Semantic Integration and Monitoring of File System Activity

Kabul Kurniawan[1,3][0000−0002−5353−7376], Andreas Ekelhart[1,2][0000−0003−3682−1364], Elmar Kiesling[1][0000−0002−7856−2113], Agnes Fröschl[1], and Fajar Ekaputra[1][0000−0003−4569−2496]

[1] TU Wien, Favoritenstraße 14, Vienna, Austria
[2] SBA Research, Floragasse 7, Vienna, Austria
[3] University of Vienna, Währingerstraße 29, Vienna, Austria

Abstract. File access activity information is an important source for identifying unauthorized data transmissions. In this paper, we present a semantic approach for the monitoring of file system activity in the context of information security. We thereby tackle limitations of existing monitoring approaches in terms of semantic integration, contextualization, and cross-system interoperability. In particular, we present a vocabulary for file activity logs and outline an architecture for log file collection, extraction, linking, and storage. We demonstrate the applicability of this approach by means of an application scenario. Finally, we show how analysts can inspect the life-cycle of files in a context-rich manner by means of SPARQL queries and a graph visualization of the results.

1 Introduction

Safeguarding the confidentiality and integrity of sensitive data is an increasingly difficult challenge that organizations face today. Data breaches can have a severe negative impact on the reputation, trustworthiness, and revenues of the affected companies, and also harm their customers and business partners. In the face of increasing data volumes and digitization, data exfiltration has become a critical concern [1]. Insider theft and improper handling of data is an even more difficult problem because users typically have legitimate access to data as well as a variety of channels to exfiltrate it at their disposal. Apart from various conventional protocols (e.g., ftp, ssh, scp, sftp), these channels also include cloud storage services, email, physical media (e.g., USB, laptop, mobile phone), messaging applications, and dns tunneling [4].

Once a data breach has occurred, a key prerequisite for an appropriate response is an understanding of its scope and impact. To this end, digital forensic methods are often applied to trace attack steps and determine the data affected. In this context, examiners often face the task of identifying data that has been copied to a removable storage device, uploaded to a user's personal cloud storage, or otherwise transmitted outside the physical or electronic walls of the organization to which the data belongs [5]. Although there are plenty of tools and

techniques that are employed during a digital investigation, the lack of integration and interoperability between them, as well as the formats of their source and resulting data, hinders the analysis process [2].

In this paper, we introduce a novel approach that leverages semantic web technologies to address these challenges. This approach has the potential to harmonize heterogeneous file activity information across various operating systems and logs, and facilitates contextualization through interlinking with relevant information and background knowledge.

To this end, we developed *(i)* a set of vocabularies for the uniform representation of file system log entries, which we introduce in Section 2; *(ii)* an architecture for file system log acquisition, file event extraction, and interlinking, covered in Section 3; and *(iii)* mechanisms that allow analysts to trace the life-cycle of files in a context-rich manner via a SPARQL query interface and a graph visualization, which we will illustrate by means of an example in Section 4.

2 Conceptualization and Vocabulary

Operating systems typically provide information about file system activity, i.e., clues when given files are being created, deleted, modified, renamed, copied, etc. Both Windows and Linux/Unix provide such information by logging multiple micro-operations. For instance, a single delete operation is commonly preceded by a sequence of operations such as handle requests and object access attempts. Consequently, these systems typically log several low-level (e.g., kernel) events that jointly indicate an event such as the copying of a file.

We collect file system activity data from heterogeneous file system logs (e.g., Windows event log, Linux audit log) and leverage a semantic model to harmonize them. Semantic representations provide machine-readability and facilitate interlinking between multiple log data formats, e.g., to trace the history of a file across systems. We organize our semantic model into two levels, i.e., *log entry* level and *file operation* level. As mentioned above, a single extracted log entry is typically not sufficient to represent a file operation event, therefore, we need to aggregate them and generate high level representations for file access events. For the *log entry* model, we extended a previously developed vocabulary [3] for generic log data. Due to space restrictions, we will not cover the file system *log entry* vocabulary in full detail and refer the interested reader to the source[4].

