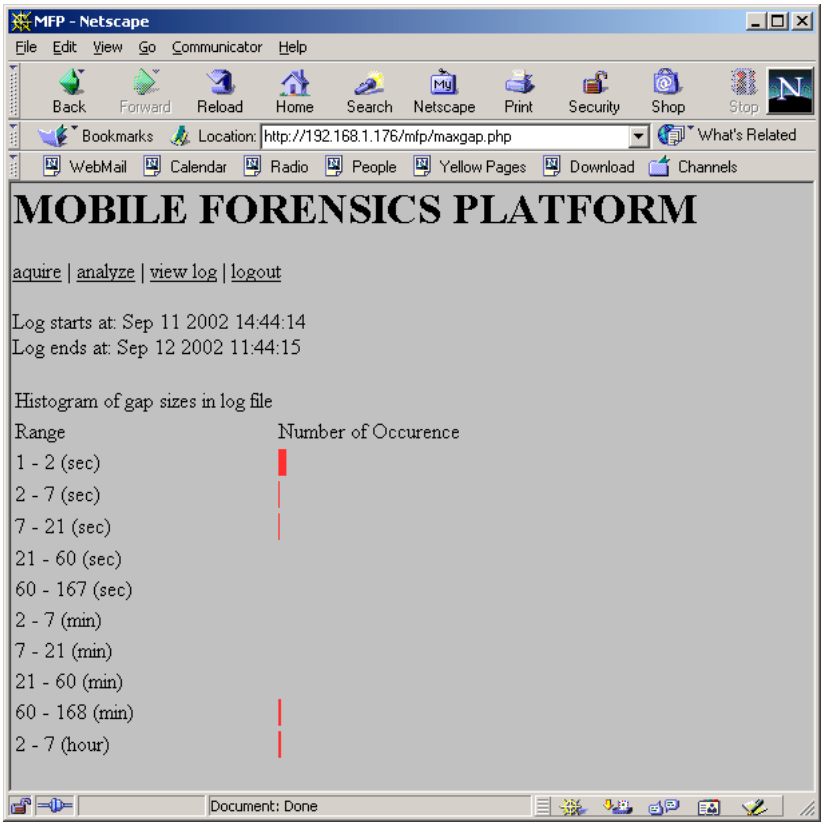

Figure 3 Sample Max Gap Analysis

6.3 Periodic

The periodic tool detects the absence of log entries that should appear at periodic intervals. The input to the tool must include a list of which periodic events to examine, the periods of those events and a string-pattern (or regular expression) which can be used to identify each event in the log file.

7 Related Work

Some automated tools are available that aid in the investigation, but they are limited. As far as we know, no one performs remote forensic analysis. The typical model used is that the machine is seized, the disk is imaged (i.e., duplicated), and the forensic analyses are performed on the duplicate disk.

Tools such as Encase[1] and FIAT[2] are designed to work on a seized disk (or, more commonly, a duplicate of such a disk). The Coroner's Toolkit (TCT) [3] must run on the system it is analyzing. The @stake Sleuth Kit (TASK) [4] is a tool kit built using tools from The Coroner's Toolkit to perform analysis of Windows and Unix systems. The Autopsy Forensic Browser [5] is an HTML-based graphical interface to TASK. Like TCT, it requires local disk access in order to perform its diagnostics.

8 Current status/Future Plans

Phase I was completed in January 2002. We demonstrated the feasibility of the concept, using discussions with various investigators as a "sanity check." We designed the plan for the full prototype. We created the structure and framework for the MFP and implemented selected components of it as a "proof of concept." These components include pieces of the user interface as well as analysis tools.

In the interim between Phase I and II, we have developed the initial demo MFP (presented at the Digital Forensics Research Workshop [6]).

In Phase II, which began on July 29, 2002, we will create a working prototype based on the work done in Phase I. We intend to work with investigators (as testers) to gather feedback and refine the MFP based on their comments. We will be adding new analysis tools and functionality (e.g., support for Windows event logs). Our goal is to work towards a Phase III, which ends with a dual-use, commercializable product.

9 Conclusion

Because expertise in forensic investigation is limited, it is vital to use investigators' time as efficiently as possible. Our new technology, the Mobile Forensic Platform, will help investigators work more efficiently by performing routine forensic tasks via a remote connection.

Our initial work has defined the basic structure of the MFP and the types of tasks it will perform. We created a flexible, extensible system, a forensic toolbox to which new tools can easily be added as the need arises. For the next phase of the project, we will create a fully functional prototype that can be used in the field and will incorporate feedback from testers.

References

[1] Distributed by Guidance Software, http://www.guidancesoftware.com/html/forensic_software.html

[2] Forensic Intrusion Analysis Tool (FIAT) was initially developed by MITRE and is now being developed by the DoD Computer Forensics Lab (DCFL).

[3] Created by Dan Farmer and Witse Venema, http://www.porcupine.org/forensics/tct.html

[4] The @stake Sleuth Kit (TASK), http://www.atstake.com/research/tools/task/.

[5] The Autopsy Forensic Browser, http://www.atstake.com/research/tools/autopsy/index.html.

[6] Digital Forensics Research Workshop, Syracuse, NY, August 2002, http://dfrws.org.