An Ontology-Based Approach for the Reconstruction and Analysis of Digital Incidents Timelines

Yoan Chabot[a,b], Aurélie Bertaux[a], Christophe Nicolle[a], Tahar Kechadi[b]

[a]*CheckSem Team, Laboratoire Le2i, UMR CNRS 6306, Faculté des sciences Mirande, Université de Bourgogne, BP47870, 21078 Dijon, France*
[b]*School of Computer Science & Informatics, University College Dublin, Belfield, Dublin 4, Ireland*

Abstract

Due to the democratisation of new technologies, computer forensics investigators have to deal with volumes of data which are becoming increasingly large and heterogeneous. Indeed, in a single machine, hundred of events occur per minute, produced and logged by the operating system and various software. Therefore, the identification of evidence, and more generally, the reconstruction of past events is a tedious and time-consuming task for the investigators. Our work aims at reconstructing and analysing automatically the events related to a digital incident, while respecting legal requirements. To tackle those three main problems (volume, heterogeneity and legal requirements), we identify seven necessary criteria that an efficient reconstruction tool must meet to address these challenges. This paper introduces an approach based on a three-layered ontology, called *ORD2I*, to represent any digital events. *ORD2I* is associated with a set of operators to analyse the resulting timeline and to ensure the reproducibility of the investigation.

Keywords: Digital Forensics, Event Reconstruction, Forensic Ontology, Knowledge Extraction, Ontology Population, Timeline Analysis

1. Introduction

Nowadays, digital investigations require the analysis of a large amount of heterogeneous data. The study of these volumes of data is a tedious task which leads to cognitive overload due to the very large amount of information to be processed. To help the investigators, many tools have been developed. Most of them extract unstructured data from various sources without bridging the gap of semantic heterogeneity and without addressing the problem of cognitive overload. To resolve these issues, a promising perspective consists of using a precise and reliable representation allowing to structure data on the one hand, and to standardise their representation on the other hand. A structured and formal knowledge representation has two goals: 1) to build automated processes more easily by making information understandable by machines and 2) to give to investigators an easy way to query, analyse and visualise information. Computer forensics investigations also have to fulfil a set of legal and juridical rules to ensure the admissibility of results in a court. It is particularly necessary to ensure that all evidence presented at a trial are credible and that the methods used to produce evidence are reproducible and did not alter the objects found in the crime scene. Problems of traceability and reproducibility of reasoning are widely discussed in the literature and provenance is particularly relevant to be applied to digital investigations. Indeed, as defined by (Gil & Miles, 2013), the provenance of a resource is a record describing the entities and the processes involved in the creation, dispersion or others activities that affect that resource. The provenance provides a fundamental basis for assessing the authenticity and the truth value of a resource and its reproducibility.

In our work, we propose an innovative digital forensic approach based on a knowledge model allowing to represent accurately a digital incident and all the steps used during an investigation to produce each result. In addition, a full set of operators to manipulate the content of this ontology is proposed. We introduce extraction and instantiation operators to build automatically the knowledge base using digital traces extracted from disk images. Then, automatic analysis operators taking advantage of this ontology are also proposed. In particular, we focus on an operator used to identify potential correlations between events.

This paper is structured as follow: Section 2 gives a comprehensive state of the art of event reconstruction approaches for digital forensics. Section 3 introduces the *SADFC* approach. In particular, the structure of the three-layered ontology is detailed and its various operators are described. Section 4 evaluates the performance of our approach and illustrates its capabilities on a case study.

2. State of the art

Event reconstruction is a complex process because of three main issues: the large volume of data, the heterogeneity of information due to the use of a large number of sources and the legal requirements that the results have to meet. This section

Email address: `yoan.chabot@hotmail.fr` (Yoan Chabot) aims to review the existing solutions described in the literature to reconstruct and analyse past events. The quality and the relevance of nine reconstruction approaches are reviewed based on the previous three issues and their limitations. These answers are then synthesised and used to guide the development of our approach to ensure that the three main issues of event reconstruction are taken into account.

2.1. Data volume

Currently, investigators are facing huge data volumes on a digital crime scene. The growth of the data size is caused by several factors. Digital devices are more and more present in our daily lives. This increases the number of devices owned by each person and therefore the number of devices found on the crime scenes. In addition, the frequency of use and the increase of storage capacity of the digital devices have caused the increase in the quantity of data stored by each device. The very large amount of data makes the analysis very complex and tedious, even causing cognitive overload. Information that is potentially relevant to reach the objectives of an investigation is diluted in the amount of data, making the investigation difficult. For example, the Plaso toolbox (Gudhjonsson, 2010), which produces timeline from hard disk image, can identify thousands of events from a wide range of sources (Apache logs, Skype conversations, Google Chrome history, Windows event logs, etc.) from an image of only a few gigabytes. In conclusion, an event reconstruction approach should be able to efficiently process information and retrieve and visualise the results in a clear and intuitive way.

In the literature, a large number of approaches provide tools to automatically extract the information and populate a central storage constituting the timeline. ECF (Chen et al., 2003) is an approach aiming to extract, store in a database and manipulate events from heterogeneous sources. It introduces a set of extractors to collect events and store them in a database, which quickly generates a temporal ordered sequence of events. These automatic extractors, a widely used concept, can also generate the timeline as in FORE (Schatz et al., 2004), FACE (Case et al., 2008), CyberForensic TimeLab (Olsson & Boldt, 2009), Plaso and PyDFT (Hargreaves & Patterson, 2012). However, in some approaches including (Gladyshev & Patel, 2004) and (James et al., 2010), the lack of automation seems difficult to address and they present very high complexity (combinatorial explosion).

The automated extraction of events from a large number of sources leads to the creation of large timeline that is difficult to read, interpret and analyse. To assist the investigators during this phase, event reconstruction approaches should provide analysis tools to carry out all or part of the reasoning and visualisation of the data in a clear and intuitive way. The use of a textual representation or a database is not suited to build high-performance analysis process. In approaches, such as ECF, the reasoning capabilities are limited by the data structure used for the storage of information.
ECF stores data in a database consisting of a table of common information about events and tables of information specific to each type of event. One of the objectives of ECF is to provide a canonical form for representing events uniformly regardless of their sources. Adopting a generic information level and a specialised information level is also used in CybOX (Barnum, 2011). This conceptual separation allows to introduce a canonical representation of events that facilitates information processing (analysis tasks for example) while preserving the specificities of each event. However, the use of a database does not allow to take full advantage of this feature. Unlike ontology, databases do not enable to explicitly represent semantic of data which constrains the understanding of data by the analysis algorithms. The FORE approach (Schatz et al., 2004) proposes a correlation tool based on rules to identify causal relationships (the event A causes the event B if the event A should happen before the event B) between events. Despite the relevance of this tool, the use of a rule-based system requires the definitions of the rules. The construction of such a set of rules is a tedious task and it cannot take into account all cases. Event reconstruction approaches must implement algorithms that can adapt to any kind of situations even those unknown by the investigators. It is, therefore, necessary to develop analysis tools not relying on the rules defined by the user. In (Gladyshev & Patel, 2004), the reconstruction process can be seen as a process of finding the sequence of transitions that satisfy the constraints imposed by the evidence. The authors try to perform the event reconstruction by representing the behaviour of the system as a finite state machine. Scenarii that do not match evidence collected are then removed. After reducing the number of potential scenarii, a backtracking algorithm is used to extract all possible scenarii. However, the lack of automation does not allow to handle complex cases. Indeed, the investigation of a single computer may involve several processes such as web browsers, file system, instant messaging software, etc. Thus, the representation of such a system with a finite state machine seems not possible. Second, the use of finite state machine is very time consuming when used in real case. (James et al., 2010) propose to convert the finite state machine into a deterministic finite state machine to limit the exponential growth of the size of the machine and therefore the number of scenarii to examine during the backtracking algorithm. Despite the reduction in size of the state machine, the experiments show that the approach still not be able to be used on real forensic cases. To allow the handling of large volumes of data, (Khan et al., 2007) introduces a neural network-based approach using footprints left in a machine to detect software activities. The main motivation of this work is to show the ability of machine learning techniques to process large datasets. The proposed tool is able to extract footprints from various sources including log files and registry. A preprocessor is then used to make data usable by the neural network that is then used to identify launched software. Unfortunately, the performance of the proposed tool is poor. The training of the neural network and the need to use it several times to get a complete scenario make the use of this tool time-consuming. In (Case et al., 2008), an approach called FACE, for Forensics Automated Correlation Engine, focusing on the interpretation and the analysis of the

data was proposed. It aims at collecting data from various sources and then identifying correlated events. At the end of the process, the investigators get a report describing the events, objects and logical relationships between events and between events and objects. The main contribution of this work is the introduction of a correlation tool handling a part of the analysis. PyDFT (Hargreaves & Patterson, 2012) is proposed as a system allowing to reconstruct high-level events from low-level events (which are extracted from information sources by log2timeline or Zeitline) in order to improve readability of the final timeline. The authors stated that the number of events generated by the super-timeline approaches make the visualisation, and therefore the analysis of the timeline, complex. To facilitate the reading of the timeline, this tool produces a timeline summary. Once the low-level timeline is built, we look for patterns and their corresponding high-level events are then added accordingly. The summarisation of a timeline is a relevant functionality as it facilitates the reading of the timeline and by extension, its analysis. A similar system is presented in (Turnbull & Randhawa, 2015). In this work, a rule-based tool, called ParFor, identifies events from low-level artefacts extracted from disk images.

Concerning the visualisation, some approaches offer ergonomically ways to access information like (Case et al., 2008) where reports, describing the entities and relationships between them, are generated by the system. Zeitline (Buchholz & Falk, 2005) is an editor allowing to create a timeline from multiple sources of information. The tool does not offer automatic analysis process, but invites the investigators to handle themselves the aggregation of events to create high-level events. The interface of the tool allows to add new events to a given timeline, to aggregate several events to build a complex event or to search for specific events using a keyword-based query tool. In (Case et al., 2008), the proposed tool is able to generate a report that offers different views on events and objects in addition to hyper links between them to make the timeline easier to read and more intuitive. In (Olsson & Boldt, 2009), a system called CyberForensic TimeLab was proposed. The focus of this work is on the conception of a system to view and navigate into the data related to an investigation in an intuitive way to discover evidence more easily. Then, the timeline is displayed on a graphical browser which is the main added value of this approach as it constitutes an improvement of the ergonomics.
Many tools do not offer an intuitive interface but only a query tool that appears to be a powerful but complex and tedious way to access the information. This is the case of approaches using databases and providing a SQL query interface which is efficient but not intuitive for an untrained user.

