Uncovering Limitations of E01 Self-Verifying Files

Jan Krasniewicz
Birmingham City University, UK

Sharon A. Cox
Birmingham City University, UK

INTRODUCTION

Teaching good practice in computer forensics is important to understand the correct operation and limitations of computer forensic hardware and software. One task is to demonstrate the self-verification feature of evidence file formats such as the EnCase E01 file format that contains an image of acquired data. The E01 file contains the data plus extra data in the form of hash values and Cyclic Redundancy Check (CRC) values used by computer forensic software to check the data contained within the file has not been tampered with. Students are taught how to carry out this task and verify the file by making a change to the generated file and observing mismatches between hash values and Cyclic Redundancy Check (CRC) values generated when the data was copied and when the file is loaded into computer forensic software. Whilst creating teaching materials for students to carry out this task an anomaly was identified in one of the forensic file formats, the E01 format, commonly used by practitioners. The anomaly allows changes to be made to certain bytes within the file that are not detected by computer forensic software when verified by the associated hash and CRC values. This paper describes the anomaly in the file format, discussed the implications for relying on the self-verification feature of the E01 file format and concludes on methods to make any change to the file contents detectable.

Background

One of the first tasks before conducting a computer forensic analysis of data is to make a forensically sound copy of the data stored on, for example, a hard disk drive. This task forms the acquisition stage of an investigation. By "forensically sound" it is meant that the copying process does not alter the source data resulting in an exact copy of the data (Casey, 2007). This task involves making a bit-for-bit copy of the data and using a method that assists in determining the integrity of the resulting copy as part of the chain of custody.

It is important to be able to determine that the copy of data has not been changed before it is analysed. It is common practice and recommended by organisations such as the Association of Chief Police Officers (ACPO) and National Institute of Standards and Technology (NIST) to use a mathematical function to calculate a unique value for the data at the time of copying. Examples of mathematical functions used to check the integrity of data are Cyclic Redundancy Check (CRC) and cryptographic hash (Schneier, 1996). These functions are implemented in computer programs to compute a value from a computer file or entire contents of a storage device. The value is recorded so that whenever the digital evidence is analysed the value is recomputed and compared to the original value.

DOI: 10.4018/978-1-5225-2255-3.ch119

Copyright © 2018, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

Computer programs have been developed to automate the copying process and calculate the integrity values for the acquired data. These values are stored within the resulting copy of the data. Storing the integrity values within the file allows the copy to be *self-verifying* when analysed with computer forensic software. When the copy is used by a computer forensic software application, such as Guidance Software's EnCase and AccessData's FTK, the application recalculates the unique value and then compares it with the value stored in the file. The program displays a warning message when the original and calculated values are different as this difference indicates the file has changed, the change could be as a result of corruption or it could be more sinister due to a deliberate change by an individual.

This paper considers the integrity values stored in the copy of the data, commonly known as the image file or digital evidence container file (Common Digital Evidence Storage Format Working Group, 2006). The paper describes mathematical functions used to calculate the integrity values and how the property of the function allows data to be validated. The paper then describes how a practical exercise to demonstrate self-verification features to students identified an anomaly where it was possible to change a byte within the file without the self-verification detecting that the copy had been changed. The paper explains why additional integrity values should be calculated based on the entire data, copy and integrity values combined, to further enhance confidence the copy has not been altered after it has been made.

Hash Functions in Computer Forensics

Hash functions are one approach to solving the problem in computer systems of being able to perform a fast lookup of data in a data store such as RAM or disk drive (Knuth, 1998). The hash function takes as input a characteristic or characteristics of data and computes a value based on the characteristic. This value is used to locate the data within a data store, for example. When a program needs to find data within a set of data the hash value is calculated and the value used to examine a location in the data store for the presence of that data. As a result, the hash value offers almost direct access to the data.

In comparison algorithms such as linear search are slow as it involves examining each data location for the presence of a specific data value from the first to the last. The amount of time to find data in the data store using a linear search is proportional to the number of data items in the data store and as the number of items increases so does the time to search the data store. The advantage of the hash function is it computes a hash value in constant time irrespective of the location of the data in the data store. Consider the example of searching for an individual's personal details based on some identifying data within a data store comprising of 1,000,000 individuals. A linear approach would inspect every location until a match is found where the time take to find the data is proportional to the amount of data, more data results in a longer search, potentially. A hash function would be applied to the identifying data and a value calculated that is then used to access a location directly.

Ideally hash values calculated are unique for different data but for some applications this is not entirely a requirement (Knuth 1998). Computer forensics make use of hash functions but a specific type called cryptographic hash functions that are designed to produce unique values for data. This property of calculating a unique value allows the value to be considered as a digital signature or fingerprint of the data allowing for data to be found by its hash value or verified because its value is unique (Cowen, 2013).

Cryptographic hash functions are used for the validation of arbitrary messages (the data) between individuals (Schneier, 1996). A key property of cryptographic hash algorithms is that they permit the verification of arbitrary streams of data by producing a unique value for that message exactly. In the case of cryptographic hash functions like

9 more pages are available in the full version of this document, which may be purchased using the "Add to Cart" button on the publisher's webpage:
www.igi-global.com/chapter/uncovering-limitations-of-e01-self-verifying-files/183852

Related Content

A Roughset Based Ensemble Framework for Network Intrusion Detection System
Sireesha Rodda and Uma Shankar Erothi (2018). *International Journal of Rough Sets and Data Analysis (pp. 71-88).*
www.irma-international.org/article/a-roughset-based-ensemble-framework-for-network-intrusion-detection-system/206878

Infinite Petri Nets as Models of Grids
Dmitry A. Zaitsev, Ivan D. Zaitsev and Tatiana R. Shmeleva (2015). *Encyclopedia of Information Science and Technology, Third Edition (pp. 187-204).*
www.irma-international.org/chapter/infinite-petri-nets-as-models-of-grids/112328

A Fuzzy Knowledge Based Fault Tolerance Mechanism for Wireless Sensor Networks
Sasmita Acharya and C. R. Tripathy (2018). *International Journal of Rough Sets and Data Analysis (pp. 99-116).*
www.irma-international.org/article/a-fuzzy-knowledge-based-fault-tolerance-mechanism-for-wireless-sensor-networks/190893

A Comparative Study of Infomax, Extended Infomax and Multi-User Kurtosis Algorithms for Blind Source Separation
Monorama Swaim, Rutuparna Panda and Prithviraj Kabisatpathy (2019). *International Journal of Rough Sets and Data Analysis (pp. 1-17).*
www.irma-international.org/article/a-comparative-study-of-infomax-extended-infomax-and-multi-user-kurtosis-algorithms-for-blind-source-separation/219807

Impact of PDS Based kNN Classifiers on Kyoto Dataset
Kailasam Swathi and Bobba Basaveswara Rao (2019). *International Journal of Rough Sets and Data Analysis (pp. 61-72).*
www.irma-international.org/article/impact-of-pds-based-knn-classifiers-on-kyoto-dataset/233598