Digital Investigation 11 (2014) 179–186

Contents lists available at ScienceDirect

Digital Investigation

journal homepage: www.elsevier.com/locate/diin

Reverse engineering a CCTV system, a case study

Lee Tobin*, Ahmed Shosha, Pavel Gladyshev

DigitalFIRE Labs, CASL Institute, University College Dublin, Dublin, Ireland

ARTICLE INFO

Article history:
Received 22 February 2014
Received in revised form 21 July 2014
Accepted 22 July 2014
Available online 20 August 2014

Keywords:
CCTV
Reverse engineering
Proprietary file-systems
Disk image analysis
Investigation
Eavesdrop

ABSTRACT

Given a disk image of a CCTV system with a non-standard file system, how is the data interpreted? Work has been done in the past detailing the reverse engineering of proprietary file systems and on the process of recovering data from CCTV systems. However, if given a disk image without the CCTV system itself, or if under time constraints, the task becomes much more difficult. This paper explains a different approach to recovering the data and how to make sense of data on a CCTV disk. The method does not require extensive reverse engineering of the CCTV system, or even to have access to the CCTV system itself.

Introduction

Techniques for reverse engineering proprietary file systems are well documented (Ariffin et al., 2013; Poole et al., 2008), it is usually feasible provided the data is intact and not encrypted. So why revisit this topic? In some investigations, only the disk or disk image from the CCTV system is available. This limits the amount of reverse engineering that can be done as the investigator does not have access to the CCTV hardware. Sometimes the speed at which information can be extracted from a CCTV system is of the utmost importance; reverse engineering can be very time consuming. In these cases, speeding up the reverse engineering process without sacrificing the precision and depth of the analysis is desirable. This paper approaches the problem in a different way; instead of trying to explain data structures and data offsets from the raw data, the problem is tackled in a pragmatic way. To explain the approach, this paper will use an example scenario. This scenario is based on an analysis performed in a real case, though heavily redacted.

Related work

Wouter S. van Dongen performed a study of a Samsung CCTV system (Dongen, 2008) in order to obtain traces of video recordings. This in-depth forensic examination of the CCTV system also analyses logs and related video data. The method employed differs from our approach as the Samsung system contained a common and recognisable file-system (ext3). This facilitated the use of data carving tools to aid in the recovery of information in unallocated space. The approach undertaken in this paper is to understand the operation of the CCTV system in as little time as possible while maintaining forensic rigour.

Poole et al. performed a detailed analysis (Poole et al., 2008) of a CCTV system. This analysis required extensive testing and experimentation in order to reverse engineer the device. This type of analysis, while very robust and complete, takes time and requires access to the CCTV hardware. It may be more appropriate to perform this type of analysis if related CCTV software can not be found, a requirement for the approach detailed in this paper.

* Corresponding author.
E-mail address: lee.tobin@ucdconnect.ie (L. Tobin).

http://dx.doi.org/10.1016/j.diin.2014.07.002
1742-2876/

Problem

The example scenario is as follows:

- An investigator is given a 500 GB disk that was taken from a CCTV system, where the make and model is not specified.
- The investigator is asked if video data exists on the disk recorded before a certain date.
- The investigator has a limited amount of time in which to provide a detailed answer and analysis.

Method

The approach taken in this paper requires a host system on which to do the analysis, a way to monitor processes on the host system and a way to view data on a disk (a low-level disk viewer).

Software and hardware used in the paper

- Tableau Forensic Bridge – Model T35e
- Intel i7 based PC with 24 GB RAM.
- Microsoft Windows 7 Professional SP1 v6.1.7601
- XWays Forensics v15.8 SR-6
- Process Monitor v3.05

The "Eavesdrop" approach

Fig. 1 shows the flowchart explaining the approach taken in this paper. This method is explained from a Windows OS perspective, however it could just as easily be undertaken using any other operating system, in which suitable monitoring tools are available.

One of the most important steps in this approach is the identification of the make and model of the CCTV system, if they are not known. This knowledge is required in order to find proprietary software for the system.

Initial step

Firstly, the disk was connected via a forensic bridge to a virtual machine running Windows 7. The make and model of the CCTV was found after a quick examination of the first few bytes of the disk. Using a low level disk viewer (such as XWays Forensics), two easily recognisable strings "AVTECH" and "FSS16A" are found. This can be seen in Fig. 2.

A quick Internet search uncovers a piece of software related to AVTECH called "Disktools". Fig. 3 shows this software recognises the disk image. The disk contains several events and after double-clicking on the "2013/03/18 00:48:09" event a dialogue box selecting a channel from a possible 16 is shown. After a channel is selected, the chosen video for that date/time and channel is displayed. Now that some software is found that recognises the disk data, the reverse engineering process can begin. The basic idea is to perform a task in Disktools, such as playing a video, and while this task is being performed the disk I/O is monitored. This way, we get information as to how the data is structured on disk without having to analyse the application's executable directly.

Offsets to start of event list

Fig. 4 shows the output from Procmon after selecting the first event and first channel.

Process Monitor (Russinovich and Cogswell, 2013) (Procmon) is an advanced monitoring tool for Windows that shows real-time file system, Registry and process/thread activity. We can use procmon to glean some details about the disk layout.

Disktools is accessing the disk at byte offset 1EEDE00h (32431616d) in 200h (512d) byte chunks, which implies sector size. 1EEDE00h divided by 200h would be the sector offset F76Fh. Examining the first few bytes of the disk again, the bytes 6FF7h can be seen at byte offset 29h. (6FF7h little endian on disk). Scanning through the disk

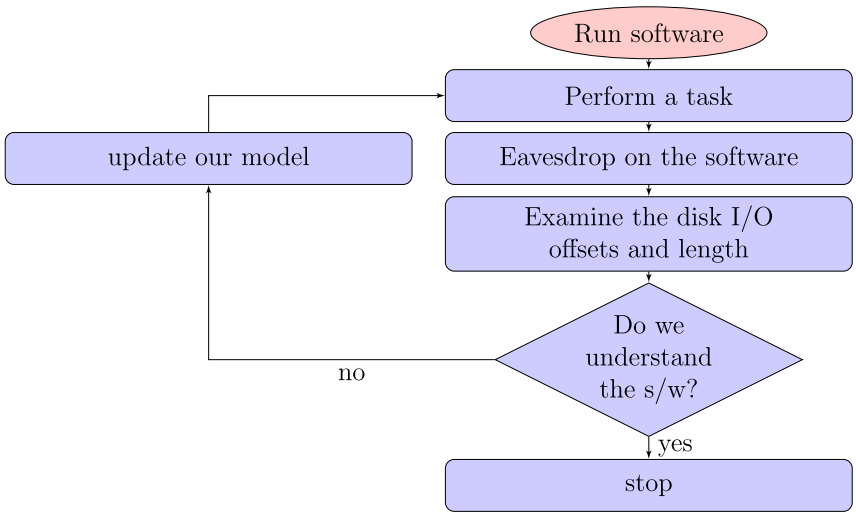

Fig. 1. Flow chart for the procmon method.