To handle large volume of data, we conclude that event reconstruction approaches must meet the following requirements:

- The automation of certain tasks which are becoming too complex to be carried manually. In order to produce tools that are able to process large volumes of data, it is necessary to automatically reconstruct and analyse timelines. The conception of automated tools requires data representation that is understandable for machines (structured, formal and with a clearly-defined and complete semantic). Therefore, we evaluate the level of the data structure to enable their analysis by machines in addition to the availability of mechanisms to facilitate the use of automation.

- The availability of tools for interpreting, analysing information and drawing conclusions. The aim of analysis tools is to highlight relevant information with the view to guide the investigators. This includes making the timeline easier to read by filtering data or summarising the timeline as proposed in (Abbott et al., 2006) and (Hargreaves & Patterson, 2012), identifying correlations between events as in (Schatz et al., 2004) and (Case et al., 2008) and producing conclusions from knowledge contained in the timeline.

- The availability of visualisation tools allowing to browse data in an efficient, clear and intuitive way. In addition to the availability of the data, the model must also allow to access to data and information easily. In particular, models should allow the use of search and visualisation tools to present the data in an understandable and intuitive form.

2.2. Heterogeneity

The second issue is the heterogeneity since the information is spread across the digital crime scene. During an investigation, many sources of information can be used: web browser histories, windows event log, file system, logs of various software, etc. In order to not miss information and to obtain a real and accurate picture of the incident, the scenario reconstruction approaches must be able to extract the information from all of these sources and process them appropriately. It should be noted that there are several kind of heterogeneity: syntactic, semantic and temporal. In this work, we study the techniques related to the first two as they are both related to our desire to reduce the cognitive overload by providing a clear and complete view on the past events.

A large part of existing approaches is able to deal with multiple and heterogeneous sources including Windows event logs, web browser histories, Apache server logs, files meta data, instant messaging software, registry, memory dumps, networks activities and user configuration files. The handling of heterogeneous sources requires on the one hand the development of automated extraction of information specific to each source and on the other hand a sufficiently complete knowledge model to represent all aspects of a digital incident. In (Chen et al., 2003), the model introduced in ECF defines the dimensions characteristics to represent forensic events such as a temporal dimension, another one related to objects used by events and a last one related to protagonists which are involved in events. As the FORE approach (Schatz et al., 2004), the ECF model does not define relationships between entities (subject, object, event). For example, it allows to model the fact that an event interacts with an object but the nature of this interaction cannot be defined accurately. FORE (Schatz et al., 2004) is an approach introducing a knowledge representation model for events occurring during an incident. In addition, a set of extractors able to extract knowledge from sources such as Apache server logs and Windows 2000 logs to populate the proposed ontology was introduced. The FORE ontology consists of classes that represent the notion of Entity (i.e. objects composing the world) and the notion of Event (i.e. the change of state of an object over time). The Event may itself be inherited by others classes in order to describe different types of events that can occur on a machine. A number of attributes are used by classes inheriting from the Event class to provide information about the user or the process that produced the event, the files, etc. The use of an ontology to represent events is an efficient way to deal with heterogeneity issues and allows the introduction of semantically rich models that are able to capture the semantics of all entities and the relationships between them. The precise and formal description of the components of the model can significantly increase analysis capabilities by enabling machines to understand the meaning of the data (in contrast to unstructured formats). The Plaso toolbox[1] is a tool that can handle a large number of information sources. Plaso (Gudhjonsson, 2010) is able to generate super-timeline (timeline integrating many sources of events) from a wide range of sources. Among tools proposed in Plaso, log2timeline allows to extract events from a disk image and psort can be used to format the result produced by log2timeline as a text file, a CSV file, a database, etc. The output format is composed of a limited number of fields to store the date and time of events, the source that has been used for the extraction of the event and a message describing the event. Using a small number of features offers better performance than more complex models. However, the features introduced in these data formats do not allow to accurately represent any event occurring in the system. Moreover, the use of a text format does not allow to explicitly and simply show the relationships between entities. Data formats offer interesting performance for simple use-cases but are limited in terms of expressiveness for complex use-cases. Some approaches suffer from lack of sufficient number of sources which can lead to a loss of relevant information. In addition, most of the proposed models are not sufficiently complete to accurately describe an incident.

To deal with the problems of heterogeneity, an approach must meet the following points:

- The completeness of the knowledge model. The proposed model must be complete enough to represent accurately the events that occur during an incident. In other words, the model must provide a sufficiently elaborated vocabulary to fully represent the entities related to a digital incident, their characteristics and the relationships between them.
- The implementation of mechanisms (e.g., parsers) to process heterogeneous sources.

2.3. Legal requirements

The last issue we are focusing on is the legal requirements that any digital forensic approach must meet. Indeed, the results produced as outputs of an approach must meet certain characteristics including credibility, integrity and reproducibility (Baryamureeba & Tushabe, 2004). The challenge is to ensure that the results are admissible in a court of law.

Few approaches are able to explain how the results are obtained. Traceability is particularly lacking in (Khan et al., 2007). Unlike others machine learning techniques, the use of neural networks does not usually allow to easily interpret the results (which is one of the justice requirements) as they can be considered as black boxes (some parameters used during the learning phase remain unknown). The model proposed in (Hargreaves & Patterson, 2012) allows to store low-level and high-level events, and is composed of two structures to represent them. The first one includes attributes to model the date, the type of the event and the source used to identify it. The second class has a similar structure, in addition it includes attributes to memorise how each high-level event is generated using low-level events. This allows to store information about data provenance. Only (Gladyshev & Patel, 2004) has theoretical support. As a prelude to their work, the authors argued that a formalisation of the event reconstruction problem is needed to simplify the automation of the process and to ensure the completeness of the reconstruction. The use of finite state machine, a theory well-known in the scientific community, gives strong theoretical foundations to this approach.

To meet legal requirements, an event reconstruction approach must meet the following points:

- The integration of information into the model to ensure the traceability of information. Indeed, to meet legal requirements, the evidence produced in a court must meet several criteria including credibility, integrity and reproducibility of the digital evidence (Baryamureeba & Tushabe, 2004). The level of credibility of evidence is directly related to the level of accuracy and verifiability of the method and the source of data used to produce the evidence. By detailing the investigation process, the modelling of the provenance of information can increase significantly the credibility of all evidence produced. The reproducibility allows independent bodies to reproduce the same conclusions using the same data inputs and allow to evaluate the quality of this process. It allows a court to fully understand the path (composed of data manipulation steps, reasoning steps, etc.) used to reach each conclusion which are presented during a trial. Thus, a model must be able to integrate information on the provenance of the evidence produced during the investigation. This includes storing each step of an investigation, the investigators involved in activities, the tools or the techniques used to produce the

[1] `http://plaso.kiddaland.net/`

information (e.g. extraction from a data source, deduction from two pieces of information already known, etc.).

- In order to produce credible results, a model based on formal theory is needed.

To summarise, we have identified seven criteria necessary to reduce the cognitive overload, deal with heterogeneity, and meet the legal requirements. These criteria were used to guide the development of an innovative approach based on an ontology which is presented in the following section.

3. An ontology-based approach for event reconstruction

To fulfil the seven criteria highlighted in Section 2, we introduced an approach based on ontology. This section aims to show how this approach meets the problem requirements and it is structured as follows. Section 3.1 gives an overview of the proposed approach and justifies the conceptual choices that have ruled its development. Section 3.2 introduces the concepts and the relationships of the ontology representing accurately a digital incident. Section 3.3 presents the implemented tools to extract knowledge from a crime scene and populate the ontology accordingly. Finally, Section 3.4 and Section 3.5 respectively introduce the tools used to carry out aspects related to reasoning and the interfaces for the visualisation of the results.

3.1. Overview

To process a large amount of data collected during an investigation, there is a need to develop automatic processes to assist investigators in the processing and the interpretation of data. Currently, a large number of forensic tools (Bodyfile, Mactime (Farmer & Venema, 2004)) works on unstructured data as plain text which does not allow to build advanced analysis process. This is mainly due to the lack of semantic information. A promising perspective is to use a precise and reliable representation allowing to structure data on one hand, and to standardise their representation on the other hand. Introducing a structured and formal knowledge representation pursues two goals: build automated processes more easily by including semantic information and make data easy to use and understandable by the users (graph visualisation, query tools, etc.). The main component of the proposed approach is an ontology implemented using OWL 2[2], a language based on description logic. An ontology is a model allowing to represent knowledge of a given domain by structuring this knowledge using entities, relationships and logical constraints. Thus, ontologies are able to represent accurately the knowledge generated during an investigation (knowledge about footprints, events, protagonists etc.). (Gruber et al., 1993) defined an ontology as "an explicit, formal specification of a shared conceptualisation". Unlike more rudimentary data formats, ontology can represent relationships between entities in addition to the underlying logic of data. The explicit and formal nature of ontology facilitates the design and the use of interpretation and analysis tools (inference of new information, checking of the knowledge consistency, etc.). Therefore, ontology is the data structure that is best suited to meet the requirements mentioned above because of the inherent characteristics of the ontology which allow to answer criteria of soundness, ease of automation and ease of use.

The ontology *ORD21* (Ontology for the Representation of Digital Incidents and Investigations) is implemented using the OWL profile OWL 2 RL[3] (a subset of OWL 2 DL), a language based on SHROIQ(D) description logic. The use of this profile is motivated by several reasons including the availability of sufficient expressiveness to model digital forensic cases. Using OWL 2 RL allows to constrain some aspects (class expressions in particular) of the language to ensure decidability and execution time (polynomial) of rule-based reasoning (Motik et al., 2009). The need to operate on large volumes of data and the desire to provide a powerful inference and analysis tools make OWL 2 RL pertinent to implement an ontology for the representation of digital incidents timeline.

The OWL 2 DL language also allows to give credibility to our model as it has a strong theoretical foundation. Indeed, this ontology language is based on description logic that defines formally the semantics of concepts and relationships. The components of the ontology and the logic associated to each of them are therefore mathematically defined which allow to check the coherence and the consistency of the knowledge. In addition, the ontology presented in this paper is based on formal definitions given in (Chabot et al., 2014).

Ontologies are also very easy to manipulate due to the use of tools such as SPARQL[4], a query language designed to query graph knowledge. Because, the structure is in the form of triples <subject, predicate, object>, it is possible to visually represent ontologies as graphs. This kind of graphical representation is very intuitive and clearer than a textual representation of data. Finally, as stated in Gruber's definition, ontologies are designed to help to build a common view of a domain, which means that this kind of structure is particularly relevant to build a consensus on the representation of events among digital forensics investigators and software developers.

