Selective Imaging of File System Data on Live Systems

By:
Fabian Faust, Aurélien Thierry, Tilo Müller and Felix Freiling

From the proceedings of
The Digital Forensic Research Conference
DFRWS EU 2021
March 29 - April 1, 2021

Forensic Science International: Digital Investigation 36 (2021) 301115

Contents lists available at ScienceDirect

Forensic Science International: Digital Investigation

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

Extended Abstract

Selective Imaging of File System Data on Live Systems

Fabian Faust [a,*], Aurélien Thierry [b], Tilo Müller [a], Felix Freiling [a]. [a] *Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), Germany;* [b] *QuoSec GmbH, Frankfurt/Main, Germany*

E-mail addresses: fabian.faust@fau.de (F. Faust), a.thierry@quosec.net (A. Thierry), tilo.mueller@cs.fau.de (T. Müller), felix.freiling@cs.fau.de (F. Freiling).

Keywords
Live forensics
Selective imaging
File system data
Forensic soundness

A B S T R A C T

In contrast to the common habit of taking full bitwise copies of storage devices before analysis, selective imaging promises to alleviate the problems created by the increasing capacity of storage devices. Imaging is selective if only selected data objects from an image that were explicitly chosen are included in the copied data. While selective imaging has been defined for post-mortem data acquisition, performing this process *live*, i.e., by using the system that contains the evidence also to execute the imaging software, is less well defined and understood. We present the design and implementation of a new live *Selective Imaging Tool* for Windows, called SIT, which is based on the DFIR ORC framework and uses AFF4 as a container format.

* Corresponding author.

E-mail addresses: fabian.faust@fau.de (F. Faust), a.thierry@quosec.net (A. Thierry), tilo.mueller@cs.fau.de (T. Müller), felix.freiling@cs.fau.de (F. Freiling).

1. Introduction

While the overall approach of forensic investigations of storage devices has changed little over the last decade, the amount of data that needs to be processed keeps increasing. Digital forensic investigators are therefore facing growing problems caused by technological advances in the size of storage devices.

In the literature, *selective imaging* (Stüttgen et al., 2013) has been advocated as the solution to these problems: The term refers to the process of only copying selected data objects, thus creating a *partial image* that needs considerably less time and space to be taken. Despite multiple proposed concepts for the classical post-mortem acquisition approach, however, the range of dedicated selective imaging tools remains scarce. As (Sack, 2017) reports from interviews with practitioners, while forensic investigators do regularly end up recognizing the need for a selective approach, they often use tools that neither are intended for forensic investigations nor fulfill even basic requirements of forensic soundness.

While selective imaging has been discussed in the context of (public) forensic investigations by law enforcement, it is also of great relevance to (private) forensics investigations performed by specialized companies within organizations, mainly with the goal of confirming the existence of unwanted software, unlawful actions by employees or external hacking attacks. The primary goal is often to gather as much relevant evidence about actions taken and caused by malicious third parties or software in the shortest amount of time possible. This is accompanied by the requirement that systems cannot be turned off for imaging, giving rise to *live forensics*, i.e., the capture of evidence using the same system to access its data. This is in stark contrast to the traditional post-mortem acquisition of data, for example, from a hard drive after shutting down the system (Heinson, 2015). Especially in larger companies, software is deployed that allows for a *triage* (Moser and Cohen, 2013), an initial classification of people, data, and objects into different priority categories for later manual analysis. However, generally, such software is not developed or intended for live selective imaging, and therefore not forensically sound for this approach.

So overall there is ample need for selective live acquisition of file system data within forensic investigations, but there is a definite lack of concepts and tools for performing this task.

In this extended abstract we present an overview of SIT, the *Selective Imaging Tool* that can perform selective imaging of file system data on live Windows systems. Through a carefully crafted design, SIT achieves a high degree of forensic soundness to safeguard the evidential value of the acquired partial image. It is based on modern investigative software components such as the DFIR ORC framework (ANSSI, 2020) and file formats such as AFF4 as its forensic container format. SIT is fully open-source and available on GitLab. We are not aware of any other open-source tool that allows the collection of evidence from live systems with similar degrees of reliability and integrity. Full details can be found in the corresponding technical report (Faust et al., 2020).