To model file operations, we chose a bottom-up approach, inspecting existing file activity logs and extracting the relevant concepts and terms. As depicted in Figure 1, the introduced *file operation* vocabulary[5] describes the concept of file access events (`fae:FileAccessEvent`) and a set of properties. The property `fae:hasAction` reflects the type of access taken to the file (e.g., create, modify, copy, rename, move, delete). The property `fae:hasUser` links the file event to the user accessing the file; `fae:hasProgram` represents the executable used to

[4] `https://w3id.org/sepses/vocab/log/win-event`, `https://w3id.org/sepses/vocab/log/unix-event`

[5] `https://w3id.org/sepses/vocab/event/file-access`

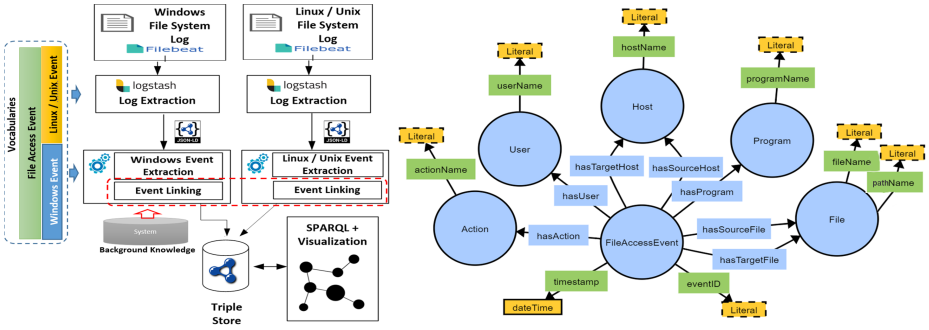

Fig. 1: System Architecture & Vocabulary

access the file, and `fae:timestamp` captures the time of the access. The properties `fae:hasSourceFile` and `fae:hasTargetFile` model the relation between an original file and a new file or location after the event. Furthermore, property `fae:hasSourceHost` and `fae:hasTargetHost` represent the hosts where the source and target files are located.

3 Solution Architecture

Figure 1 illustrates our approach, which is capable of monitoring file activity across multiple machines and operating systems. We use *Filebeat*[6], an open source tool, to ship raw file system logs from machines to the log extractor. Furthermore, we integrate *Logstash*[7] as an extraction component and configure it with custom filters in order to inject JSON-LD[8] `@context` annotations into the log stream. Consequently, the output conforms to our vocabulary introduced in the previous section.

The resulting JSON-LD log stream contains a large number of low level file system events. To aggregate these events, we developed a Java-based event extractor[9] that generates access events (e.g. create, modify, copy, rename, move, delete) based on low level (e.g., kernel-level) file system events. During event extraction, we also create links to existing background knowledge (e.g., about hosts and users)[10] to enrich the event data. Once the file access events have been generated, we can store them in a triple store[11] for further querying and analysis. We also developed a simple web-based graph visualization interface[12] that represents the life-cycle of a file visually.

[6] `https://www.elastic.co/products/beats/filebeat`
[7] `https://www.elastic.co/products/logstash`
[8] `https://json-ld.org/`
[9] `https://github.com/kabulkurniawan/fileAccessExtractor`
[10] `https://w3id.org/sepses/example/system-knowledge`
[11] We use Virtuoso in our prototypical implementation.
[12] `https://w3id.org/sepses/sparqlplus`

```
PREFIX fae: <http://w3id.org/sepses/vocab/event/file-access#>
PREFIX cl: <http://w3id.org/sepses/vocab/log/core#>
SELECT distinct  ?time ?sourceFile ?action ?targetFile  ?user ?host ?ipaddress WHERE {
  ?y fae:timestamp ?time.
  ?y fae:hasAction/fae:actionName ?action.
  ?y fae:hasUser/fae:userName ?user.
  ?y fae:hasSourceFile/fae:pathName ?sourceFile.
  ?y fae:hasTargetFile/fae:pathName ?targetFile.
  ?y fae:hasTargetHost ?h.
  ?h cl:hostName ?host.
  ?h cl:IPAddress ?ipaddress.
  ?x fae:hasSourceFile/fae:fileName "wordpad.rtf" .
  ?x fae:relatedTo* ?y .
}
  ORDER BY ASC(?time)
```

Listing 1: File Access Monitoring

4 Application Scenario

To illustrate the approach, we implemented an application scenario for file system monitoring, involving a Windows 8 and Ubuntu 18.04 file server and a folder with files we want to monitor. To simulate user activity, we implemented a batch file to trigger certain file access operations. The generated log data was shipped to the log extractor, automatically parsed, extracted, and finally stored in the triple store (cf. Section 3). With this data store, a security analyst can analyze the life-cycle of a certain file by means of SPARQL queries (cf. Listing 1). Starting from the filename (e.g., *wordpad.rtf*, we want to query all related file events (sequentially), including the timestamp, type of action, user, hostname, etc. Table 1 shows a query result for this example.