Our overall system consists of an ontology and four layers which are knowledge extraction, ontology management, knowledge analysis and visualise of the final results (Figure 1). First, the *Extraction Layer* provides tools to extract knowledge from an image seized on a crime scene. In order to process heterogeneous and large information sources, the extraction process takes advantage of Plaso, a toolbox allowing to extract automatically information from a large number of sources including Windows registry, browsers histories, file systems and others (i.e. Skype, Google Drive, Java IDX, etc.). The knowledge extracted is then used to populate the ontology. The *Knowledge Layer* is used to filter the knowledge extracted and to instantiate the ontology with it. This layer also provides

[2]http://www.w3.org/TR/owl2-overview/

[3]http://www.w3.org/TR/owl2-profiles/

[4]http://www.w3.org/TR/rdf-sparql-query/

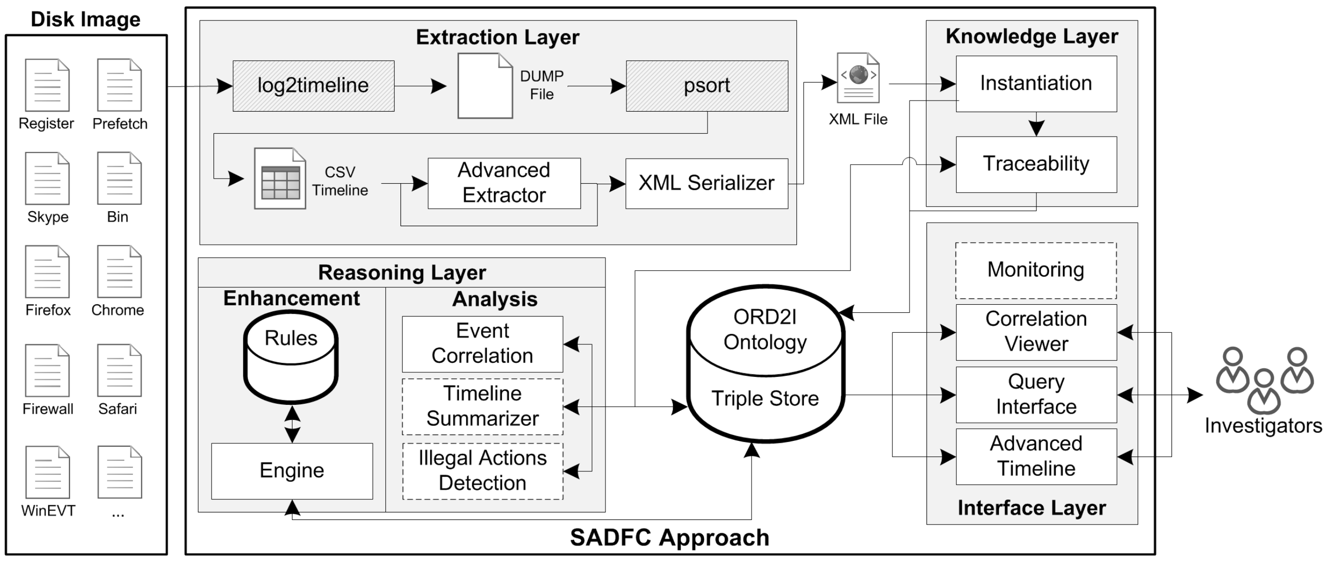

Figure 1: Overview of the ontology-based approach made of four layers (boxes in light gray) and fourteen modules (white boxes and solid black line: operational modules, white boxes and dashed black line: modules not yet implemented, hatched boxes: modules from the toolbox Plaso, data streams are represented using solid black arrow)

mechanisms to manage traceability information. After the instantiation of the ontology, the *Reasoning Layer* is used to help analysing and interpreting the knowledge about the incident. This layer, on the one hand, deduces new facts using knowledge about the incident (Enhancement) and on the other hand analyses and draw conclusions. Finally, the *Interface Layer* provides visualisation tools and a query tool to display the data in an efficient and intuitive way.

3.2. *ORD2I: an Ontology for the Representation of Digital Incidents and Investigations*

The ontology *ORD2I* is built on four bases:

- Design an ontology that is complete and accurate enough to model any digital incident.
- Integrate generic entities to the ontology to facilitate the analysis (canonic form).
- Take into account specialised knowledge from forensic experts and software developers to integrate the characteristics of each information source.
- Model provenance of information.

To meet these needs, *ORD2I* is divided into three layers, each allowing to model different kinds of knowledge and answering different questions: the *Common Knowledge Layer* (*CKL*) addresses the need to integrate generic entities to the ontology, the *Specialised Knowledge Layer* (*SKL*) handles specialised knowledge and the *Traceability Knowledge Layer* (*TKL*) deal with traceability. Each of these layers is presented below and illustrated in Figure 2. This figure shows the classes (*owl:Class*) composing each layer and the properties linking these classes. Object properties (*owl:ObjectProperty*) are represented using standard solid arrows and inheritance links (*rdfs:SubClassOf*) are represented using UML generalisation relationship. To enhance the readability of the figure, the class attributes and the namespaces are not represented.

The proposed ontology is inspired by CybOX (Barnum, 2011) and the ontology PROV-O (Lebo et al., 2013). In the CybOX project (Barnum, 2011), a set of XSD schemas are introduced to represent any entity (process or object, i.e. cyber-observables) observed during an incident in addition to the interactions between the cyber-observables and events affecting them. The proposed schemas also allow to integrate knowledge from experts through a system of objects specialising the abstract notion of an object. The large set of objects offers the ability to represent a PDF file, an HTTP session, a web history, a Windows process, a network connection, etc. CybOX was then enriched by DFAX (Casey et al., 2015), an ontology representing information about the provenance of information. DFAX provides an extra layer to represent forensic actions initiated by the investigators. PROV-O (Lebo et al., 2013) is a W3C recommendation that describes an ontology for representing the provenance of information. PROV-O was used as a starting point for the design of the *Traceability Knowledge Layer*. This ontology is composed of concepts and relationships allowing to define a piece of information, to assign this information to a user or an entity and to represent the process used to produce the information.

3.2.1. *Common knowledge about the incident*

The Common Knowledge Layer (*CKL*) stores common knowledge about events that occurred during an incident. This layer contains time information about events, information about their resources and the information on the subjects involved in the event. This layer provides a canonical representation of events using the principle of polymorphism. Due to the

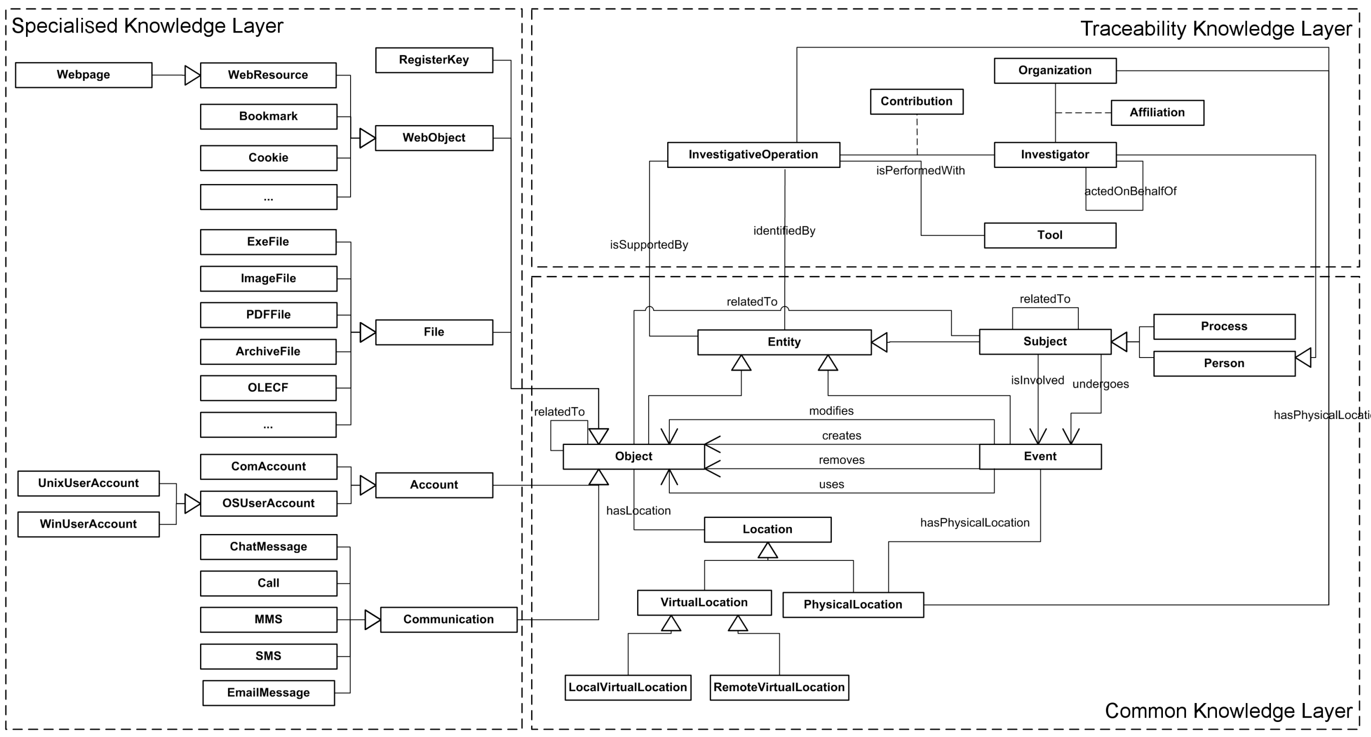

Figure 2: Overview of *ORD2I*

variety of sources, all events have a number of specific attributes and a set of common characteristics. For example, if we consider an event produced by an Apache server and another event from the Firefox browser, it is difficult to analyse such events to look for new knowledge because each of them has specific characteristics. An Apache event carries out information about IP addresses, which is requested by the connection, whereas Firefox does not need it. However, these two events have also common characteristics (e.g., the time at which the events occurred). All events belong to the same concept which is the general concept of *Event*. The use of polymorphism creates a consistent representation for all events without removing the specific data of each type of event (specific data is stored in the Specialised Knowledge Layer (*SKL*)). This uniform representation allows events to be analysed in the same way. The concepts and relationships within *CKL* are illustrated in the bottom right part of Figure 2. Concerning *CKL*, the class *ord2i:Entity* is subsumed by the classes *ord2i:Event*, *ord2i:Subject* and *ord2i:Object*. First, the *ord2i:Event* class allows to model any digital action happening on a machine and is defined by the type of the action performed (e.g. ”File Creation”, ”Webpage Visit”, etc.), the date and the time when the action occurred and the accuracy and the granularity of the date and a status (e.g. ”success”, ”fail”, ”error”, etc.). Dates are defined using the *Interval* class of the OWL-Time ontology. An interval is defined by a start and an end allowing to represent the notion of uncertainty. For example, when the start time of an event cannot be determined accurately, the use of a time-interval allows us to approximate it. The *ord2i:Subject*, *ord2i:Process* and *ord2i:Person* classes are used to model protagonists involved in the events. A subject can participate (*ord2i:isInvolved*) in an event or undergoes (*ord2i:undergoes*) an event. The *ord2i:Object* class is used to represent resources that are used (*ord2i:uses*), modified (*ord2i:modifies*), removed (*ord2i:removes*) or created (*ord2i:creates*) by events. An object has a state used to define the status of a resource (e.g., exists, does not exist, closed, locked, unlocked, etc.).