2. Selective imaging on live systems

Despite the fact that a live selective imaging approach offers a variety of significant benefits and can be the only option in certain cases, the

https://doi.org/10.1016/j.fsidi.2021.301115

challenges caused by the nature of working on live systems, that are often beyond the investigator's control prior to access, are significant. Some of these challenges make problems inherent to selective imaging in general more critical, while others originate solely from the live environment and the options for anti-forensic interferences with the investigation it offers. Before we develop general criteria for selective imaging on live systems, we briefly revisit the concept of forensic soundness.

2.1. Forensic soundness

Forensic soundness serves the goal of ensuring that the collected evidence is not altered in any way from the source, (McKemmish, 2008, p.8) defines forensic soundness as "the application of a transparent digital forensic process that preserves the original meaning of the data for production in a court of law."
According to (McKemmish, 2008) the main priorities of forensic soundness can be summarized as follows:

1. The acquisition and subsequent analysis of electronic data has been undertaken with all due regard to preserving the data in the state in which it was first discovered.
2. The forensic process does not in any way diminish the evidentiary value of the electronic data through technical, procedural or interpretive errors.

2.2. Forensic soundness on live systems

While it has been pointed out (Casey, 2011) that even the routine task of post-mortem data acquisition from a hard drive with a write-blocker alters the original state of the source, the copied bits of data are usually acquired in a reliable and repeatable manner since many influencing factors like the hardware and software used for copying are under full control of the analyst. The situation for live systems is totally different since the used hardware might fail during acquisition and the operating system of the target system might have been manipulated in diverse ways. The situation is therefore similar to the acquisition of volatile evidence such as RAM (Vömel and Freiling, 2011) or data about cryptocurrencies from a blockchain Frömis et al. (2019).
The following general requirements present a set of priorities for maintaining forensic soundness on live systems, further defined as a concrete set of rules in Faust et al. (2020).

- Minimize source corruption
- Ensure evidence data authenticity and integrity
- Provide extensive documentation
- Ensure digital reliability and security
- Ensure physical reliability and security

2.3. Documentation of the selection process

The selection process itself, whilst being an integral part of the selective imaging concept, should on an implementation level be treated separately from the actual imaging tool containing all the features required to proceed after the selection targets have been chosen. This is due to the fact that for the selection itself, any analysis tools that have minimal side-effects on the machine may be used. This includes both statically preselected lists of files to be acquired as well as a live analysis involving manual browsing and file selection. To satisfy the aspect of *verifiability* and the *chain of evidence*, the selection process must be sufficiently understandable, increasing the burden of documentation in case live browsing and manual selection are chosen.

3. SIT design and implementation

The main functionality of SIT is to allow the selective collection of forensic artifacts on file system level, alongside key metadata, validating the results to detect unexpected results and external interferences, integrating the results into an AFF4 forensic image, and then verifying the artifacts using hash codes, all while maintaining the new live forensic soundness rules. The project is available in its GitLab repository (Faust, 2020).