Figure 2 displays a visual representation of the result data, which makes it easier to follow the sequence of actions. As we can see, on the Windows system, *User1* renamed the file, then copied it to a Dropbox folder (*Dropbox\wordpad_ren.rtf*), and finally, moved the original file to the recycle bin. Due to Dropbox synchronization, the file *wordpad_ren.rtf* was created on the Linux server as well. Subsequently, the Linux user *User2* moved the document to the folder *UserDoc* and from this location copied the file to a removable media (*/Volumes/USER2*). Finally, this user deleted the original file from his folder. In an interactive model, the analyst could display additional information from the vocabulary (e.g., the timestamps) for further inspection.

Time	SourceFile	Action	TargetFile	User	Host	IPAddress
16:34:16	C:\..wordpad.rtf	created		User1	Win8	192.168.1.2
16:34:38	C:\..wordpad.rtf	renamed	C:\..wordpad_ren.rtf	User1	Win8	192.168.1.2
16:35:10	C:\..wordpad_ren.rtf	copied	C:\..Dropbox\wordpad_ren.rtf	User1	Win8	192.168.1.2
16:35:10	C:\..wordpad_ren.rtf	deleted	C:\..Recycle.Bin\..GHH.rtf	User1	Win8	192.168.1.2
16:36:12	/User2/Dropbox/wor..ren.rtf	created		User2	Ubuntu	192.168.1.3
18:03:24	/User2/Dropbox/wor..ren.rtf	moved	/User2/UserDoc/wor..ren.rtf	User2	Ubuntu	192.168.1.3
18:04:25	/User2/UserDoc/w..ren.rtf	copied	/Volumes/USER2/w..ren.rtf	User2	Ubuntu	192.168.1.3
18:06:06	/User2/UserDoc/w..ren.rtf	deleted	/User2/.Trash/w..ren.rtf	User2	Ubuntu	192.168.1.3

Table 1: File Access Query Results

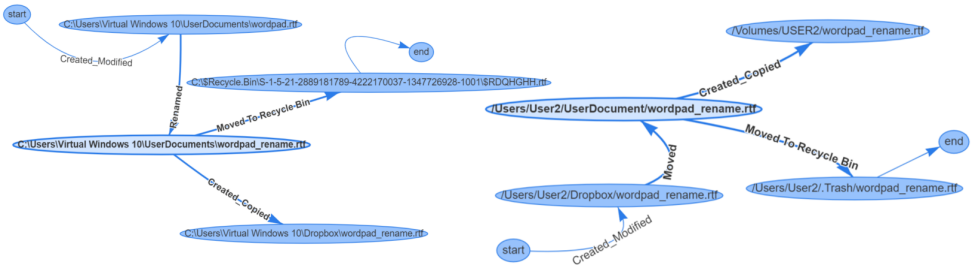

Fig. 2: File Access Graph Representation

5 Conclusion and Future Work

In this paper, we introduced a semantic approach to understand file events and their change history. We thereby tackle the lack of interoperability of existing file monitoring solutions to support security analysts, e.g., in detecting the exfiltration of sensitive data. Our framework collects file activity logs from heterogeneous sources, integrates and semantically enriches the information with background knowledge, and stores the results as RDF triples. We developed a vocabulary to uniformly represent the file system log entries and a prototype implementation of the solution concept. Furthermore, we demonstrated the applicability of the approach by means of an application scenario, simulating user-triggered file activity on a Windows and Linux system. Finally, we present a SPARQL query to trace the life-cycle of a file and present a graph visualization to support understanding.

Based on this work in progress, we will extend and optimize the prototype and the interactive visualization for analysts, and conduct an extensive evaluation of the approach. To this end, we will develop additional scenarios and extend our testing environment to support complex, multi-system scenarios. Another future research direction is to investigate the potential of stream reasoning engines for continuous monitoring of security-relevant file system activities as well as the use of Ontology-Based Data Access (OBDA) methods for large-scale log monitoring.

References

1. Cheng, L., Liu, F., Yao, D.: Enterprise data breach: causes, challenges, prevention, and future directions. WIREs Data Mining and Knowledge Discovery (2017)
2. Cuzzocrea, A., Pirró, G.: A semantic-web-technology-based framework for supporting knowledge-driven digital forensics. In: Proceedings of the 8th MEDES Conference (2016)
3. Ekelhart, A., Kiesling, E., Kurniawan, K.: Taming the logs vocabularies for semantic security analysis. In: Proceedings of the 14th SEMANTiCS Conference (2018)
4. Gordon, P.: Data leakage - threats and mitigation. Report, SANS Institute Reading Room (2007)
5. Sindhu, K., Meshram, B.: Digital forensic investigation tools and procedures. Computer Network and Information Security (2012)