3.2.2. Specialised knowledge about the incident

The Specialised Knowledge Layer (*SKL*) is used to store specialised information of objects. *SKL* models technical knowledge about every kind of objects that can be found in a cyberenvironment. The concepts composing *SKL* are illustrated in the left part of the Figure 2. One advantage of *SKL* is to manipulate technical knowledge about events. For example, IP addresses, file paths or metadata files are all valuable information during analysis. The objective of *SKL* is to provide the structure needed for this technical knowledge. *SKL* is not intended to be exhaustive and the use of an ontology allows to integrate easily new classes or to modify existing ones. All classes and attributes of *SKL* do not appear on the figure to improve its readability. This layer provides an important range of classes to represent a large number of objects:

- Files (*ord2i:File*): *ord2i:Link*, *ord2i:ArchiveFile*, *ord2i:ImageFile*, *ord2i:PDFFile*, *ord2i:ExeFile*, etc.
- User account (*ord2i:Account*): *ord2i:UnixUserAccount*, *ord2i:WinUserAccount*, *ord2i:ComAccount*.
- Objects related to the Web (*ord2i:Web*): *ord2i:Webpage*, *ord2i:WebResource*, *ord2i:Bookmark*, *ord2i:Cookie*,

ord2i:Website, etc.

- Objects related to digital communications (*ord2i:Communication*): *ord2i:MMS*, *ord2i:SMS*, *ord2i:ChatMessage*, *ord2i:Call*, *ord2i:EmailMessage*.
- Registry keys (*ord2i:RegisterKey*).
- etc.

3.2.3. *Traceability and reproducibility of the investigation process*

The Traceability Knowledge Layer (*TKL*) stores information about how the investigation is conducted. This includes information about investigative activities, the information used and the agents involved. For example, this layer is used to memorise how each result of the investigation is produced. The aim of this layer is to satisfy some legal requirements It ensures reproducibility of the results by storing all actions taken at each stage of the investigation. It also ensures that the conclusions have been supported by credible data and evidence. The concepts and relationships within *TKL* are illustrated in the upper part of Figure 2. The *ord2i:InvestigativeOperation* represents any task undertaken during an investigation. Each task is characterised by a set of attributes including: the type of techniques used (i.e. extraction from an information source, inference of new knowledge, event correlation, etc.), the information source used (i.e. Windows registry, Web browsers histories, etc.), the date and the place where the task was performed, a description of the task and a numerical value quantifying the degree of confidence of the result of the task. For example, in the case of an operation using information that may have been corrupted or obfuscated by attackers, the truthfulness will be low because the results of the operation are unreliable. Each task belonging to the *ord2i:InvestigativeOperation* class is used to deduce new entities in order to know what happened in the past by identifying events that occurred and the subjects and the objects that have interacted with events. Therefore, the *ord2i:identifiedBy* object property models the fact that every entity is identified using an investigative operation (e.g. the task of extracting information from a web history can lead to the identification of an event of bookmark creation or the identification of a webpage visited by the user). For some kind of investigative task, investigators have to use information that is already known to deduce new knowledge. The *ord2i:isSupportedBy* object property models this fact by linking an investigative operation to the information used by it to deduce new knowledge. Each instance belonging to the *ord2i:InvestigativeOperation* class is linked to the tools (*ord2i:Tool*) used and the investigators (*ord2i:Investigator*) involved using respectively the *ord2i:isPerformedWith* property and the association class *ord2i:Contribution*.

3.3. *Extraction and Instantiation of ORD2I*

This section aims to illustrate the instantiation of *ORD2I* using data extracted from a crime scene. The integration of automated instantiation techniques is critical to process large volumes of data extracted during an investigation. The method of instantiation used in our approach is a sequential process starting with the collection of digital traces found on a disk image and ending with the instantiation of concepts and properties of *ORD2I*.

3.3.1. *Information extraction of digital footprints using the Plaso toolbox*

The first step is the extraction of all the information contained in the various sources of the image disk under investigation. During an investigation, many sources can be used to obtain information about what happened. To manage all these sources, the tool log2timeline (Gudhjonsson, 2010), proposed as a component of Plaso, is used. log2timeline collects information from many sources including: sources inherent to the operating system (e.g. registry, file system, recycle bin, etc.); histories, cookies and cache files of web browsers; files and logs generated by various software including Skype, Google Drive, etc. The result is a DUMP file containing all the footprints retrieved from the disk image. Then, transforming the result is necessary to make the data usable by downstream processes. For this, the tool psort of Plaso is used. This tool allows to serialise the data produced by log2timeline in many formats including CSV. An example of the output of the tool is given in Figure 3. This dataset was constructed from a disk image of a virtual machine. Each entry of this file describes an action that has occurred:

- **E1** : Execution of the process *chrome.exe* corresponding to the start of the web browser *Google Chrome*.
- **E2** and **E3** : Visit of two webpages hosted on the website `http://malwareWebsite.com`.
- **E4** : Download of the executable *contagioMalware.exe* from `http://malwareWebsite.com/MalwareDL/contagioMalware.exe` locally saved on the *Download* local folder.
- **E5** : Execution of the process *contagiomalware.exe* corresponding to the use of the malware previously downloaded.
- **E6** : Visit of the homepage of the website `http://www.bbc.co.uk`.
- **E7** : Deletion of the file *contagioMalware.exe* previously downloaded and launched.

3.3.2. *Advanced information extraction and serialisation*

The serialisation deals with seventeen attributes including temporal information (*date*, *time* and *timezone*), a description of the source of the information (*source* and *sourcetype*), a description of the event and its consequences (*MACB*, *type*, *short*, *desc* and *extra*) and information about the tools and the files used during the extraction (*version*, *filename*, *inode*, *notes* and *format*). Some fields (*date*, *time*, etc.) do not require additional processing as they are highly structured. Other fields such as the *desc* field require more treatments because their content depends on the type and the status of the event. The variability of their content makes the extraction of the knowledge difficult.

```
date,time,timezone,MACB,source,sourcetype,type,user,host,short,desc,version,filename,inode,notes,format,extra
(E1) 03/04/2015, 08:13:13, UTC,.A.., LOG, WinPrefetch, Last Time Executed, -, WIN-LPAH04KASIA, CHROME.EXE was run 122
     time(s), Prefetch [CHROME.EXE] was executed - run count 122 path: \PROGRAM FILES\GOOGLE\CHROME\APPLICATION\CHROME.EXE
     hash: 0x0548EF22 volume: 1 [serial number: 0xD4420B4A  device path: \DEVICE\HARDDISKVOLUME1], 2,
     TSK:/Windows/Prefetch/CHROME.EXE-0548EF22.pf, 44136, -, prefetch, number_of_volumes: 1  volume_device_paths:
     [u'\\DEVICE\\HARDDISKVOLUME1']  volume_serial_numbers: [3561098058L]  version: 23  prefetch_hash: 88665890
(E2) 03/04/2015, 08:13:24, UTC, .A.., WEBHIST, Chrome History, Page Visited, -, WIN-LPAH04KASIA,
     http://malwareWebsite.com/MalwareDL/index.html (Download Malware Sources and B...,
     http://malwareWebsite.com/MalwareDL/index.html (Download Malware Sources and Binaries) [count: 2] Host:
     malwareWebsite.com (typed 2 times - not indicating directly typed count), 2,
     TSK:/Users/UserX/AppData/Local/Google/Chrome/User Data/Default/History, 43419, -, sqlite, plugin: chrome_history
(E3) 03/04/2015, 08:13:27, UTC, .A.., WEBHIST, Chrome History, Page Visited, -, WIN-LPAH04KASIA,
     http://malwareWebsite.com/MalwareDL/contagio.html (Download Contagio...,
     http://malwareWebsite.com/MalwareDL/contagio.html (Download Contagio) [count: 0] Host: malwareWebsite.com Visit from:
     http://malwareWebsite.com/MalwareDL/index.html (Download Contagio) (URL not typed directly - no typed count), 2,
     TSK:/Users/UserX/AppData/Local/Google/Chrome/User Data/Default/History, 43419, -, sqlite, plugin: chrome_history
(E4) 03/04/2015, 08:13:28, UTC, ...B, WEBHIST, Chrome History, File Downloaded, -, WIN-LPAH04KASIA,
     C:\Users\UserX\Downloads\contagioMalware.exe downloaded (174054 bytes),
     http://malwareWebsite.com/MalwareDL/contagioMalware.exe (C:\Users\UserX\Downloads\contagioMalware.exe). Received:
     174054 bytes out of: 174054 bytes., 2, TSK:/Users/UserX/AppData/Local/Google/Chrome/User Data/Default/History, 43419,
     -, sqlite, plugin: chrome_history
(E5) 03/04/2015, 08:14:12, UTC, .A.., LOG, WinPrefetch, Last Time Executed, -, WIN-LPAH04KASIA, CONTAGIOMALWARE.EXE was run
     1 time(s), Prefetch [CONTAGIOMALWARE.EXE] was executed - run count 1 path: \USERS\USERX\DOWNLOADS\CONTAGIOMALWARE.EXE
     hash: 0x82B5008B volume: 1 [serial number: 0xD4420B4A  device path: \DEVICE\HARDDISKVOLUME1], 2,
     TSK:/Windows/Prefetch/CONTAGIOMALWARE.EXE-82B5008B.pf, 51001, -, prefetch, number_of_volumes: 1  volume_device_paths:
     [u'\\DEVICE\\HARDDISKVOLUME1']  volume_serial_numbers: [3561098058L]  version: 23  prefetch_hash: 2192900235
(E6) 03/04/2015, 08:14:37, UTC, .A.., WEBHIST, Chrome History, Page Visited, -, WIN-LPAH04KASIA, http://www.bbc.co.uk/news/
     (BBC News - Home), http://www.bbc.co.uk/news/ (BBC News - Home) [count: 0] Host: www.bbc.co.uk (URL not typed directly
     - no typed count), 2, TSK:/Users/UserX/AppData/Local/Google/Chrome/User Data/Default/History, 43419, -, sqlite,
     plugin: chrome_history
(E7) 03/04/2015, 08:14:47, UTC, M..., RECBIN, Recycle Bin, Content Deletion Time, -, WIN-LPAH04KASIA, Deleted file:
     C:\Users\UserX\Downloads\contagioMalware.exe, C:\Users\UserX\Downloads\contagioMalware.exe, 2,
     TSK:/$Recycle.Bin/S-1-5-21-1319092878-1000/$IQRSRGM.exe, 43673, -, recycle_bin, file_size: 174054
```