3.1. Design & architecture

The main goal of our implementation was to create a modular framework for selective imaging on live Windows systems that implements the rules established in. In order to achieve this, the software had to be portable, i.e. a binary that can be moved between different systems without the need for a prior installation and which is running with minimal external dependencies. Furthermore, in addition to the execution from an external flash drive, the usage of a custom temporary directory, and the extensive verification using more than one hash code for each artifact, more secondary measures were implemented.
A separate validation step after the artifact acquisition phase creates a redundancy to identify obvious interferences and attacks on the acquired data by malicious software or corruption caused by errors. As a suitable storage container, the `AFF4` format was chosen for its direct artifact–metadata association mechanic, intuitive metadata representation using RDF turtles, and the lightweight compression algorithm (Cohen et al., 2009). Providing extensive user feedback and logging was another priority, alongside sufficient error handling for basic security. Since external libraries need to be statically-linked in order to maintain portability and compatibility with different Windows versions, an aspect which increases the software's footprint and decreases its efficiency, one secondary goal was to directly implement simpler functions such as RDF serialization, thus avoiding the usage of a library. Lastly, a backup archive of all acquired artifacts serves as another redundancy in the case of data corruption. The effectiveness of these measures in achieving forensic soundness is evaluated in Faust et al. (2020).
As a foundation, SIT is using the DFIR ORC framework (ANSSI, 2020) to create a single portable preconfigured binary that can be run as a command-line tool. The intention is to execute it from an external flash drive in order to prevent overwriting files on the system storage devices and to have an option to extract the results.

3.2. SIT modules

SIT is made up of four logical modules. They are designed to run sequentially and while each module is built to work with the results of the previous one, they can also be repeated, executed independently, or disabled. If efficiency is a priority, disabling modules such as the Verification Module will improve performance at the cost of an on-site integrity check. In case of an unexpected shutdown by the live system or crashes during the imaging process, the intermediate results up to that point can be used to continue the process. In addition to the SIT modules, any external binary tool can be integrated into the portable binary.
Each module is giving extensive user feedback via console output and creates logs of every relevant action taken. The console feedback allows the user to react to unexpected behavior by the software, unusually long acquisition times, or anticipated system failure on longer operations, by stopping the process at a suitable process step, while retaining the results up to that point. It is then possible to restart the process with different parameters, including previously disabled modules.

3.2.1. Artifact module

The artifact module consists of a modified version of a DFIR ORC tool called *GetThis*, developed as a forensically sound all-purpose file acquisition tool. It can acquire files by searching file system entries for parameters such as name, path, and size, determined during the configuration process. Its main focus is on NTFS file system entries, as for modern Windows systems, XP and newer, this is the default file system, with alternatives such as FAT being more relevant in external storage devices for example. Each file artifact is copied without changing the source and a wide range of metadata categories is also acquired, including MD5, SHA1, and SHA256 hash codes created immediately after acquisition. The results are then stored in ZIP archives, serving as backup and base for the next steps. Inside the

archive, the metadata is temporarily stored using a CSV file.

3.2.2. Validation module

Once the artifacts and metadata are acquired and stored in the backup archive, the validation module is executed, serving as a fail-safe that aims to identify unusual results and inform the user. The artifact module interacts most with the file system and files stored on the system and is therefore very vulnerable to interferences, data corruption, and crashes. As the goal is to detect such cases as quickly as possible, three steps are involved. Firstly each module is responsible for handling its errors reliably and documenting any unexpected behavior. Secondly, the artifact and metadata output is validated by the validation module checking if any inconsistencies, such as missing metadata for collected artifacts, missing artifacts for collected metadata, or incorrect data types can be identified. Lastly, verification of data integrity is done in the last step, the verification module. In addition to validating the output from the artifact module, the validation module also converts the metadata into an RDF turtle, in preparation for integrating it into the central metadata registry of the AFF4 image.

3.2.3. AFF4 module

AFF4, short for Advanced Forensics File (Format) 4, is an open, ZIP-based, extensible file format for storing evidence and case-related information. It uses an object-oriented approach to store data objects and metadata, using a central data store, called the resolver to manage references between objects. Each reference is maintained using Uniform Resource Identifiers (URI), made up of either internal AFF object Uniform Resource Names (URN), uniquely generated as part of the aff 4 namespace, or a more general Uniform Resource Locator (URL). Every AFF4 object has its own URN and can therefore be internally identified by the resolver and associated with its metadata (Cohen et al., 2009).
Metadata is a central aspect of AFF4 and can also exist independently from a data object. It is stored in (Subject, Attribute, Value) tuples inside a central RDF turtle file, bundled into unique URN entries, which allow *direct association* with the corresponding file object or identification as an abstract metadata object. Metadata that is not part of the AFF4 created data such as compression or size, is internally stored using an XML Schema Definition (XSD) type such as xsd:string or xsd:dateTime (Cohen and Schatz, 2010). While each object has its own URN, URLs may be used interchangeably with a URN to facilitate the sharing of evidence files between investigators (Cohen et al., 2009).