Figure 3: Data produced by log2timeline and formatted using psort

The size is the second issue of the timeline produced by Plaso. Indeed, a timeline produced using these tools from a disk image of a machine that operated about thirty minutes contains about 300,000 entries. The analysis of the produced file is almost done manually (e.g. grep, search by dates, ad hoc analysis tools such as analysis of browser search history, etc.) by investigators and the interpretation of the timeline is, therefore, tedious and complex.
With the aim at facilitating the instantiation of *ORD2I* from the data produced by Plaso, an intermediate step of information extraction and XML serialisation is used in our process. For this, the structured information such as temporal information, information about the user, information about the host and information about the type of the event are first extracted. The information contained in less structured fields, such as *desc* are then extracted. The contents of the field *desc* depends on the source of information and the type of events, therefore, a set of regular expressions is defined to properly extract the knowledge. In the **E4** case (see Figure 3), which corresponds to the download of a file using Google Chrome, a regular expression is used to extract from *desc* the URL of the downloaded file, the local path used to store it and the size of the file. The next step consists of filtering the collected data to include only the relevant data to instantiate *ORD2I* in order to reduce the amount of data, improve readability of the visualisation and optimise the processing times. After filtering, the footprints are then serialised in a XML file which is a simple and universal way to store the data.

3.3.3. ORD2I Instantiation

This step consists of populating the ontology using the result of the extraction process. For each *footprint* item of the XML file, the *CKL* and *SKL* are populated by creating instances of events, objects and subjects and links between individuals according to the formal properties defined in *ORD2I*. The relationships between an event and an object or a subject are deduced from the type of the event. For example, if a file is sent to the recycle bin, the event representing this action is linked to the file using the property *ord2i:removes*. In the case of a file download, the related event is linked to the web resource using the property *ord2i:uses* and to the local file using the property *ord2i:creates*. Figure 4 shows a Turtle[5] serialisation of knowledge related to the event **E4** of Figure 3. The individual *ord2i:FileDownload40* (lines 11-18) represents this event and is linked to the individuals *ord2i:webResource44* and *ord2i:File46*. The individual *ord2i:webResource44* (lines 31-34) belonging to *ord2i:WebResource* represents the remote resource download. *ord2i:webResource44* has a location represented by *ord2i:Location45* (lines 35-36) allowing to store the URL and the hostname of the remote resource. *ord2i:FileDownload40* is linked to *ord2i:webResource44* using the property *ord2i:uses* as the download is performed by using this remote object. *ord2i:File46* (line 38-41) belonging to *ord2i:File* is an individual representing the local file download. *ord2i:File46* has a location represented by *ord2i:Location47* (lines 42-43) allowing to store its path and

[5]http://www.w3.org/TR/turtle/

```
1  @prefix : <http://www.w3.org/2002/07/owl#> .
2  @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
3  @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
4  @prefix ord2i: <http://www.semanticweb.org/ord2i> .
5  @prefix time: <http://www.w3.org/2006/time#> .
6  @base <http://www.semanticweb.org/ord2i> .
7  <http://www.semanticweb.org/ord2i> rdf:type :Ontology ;
8  :versionIRI <http://www.semanticweb.org/ord2i/1.0.0> ;
9  :imports <http://www.w3.org/2006/time> .
10
11 <ord2i:FileDownload40> a <ord2i:Event> ;
12 <ord2i:hasEventSubtype> ”File Download” ;
13 <ord2i:hasEventType> ”Chrome History” ;
14 <ord2i:creates> <ord2i:File46> ;
15 <ord2i:uses> <ord2i:WebResource44> ;
16 <ord2i:isIdentifiedBy> <ord2i:InvestigativeOperation36> ;
17 <ord2i:hasDateTime> <ord2i:Interval42> ;  <time:hasBeginning> <ord2i:Instant41> ;
18 <ord2i:Instant41> a <time:Instant> ;  <time:inXSDDateTime> ”2015-04-03T08:13:28+02:00”^^<xsd:dateTime> .
19
20 <ord2i:InvestigativeOperation36> a <ord2i:InvestigativeOperation> ;
21 <ord2i:hasSourceName> ”File Download” ;
22 <ord2i:hasSourceType> ”Chrome History” ;
23 <ord2i:hasTechniqueName> ”Extraction using Plaso” ;
24 <ord2i:hasTechniqueType> ”Information Source” ;
25 <ord2i:isPerformedWith> <ord2i:Tool0> ; <ord2i:Tool0> a <ord2i:Tool> ;
26 <ord2i:hasName> ”Plaso” ; <ord2i:hasVersion> ”1.2.0” .
27 <ord2i:hasTruthfulness> ”100.0”^^<xsd:double> .
28 <ord2i:hasDateTime> <ord2i:Interval38> ;  <time:hasBeginning> <ord2i:Instant37> ;
29 <ord2i:Instant37> a <time:Instant> ;  <time:inXSDDateTime> ”2015-03-04T15:50:34+01:00”^^<xsd:dateTime> .
30
31 <ord2i:WebResource44> a <ord2i:WebResource> ;
32 <ord2i:hasLocation> <ord2i:Location45> ;
33 <ord2i:isIdentifiedBy> <ord2i:InvestigativeOperation36> ;
34 <ord2i:hasSize> ”174054”^^<xsd:integer> .
35 <ord2i:Location45> a <ord2i:RemoteVirtualLocation> ;  <ord2i:hasHostname> ”malwarewebsite.com” ;
36 <ord2i:hasURL> ”http://malwarewebsite.com/malwaredl/contagiomalware.exe” .
37
38 <ord2i:File46> a <ord2i:File> ;
39 <ord2i:hasLocation> <ord2i:Location47> ;
40 <ord2i:isIdentifiedBy> <ord2i:InvestigativeOperation36>;
41 <ord2i:hasSize> ”174054”^^<xsd:integer> .
42 <ord2i:Location47> a <ord2i:LocalVirtualLocation> ; <ord2i:hasFilename> ”contagiomalware.exe” ;
43 <ord2i:hasPath> ”c:\\users\\userx\\downloads\\” .
44
45 <ord2i:Process21> a <ord2i:Process> ;
46 <ord2i:hasName> ”Google Chrome” .
47 <ord2i:isInvolved> <ord2i:FileDownload40>;
48 <ord2i:isIdentifiedBy> <ord2i:InvestigativeOperation36>;
49
50 <ord2i:Person22> a <ord2i:Person> ;
51 <ord2i:hasPseudo> ”UserX” ;
52 <ord2i:isIdentifiedBy> <ord2i:InvestigativeOperation36>;
53 <ord2i:isInvolved> <ord2i:FileDownload40>;
```

Figure 4: Turtle serialisation of knowledge generated by the instantiation of *ORD2I*

its filename. *ord2i:FileDownload40* is linked to *ord2i:File46* using the property *ord2i:creates*. Regarding subjects, Google Chrome, used to carry out the download, is represented by the individual *ord2i:Process21* (line 45-48) and it is linked to the event using the property *ord2i:isInvolved*. The individual *ord2i:Person22* (lines 50-53) represents the user who starts the download (linked to the event using the property *ord2i:isInvolved*).
The knowledge describing how previous knowledge were identified is then added in the *TKL* of the ontology. *ord2i:investigativeOperation1* (line 20-29) represents the task of information extraction performed by the tool Plaso (line 25-26). As property *ord2i:isIdentifyBy* states, this task allows to identify several entities including *ord2i:FileDownload40*, *ord2i:WebResource44*, *ord2i:File46*, *ord2i:Process21* and *ord2i:Person22*. To conclude, the event *ord2i:FileDownload40* and the task ord2i:InvestigativeOperation36 are localised in time using the ontology OWL-Time (lines 17-18 and lines 28-29).

3.4. Analysis

After the instantiation of the ontology, a large knowledge graph representing the information contained in the output of Plaso is obtained. The structure of this graph is derived from the schema of the *ORD2I* ontology. This graph has many advantages: it structures information and it facilitates the interpretation of the information and the development of automatic analysis processes. As mentioned in Section 2, the analysis of this large volume of knowledge requires the use of some very sophisticated tools. In this paper, we present the inference tools and automated analysis processes proposed in our approach. It is important to note that the proposed operators are based on assumptions made by the authors that can be subject to discussion. However, the main objective is to demonstrate the relevance of an ontology as this greatly facilitates the development of the analysis tools.

3.4.1. Knowledge enhancement

The goal of the enhancement step is to enrich the knowledge base with new deduced facts to complete the investigators' knowledge on the incident. The inference of new facts increases the effectiveness of the analysis and the quality of final results. Indeed, adding new knowledge about the incident during the enhancement stage improves the quality of the conclusions.
Inference is an operation taking as input one or more established facts and outputting a new fact, called conclusion. In our work, we have implemented several inference rules allowing for example to infer the type of a file based on its extension (e.g. an individual belonging to the *ord2i:File* class which has the extension *.exe* is automatically classified in the *ord2i:ExeFile* class) or to deduce the OS user session associated to a given event when this information is unknown.
This second example of inference is detailed below. Any information on the user session associated to an event is valuable for an investigation. It is therefore interesting to provide a tool to identify an association between a given user and an event. Some kind of events provide information about the users involved in them. This is the case of *Windows EVTX* entries for which the user field is filled in Plaso. This is also the case of entries generated by web browsers for which the user name is contained in the path of history files that are used to generate the entry (i.e. in the case of Google Chrome, every history file is located in `TSK:/Users/UserX/AppData/Local/Google/Chrome/UserData/Default/History`). To deduce the user associated to an event, we define two inference rules which are based on the temporal location of events which is defined using Allen algebra (Allen, 1983) and the notion of time interval. Our ontology represents the beginning and the end of each event to model its duration (i.e. downloading a file that lasts for several minutes). However, because of the granularity of the timestamps and the execution speed, most of the events are considered instantaneous. In addition, the end of the event is not often available in the output of Plaso. In most cases, therefore, we consider that each event is localised in time by an interval that has a start time and an end time equal to the datetime provided by Plaso.

The two inference rules used to identify users involvement are:

- Rule №1: Given an event A for which the user is unknown and an event B associated to the user P, if A and B are temporally linked by a property $p \in equal, overlaps, during, starts, meets, finishes$, then the user associated with A is P.

- Rule №2: Given two events A and B associated to the user P and located at both extremities of a chain of events exclusively composed of events for which the user is unknown, then each event composing this chain is associated with the user P.

Event	Time	User
ApplicationPrefetch5 (**E1**)	2015-04-03T08:13:13	
WebpageVisit17 (**E2**)	2015-04-03T08:13:24	*Person22*
WebpageVisit30 (**E3**)	2015-04-03T08:13:27	*Person22*
FileDownload40 (**E4**)	2015-04-03T08:13:28	*Person22*
ApplicationPrefetch52 (**E5**)	2015-04-03T08:14:12	***Person22***
WebpageVisit62 (**E6**)	2015-04-03T08:14:37	*Person22*
FileDeletion72 (**E7**)	2015-04-03T08:14:47	

Table 1: Identification of users involvement using knowledge enhancement

These two rules are illustrated in Figure 5 (users in bold italic are the results of the enhancement). The upper part of the Figure illustrates the rule №1. This rule is used to deduce the user that is associated with the events e2, e4, e6, e8, e10, e12. The bottom part of the Figure illustrates the rule №2 and shows that the UserX is involved in e17 and e18.

After the instantiation of the ontology by using as input the entries given in Figure 3, the knowledge we have about user involved in each event can be retrieved using a SPARQL query. The column *user* in Table 1 shows the relation between user and event. The users with normal font are known directly after the instantiation of the ontology (as the information is available in the output of Plaso) while the users appearing in bold italic are deduced during the enhancement phase. After the enhancement phase and a new execution of the SPARQL query, we see that a link between *ord2i:ApplicationPrefetch22* and *ord2i:Person22* has been identified by the rule №2. As the user *ord2i:Person22* is involved in *ord2i:FileDownload40* and *ord2i:WebpageVisit62* and as *ord2i:ApplicationPrefetch52* is located between these two events, the system states that *ord2i:Person22* is also involved in this event. It should be noted that the inference of new associations between users and events cannot be made with certainty. Indeed, another user can take the place of the current user or the events can be initiated by a person or a process beyond the control of the user who opened the session. Thus, the score of confidence associated to the operation (belonging to *ord2i:InvestigativeOperation*) that has deducted the association is low (to invite the investigators to use this information cautiously).

3.4.2. Timeline Analysis

The first analysis tool proposed in our approach is a process (based on (Chabot et al., 2014)) detecting correlation between a pair of events. The correlation is a relationship with a broad semantic that covers causal relationships and other semantic links. The identification of such relationships is particularly relevant for the investigators as it allows them to quickly have an overall view of the events composing an incident and the links between them. The relevance of this tool will be shown in Section 4. The identification of correlated pairs of events is performed using four criteria: the interaction of the two events with 1) common objects or 2) subjects, 3) the temporal proximity and 4) a set of

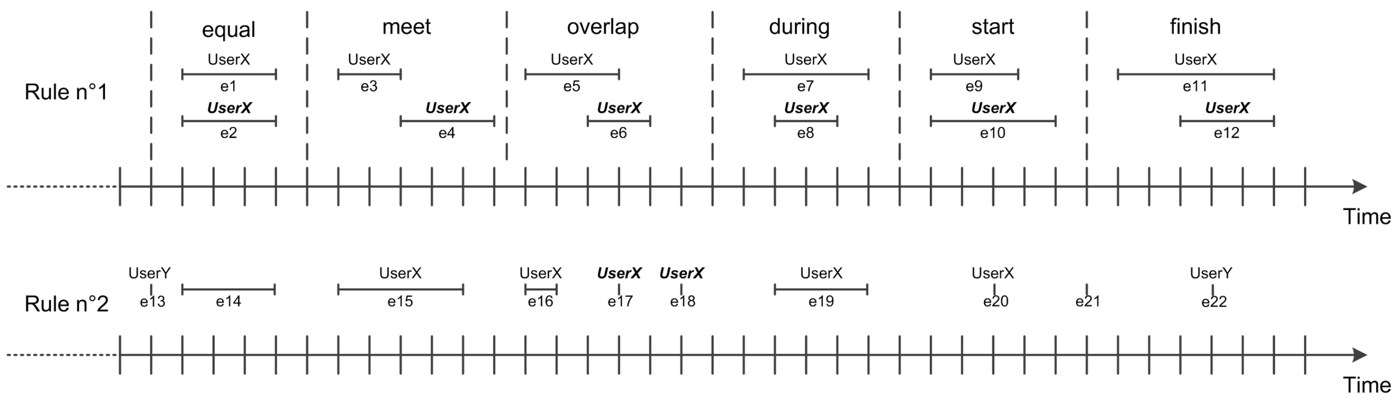

Figure 5: Patterns used to deduce new relationships between users and events

rules that are satisfied or not. For each pair of events(e_1, e_2), a score of correlation is calculated using the following formula:

$$Corr(e_1, e_2) = Corr_T(e_1, e_2) + Corr_S(e_1, e_2) + Corr_O(e_1, e_2) \quad (1)$$

where each score is between 0.0 and 1.0 after normalisation and where

- $Corr_T(e_1, e_2)$ is a score quantifying the temporal correlation of the two events. The hypothesis used in this criterion is that if two events occur at the same time, the temporal correlation is equal to 1.0. Else, the more the two events are closed in time, the more they are correlated.
- $Corr_S(e_1, e_2)$ is a score quantifying the correlation in the light of subjects interacting with the two events. The more the two events interact with common subjects (process or person), the greater the subject correlation is. Therefore, the subject correlation is maximal for two events interacting with only two subjects, the Google Chrome process and the same user for example. In the case of two events interacting with the Google Chrome process but two different users, the score is equal to 0.5.
- $Corr_O(e_1, e_2)$ is a score quantifying the correlation in the light of objects interacting with the two events. The more the two events interact with common objects, the greater the object correlation is. To obtain a finer system, we choose to not only consider common objects but also objects whose location is near. In the case of two events interacting with two objects sharing the same hostname (in case of remote resources) or the same path (in case of local resources), the score is increased.

Each criterion can be weighted to allow to give more importance to one of the correlation scores (in this study, all scores are equivalently weighted). The overall correlation score is between 0.0 (meaning that the two events are not correlated) and 1.0 (meaning that the two events are strongly correlated), which is the average of the three scores. The first three criteria are not dependent of the type of events as they are based on generic features (time, object and subjects). This will allow to discover correlations involving events from unknown sources of information that are therefore not intended by the investigators. To take into account specific knowledge of every type of object model in *ORD2I*, we introduce a fourth score: $Corr_{EK}(e_1, e_2)$. This score is computed using a set of rules defined by experts (called expert knowledge-based rules). If two events satisfy a certain given rule, the overall correlation score between them is maximal. For example, in our experiments, we add the following two rules: The first rule produces a high correlation score in the case of two events; one representing the download of an executable file and the second representing its execution. The second rule allows to get a high correlation score in the case of an event representing the creation of a bookmark for a webpage and an event representing a visit of this same webpage using the bookmark.

To illustrate the correlation process, we use our approach on the dataset composed of seven events shown in Listing 3. The story of this dataset appears to be the following. First, the user started the web browser Chrome (*ord2i:ApplicationPrefetch5*). Then, he visited two webpages (*ord2i:WebpageVisit17* and *ord2i:WebpageVisit30*) on a website that seems to provide sources and binaries of malwares. On the second webpage, he found a link allowing him to download an executable file named *contagiomalware.exe* (*ord2i:FileDownload40*). Once the download was complete, the user ran the executable (*ord2i:ApplicationPrefetch52*). He continued to browse the web on a non related website (*ord2i:WebpageVisit62*). Finally, the user deleted the file *contagiomalware.exe* (*ord2i:FileDeletion72*). Among these events, it is possible to identify a logical chain of events using intuition. This chain is composed of the events *ord2i:WebpageVisit17*, *ord2i:WebpageVisit30*, *ord2i:FileDownload40*, *ord2i:ApplicationPrefetch52* and *ord2i:FileDeletion72*. These events make up a coherent story in which the user gets a malware provided by a website, executes it and attempts to erase the traces by deleting the file. The main objective of the proposed correlation tool is to highlight this type of chain of events among a large number of events composing a case. Thus, on one hand one can understand the case quickly and on the other hand see the whole chains of events. The third quartile of correlation score is given in Table 2. This table contains nine pairs of correlated events for which five scores are given: the temporal correlation, the subject correlation, the object correlation, the correlation based on rules and finally the overall correlation. The scores show four pairs of highly correlated events (*overallscore* > 0.5):

- *ord2i:ApplicationPrefetch52* and *ord2i:FileDownload40*: these two events satisfy the rule defined below (”download of an executable file and execution of the same file”).
- *ord2i:FileDownload40* and *ord2i:WebpageVisit30*: the subject correlation between these events is maximal as they are both related to only two subjects, Google Chrome and the user *ord2i:Person22*. In addition, as only one second has elapsed between the visit of the webpage containing the download link of *contagiomalware.exe* and the start of the download, the temporal correlation is high too. Finally, as the URL of the remote resource used for the download and the URL of the visited webpage are hosted by the same website, the object correlation is modified accordingly.
- *ord2i:WebpageVisit17* and *ord2i:WebpageVisit30*: the object and subject correlation scores can be explained in the same way as above.
- *ord2i:FileDownload40* and *ord2i:WebpageVisit17*: the same explanation can be used for this case too.