Verification Module. The verification module is part of the AFF4 module's source code, to improve efficiency and reduce storage space of the SIT binary, as it uses the same functions to access the AFF4 image. To perform the hash verification, all the artifacts stored in the AFF4 image are copied into the temporary directory and the MD5, SHA1, and SHA256 hash codes are calculated. These are then compared to the hash values collected by the artifact module upon acquisition and stored in each artifact's metadata set. Should any artifact have a missing hash code entry, the code calculation fail for any reason, or the codes not be equal, the verification is considered not successful for this artifact and the user is notified. The usage of multiple hash codes is intended to make unnoticed attacks on the integrity of collected evidence more difficult and in the case of SHA256 provide a collision resistant verification option conforming to current NIST (National Institute of Sta, 2015) guidelines.

References

ANSSI, 2020. DFIR ORC GitHub repository. https://github.com/DFIR-ORC/dfir-orc.
Casey, E., 2011. Digital Evidence and Computer Crime: Forensic Science, Computers, and the Internet. Academic Press.
Cohen, M., Schatz, B., 2010. Hash based disk imaging using AFF4. Digit. Invest. 17, 121–128.
Cohen, M., Garfinkel, S., Schatz, B., 2009. Extending the advanced forensic format to accommodate multiple data sources, logical evidence, arbitrary information and forensic workflow. Digit. Invest. 6, 57–68.
Faust, F., 2020. SIT GitLab Repository. https://gitlab.cs.fau.de/op64ycuz/sit.
Faust, F., Thierry, A., Müller, T., Freiling, F., 2020. Technical Report: Selective Imaging of File System Data on Live Systems. https://arxiv.org/abs/2012.02573.
Fröwis, M., Gottschalk, T., Haslhofer, B., Rückert, C., Pesch, P., 2020. Safeguarding the evidential value of forensic cryptocurrency investigations. Forensic Sci. Int. Digit. Invest 33, 200902. http://arxiv.org/abs/1906.12221.
Heinson, D., 2015. IT-Forensik : Zur Erhebung und Verwertung von Beweisen aus informationstechnischen Systemen, Veröffentlichungen zum Verfahrensrecht, vol. 119. Mohr Siebeck.
McKemmish, R., 2008. When Is Digital Evidence Forensically Sound? Springer US, pp. 3–15.
Moser, A., Cohen, M.I., 2013. Hunting in the enterprise: forensic triage and incident response. Digit. Invest. 10, 89–98.
National Institute of Standards and Technology, N, 2015. NIST Policy on Hash Functions. https://csrc.nist.gov/Projects/Hash-Functions/NIST-Policy-on-Hash-Functions.
Sack, K., 2017. Selektion in der Digitalen Forensik. PhD thesis. Friedrich-Alexander-Universität Erlangen-Nürnberg.
Stüttgen, J., Dewald, A., Freiling, F.C., 2013. Selective imaging revisited. In: Morgenstern, H., Ehlert, R., Freiling, F.C., Frings, S., Göbel, O., Günther, D., Kiltz, S. , Nedon, J., Schadt, D. (Eds.), Seventh International Conference on IT Security Incident Management and IT Forensics, IMF 2013, Nuremberg, Germany, March 12-14, 2013. IEEE Computer Society, pp. 45–58. https://doi.org/10.1109/IMF.2013.16.
Vömel, S., Freiling, F.C., 2011. A survey of main memory acquisition and analysis techniques for the windows operating system. Digit. Invest. 8 (1), 3–22. https://doi.org/10.1016/j.diin.2011.06.002.