The correlation score in the fifth position (*ord2i:ApplicationPrefetch52* and *ord2i:FileDeletion72*) can be explained by the interaction with the same object, one event downloading an object and the other event deleting it. The same explanation can be applied to the last correlation score (*ord2i:FileDeletion72* and *ord2i:FileDownload40*). However, for these two correlation scores, the subject correlation score and the temporal correlation score are low as they do not interact with common subjects and they are not close in time (compared to other events). Finally, the three correlations involving *ord2i:WebpageVisit62* are exclusively due to the interaction with common subjects (Google Chrome and *ord2i:Person22*).

evt1	evt2	T	S	O	EK	Ov.
App.Prefetch52 (**E5**)	*FileDownload40* (**E4**)	0.01	0.5	1.0	1.0	1.0
FileDownload40 (**E4**)	*WebpageVisit30* (**E3**)	1.0	1.0	0.25	0.0	1.0
WebpageVisit17 (**E2**)	*WebpageVisit30* (**E3**)	0.33	1.0	0.25	0.0	0.7
FileDownload40 (**E4**)	*WebpageVisit17* (**E2**)	0.24	1.0	0.25	0.0	0.66
App.Prefetch52 (**E5**)	*FileDeletion72* (**E7**)	0.02	0.0	1.0	0.0	0.45
FileDownload40 (**E4**)	*WebpageVisit62* (**E6**)	0.0	1.0	0.0	0.0	0.45
WebpageVisit30 (**E3**)	*WebpageVisit62* (**E6**)	0.0	1.0	0.0	0.0	0.45
WebpageVisit17 (**E2**)	*WebpageVisit62* (**E6**)	0.0	1.0	0.0	0.0	0.45
FileDeletion72 (**E7**)	*FileDownload40* (**E4**)	0.0	0.0	1.0	0.0	0.45

Table 2: Third quartile of correlation scores for the events from Figure 3

3.5. Interface: Visualisation and SPARQL querying

The *Interface Layer* has several functions:

- It allows investigators to visualise the data via intuitive and clear visualisation tool.
- It provides a monitoring interface allowing to manage the settings of the system including the numerical values used in the correlation process (weight, size of the sliding window, etc.) and the information sources at the input of the instantiation step.
- It provides an advanced query tool for users that existing visualisation tools are not able to perform.

A tool has been implemented to visualise the correlations between the events. This visualisation takes the form of a graph containing all the events composing the incident and the correlations between them. The use of this graph allows to quickly have an overall view of the case. The links between events show chains of related events. The visual style (color, line type, thickness) of each link depends on the strength of the correlation. The investigator can interact with the graph by clicking on the elements. A click on a link gives information about the correlation such as the overall correlation score, the temporal correlation score, the subject correlation score, the object correlation score and the correlation score based on rules. A click on a node gives information about the event (URI, description, objects and subjects interacting with the event). The collection of this information from the triple store is done with the dynamic creation of SPARQL queries.
An overview of the correlation graph corresponding to the dataset of Figure 3 is given in Figure 6. There appears a chain

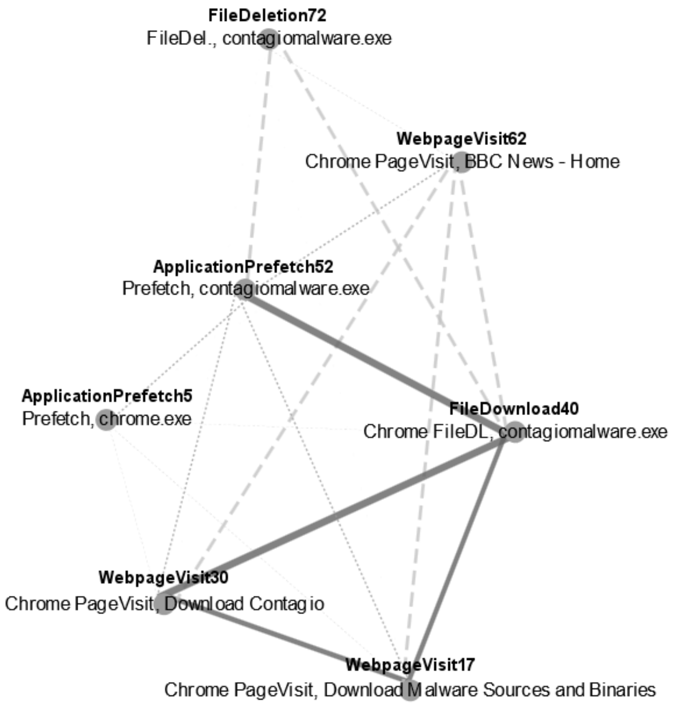
FileDeletion72
FileDel., contagiomalware.exe
WebpageVisit62
Chrome PageVisit, BBC News - Home
ApplicationPrefetch52
Prefetch, contagiomalware.exe
ApplicationPrefetch5
Prefetch, chrome.exe
FileDownload40
Chrome FileDL, contagiomalware.exe
WebpageVisit30
Chrome PageVisit, Download Contagio
WebpageVisit17
Chrome PageVisit, Download Malware Sources and Binaries

Figure 6: Correlation graph of events given in Figure 3 (solid edges (*score* >= 0.6), dashed edges (*score* >= 0.4 && *score* < 0.6), dotted edges (*score* < 0.4))

of strongly correlated events including the visit of the webpage, the download of the resource, the launch of this resource and its removal. This graph allows to identify the chain of events composed of *ord2i:WebpageVisit17*, *ord2i:WebpageVisit30*, *ord2i:FileDownload40*, *ord2i:ApplicationPrefetch52* and *ord2i:FileDeletion72* discussed above.

To go further and display information which does not appear in the visualisation tool, one can use the SPARQL endpoint to query directly the knowledge contained in the triple store (the knowledge base). For example, a query retrieving information about all webpages visited by the user named UserX is given in Figure 7. The results of this query are given in Table 3. Other examples of the use of SPARQL are given in Section 4. Despite its power, the SPARQL language (as SQL) is not an intuitive and simple way to access knowledge and it requires technical knowledge to be used efficiently. Therefore, we have the desire to minimise the use of SPARQL by integrating a large amount of data in the visualisation tools.

```
prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
prefix owl: <http://www.w3.org/2002/07/owl#>
prefix xsd: <http://www.w3.org/2001/XMLSchema#>
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix time: <http://www.w3.org/2006/time#>
prefix ord2i: <http://checksem.u-bourgogne.fr/ord2i#>
SELECT ?titleWebpage ?urlWebpage ?hostWebpage
WHERE
{
  ?event a ord2i:Event.
  ?event ord2i:hasEventSubtype "Webpage Visit".
  ?event ord2i:uses ?object.
  ?object ord2i:hasTitle ?titleWebpage.
  ?object ord2i:hasLocation ?location.
  ?location ord2i:hasURL ?urlWebpage.
  ?location ord2i:hasHostname ?hostWebpage.
  ?user ord2i:isInvolved ?event.
  ?user ord2i:hasPseudo ?pseudo.
  FILTER(?pseudo = "UserX").
}
```

Figure 7: SPARQL query retrieving information about all webpages visited by the user named UserX

titleWebpage	hostWebpage
Download Malware Sources and Binaries	malwarewebsite.com
Download Contagio	malwarewebsite.com
BBC News - Home	www.bbc.co.uk

Table 3: Results of the query retrieving information about all webpages visited by the user named UserX

This section has presented the *SADFC* approach. It has demonstrated the technical feasibility of such an approach and the relevance of its features. Among them, the analysis and visualisation tools based on the *ORD2I* ontology appear to be promising to investigate efficiently on a forensic case. These features help to save time by identifying and displaying on the screen the strength of the correlations between events. This will help to quickly understand the physiognomy of the incident.

4. Experimentation and results

This section aims to demonstrate the capabilities of our approach in the context of a digital investigation analysis. For this, we propose a study of the performance and relevance of the approach through simulated cases. The configuration of the machine used to run the experiment and hosting the triple store (Stardog server 2.2.4) has a 3.20GHz Intel Core i5-3470 processor and 8GB RAM. Disk images used are generated from a virtual machine running Windows 7.

The first task of the experiment is to evaluate the execution time of our tool and the volume of data handled and generated through the process. For comparison, we give the results for two others datasets of different sizes. Table 4 provides information on the volumes with the number of entries composing the timeline, the number of entries supported by our tool (which do not support all sources of information), the number of entries after filtering, the number of triples generated during the instantiation phase, the number of triples deducted during the enhancement phase and finally the number of significant correlations found (*score* > 0.5, this threshold can be modified by the user via the interface). Table 5 gives the execution times (in seconds) for the different phases of the process.

Criterion \Datasets	Malware case	Dataset №2	Dataset №3
Number of lines (timeline)	8664	5241	21424
Supported entries (extraction)	7918	4600	17863
Entries after filtering	222	130	774
Generated triples (instantiation)	10186	12400	33498
Deduced triples (enhancement)	21	17	28
Correlations (>= 0.5)	1411	485	15356

Table 4: Quantification of processed data volumes through the reconstruction and analysis process

Steps \Datasets	Malware case	Dataset №2	Dataset №3
Extraction	1.17	1.135	1.85
Instantiation	26.81	19.9	186.295
Enhancement	0.31	0.231	0.353
Analysis	814.2	254.3	13541.8

Table 5: Execution times of the different stages of the process

Note that we reduce the number of entries by using some filtering mechanism during the instantiation phase. These filters are to used only the entries of certain sources of information. Thus, entries from Chrome cookies, Chrome cache, Internet Explorer cache and the file system are filtered. From these results, we can see that the extraction, instantiation and enhancement phases are carried out quickly. On the other hand, the analysis phase is a time-consuming step which emphasises the importance of filtering the sources of non-critical information to reduce the amount of data to analyse. It is also important to see that the number of triples does not increase linearly with the number of entries in the timeline. Indeed, the number of triples generated for an event depends on its type. Some types of events carry more information and therefore require a larger

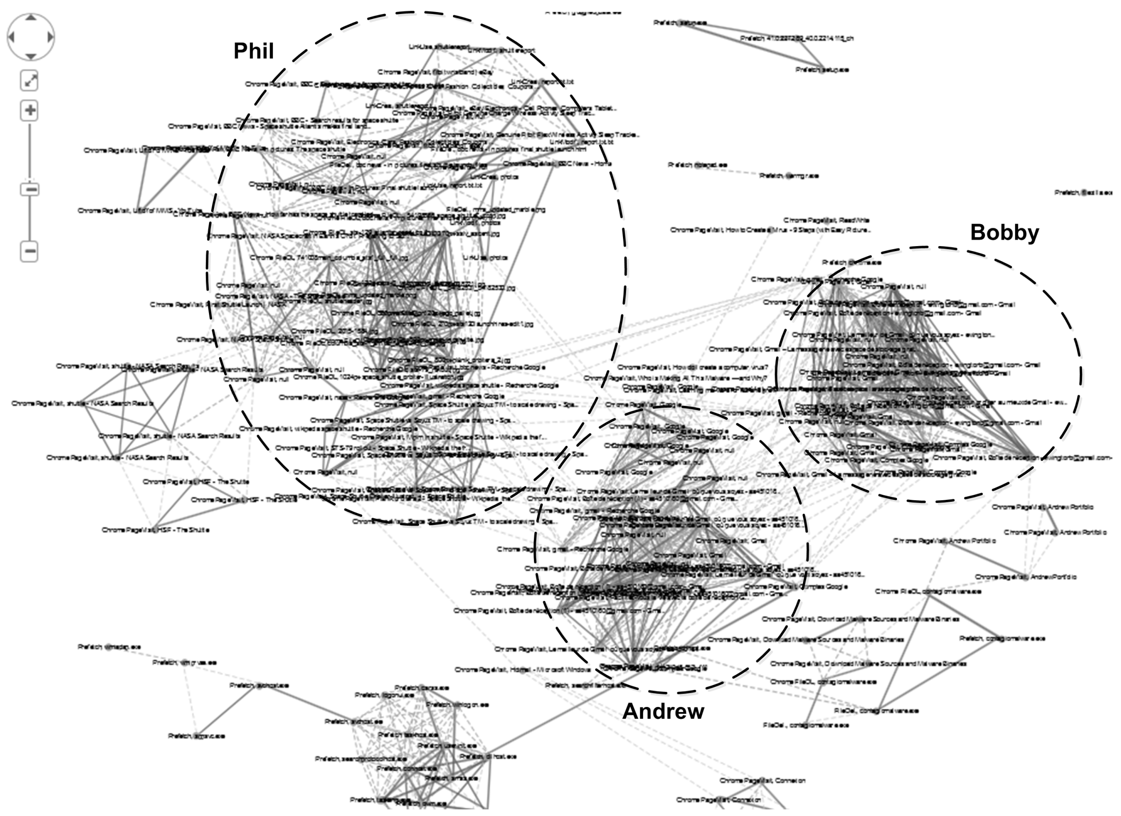

Figure 10: Correlation graph viewer: wide zoom on the events composing the case

number of triples to be modelled in the ontology.

The second task of the experiment illustrates the capabilities and the relevance of our approach. The investigation started after that Phil, an employee of CompanyAndCo, has reported to his IT manager that his machine generates abnormal behaviours. To understand what has happened, the investigator in charge started by collecting the hard disk of the suspect machine, and applied the *SADFC* approach to study the disk image. After successfully completing the extraction phase, the instantiation phase and the enhancement phase, the investigator starts the analysis by searching all significant correlations between events (*score* > 0.5). From the testimony of Phil on the behaviour of the machine, the investigator assumes that a malware is the cause of the problem. Therefore, he decided to search all traces of potentially suspicious executables that were run on the machine. To do this, he used the SPARQL query shown in Figure 8 to retrieve all executables that can be identified in the timeline. Among the results of the query given in Table 6, he identified two entries with a suspicious name (*contagiomalware.exe*) located in `c:\users\andrew\downloads\` and `c:\users\bobby\downloads\`. The investigator then tried to figured out who interacts with this malware and in which circumstances. This information can be obtained using the correlation graph viewer. Indeed, the viewer allows to manually retrieve the events interacting with the executables and then get information about the events. The information can also be retrieved using the SPARQL query given in Figure 9.

```
prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
prefix owl: <http://www.w3.org/2002/07/owl#>
prefix xsd: <http://www.w3.org/2001/XMLSchema#>
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix ord2i: <http://checksem.u-bourgogne.fr/ord2i#>
SELECT ?path ?filename
WHERE
{
  ?exe a ord2i:WinExeFile.
  ?exe ord2i:hasLocation ?location.
  ?location ord2i:hasFilename ?filename.
  ?location ord2i:hasPath ?path
}
```

Figure 8: SPARQL query used to retrieve all the executables

This query allows to retrieve each event interacting with the executable named *contagiomalware.exe*. For each of these events, the query retrieves the name of the user involved, the type and subtype of the event, the date and time it occurred. The result of this query is shown in Table 7. The investigator noted that the users Andrew and Bobby have both used *contagiomalware.exe*. Andrew has downloaded it and then removed it. Bobby, on his side, has downloaded the malware, launched it and finally deleted it. The investigator then tried to find out if both Andrew and Bobby are related and if so, why? To answer these questions, he uses the correlation graph viewer. The consultation of a wide view of the graph allows to distinguish different clusters of events (Figure 10). In particular, there are three clusters corresponding to the

path	filename
`c:\windows\temp\cr_e7cf0.tmp\`	setup.exe
`c:\windows\system32\`	sppsvc.exe
`c:\windows\system32\wbem\`	wmiprvse.exe
`c:\windows\system32\wbem\`	wmiadap.exe
`c:\users\andrew\downloads\`	contagiomalware.exe
`c:\programfiles\google\update\`	googleupdate.exe
`c:\windows\system32\`	wermgr.exe
`c:\windows\system32\`	notepad.exe
`c:\windows\system32\`	smss.exe
`c:\windows\system32\`	csrss.exe
`c:\windows\system32\`	winlogon.exe
`c:\windows\system32\`	taskhost.exe
`c:\windows\system32\`	dllhost.exe
`c:\windows\system32\`	userinit.exe
`c:\windows\system32\`	dwm.exe
`c:\windows\system32\`	taskeng.exe
`c:\windows\`	explorer.exe
`c:\programfiles\vmware\vmwaretools\`	tpautoconnect.exe
`c:\windows\system32\`	conhost.exe
`c:\programfiles\vmware\vmwaretools\`	vmtoolsd.exe
`c:\windows\system32\`	searchprotocolhost.exe
`c:\windows\system32\`	searchfilterhost.exe
`c:\users\bobby\downloads\`	contagiomalware.exe
`c:\programfiles\google\chrome\application\`	chrome.exe
`c:\windows\system32\`	logonui.exe

Table 6: Results of the query retrieving all the executable files

pseudo	type	subtype	datetime
Andrew	Chrome History	File Download	2016-01-03T07:21:16.000+01:00
Bobby	Chrome History	File Download	2016-01-03T08:12:09.000+01:00
Bobby	Recycle Bin	File Deletion	2016-01-03T08:12:43.000+01:00
Bobby	Windows Prefetch	App. Prefetch	2016-01-03T08:12:25.000+01:00

Table 7: Results of the query retrieving information about events related to *contagiomalware.exe*

events belonging to each user. A quick study of the cluster related to Phil shows that he browsed the web and consulted his emails. The most relevant information for the investigation is located on the clusters related to Andrew and Bobby. The existence of many links between these two clusters indicates that these two users have strong interactions. The investigator then zooms on the small group of events between the two clusters; looking for information about the use of the executable *contagiomalware.exe* by one of the two users. After having found one of the events, he can view the closest neighbourhood of the event using correlation links shown in Figure 11. The chain of correlated events found represents the life cycle of the malware on the computer. By clicking on the nodes, he can consult the information about the events (date and time, related objects and subjects, description). To get more details about about an event, he can also use the SPARQL interface. The use of the correlation graph viewer and the SPARQL interface, two complementary tools, allows him to identify the following information. Regarding Andrew, we can see that he has viewed a website entitled "Download Malware Sources and Malware Binaries" at `http://malwarewebsite.com`. He has then downloaded the file *contagiomalware.exe* from this website. He then went on his personal website named "Andrew Portfolio". By clicking on the corresponding node, we obtain the URL of Andrew's website `http://andrew-and-co.com`. Regarding Bobby, we can see that he visited the website named "Andrew Portfolio". He consulted on this website a folder called "private" from which he downloaded the file *contagiomalware.exe*. The graph also confirms that Bobby launched this executable and then deleted it.

```
prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
prefix owl: <http://www.w3.org/2002/07/owl#>
prefix xsd: <http://www.w3.org/2001/XMLSchema#>
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix time: <http://www.w3.org/2006/time#>
prefix ord2i: <http://checksem.u-bourgogne.fr/ord2i#>
SELECT ?pseudo ?type ?subtype ?datetime ?desc
WHERE
{
  ?user a ord2i:Person.
  ?user ord2i:hasPseudo ?pseudo.
  ?user ord2i:isInvolved ?event.
  ?event ?property ?exe.
  ?event ord2i:hasEventType ?type.
  ?event ord2i:hasEventSubtype ?subtype.
  ?event ord2i:hasDateTime ?date.
  ?event ord2i:hasDescription ?desc.
  ?date time:hasBeginning ?begin.
  ?begin time:inXSDDateTime ?datetime.
  ?property rdfs:subPropertyOf* ord2i:interacts .
  ?exe a ord2i:WinExeFile.
  ?exe ord2i:hasLocation ?location.
  ?location ord2i:hasFilename ?filename.
  FILTER(?filename="contagiomalware.exe")
}
```

Figure 9: SPARQL query retrieving information about events interacting with *contagiomalware.exe*

In conclusion, we have seen that the proposed approach allows to quickly extract the knowledge related to a digital incident. This knowledge can then be enriched using the enhancement process which allows to deduce new knowledge from the existing one. After the analysis phase, the use of the correlation graph viewer and the SPARQL interface allow to understand the nature of the interactions between the two users. It appears clearly that Andrew and Bobby have strong interactions. In particular, we have seen that they both used the file *contagiomalware.exe* and the website `http://andrew-and-co.com`. Current tools of *SADFC* do not allow to determine with certainty the nature of this interaction. However, the investigator may assume that Andrew actually downloaded the malware from the website `http://malwarewebsite.com` and he then made it available on his personal webiste `http://andrew-and-co.com`. Bobby has then downloaded and launched the executable. The effects on the machine remain unknown at this stage of the investigation. Therefore, we find that the proposed tool only provides good research directions but the investigators analytical faculties are still required to complete the investigation.

5. Conclusion and future works

In this paper, we proposed an approach called *SADFC* for Semantic Analysis of Digital Forensic Cases. This approach is based on an ontology to represent accurately a digital incident and the associated digital investigation. The ontology, named *ORD2I*, is associated with a set of tools for extracting information from disk images seized on crime scenes, instantiating the

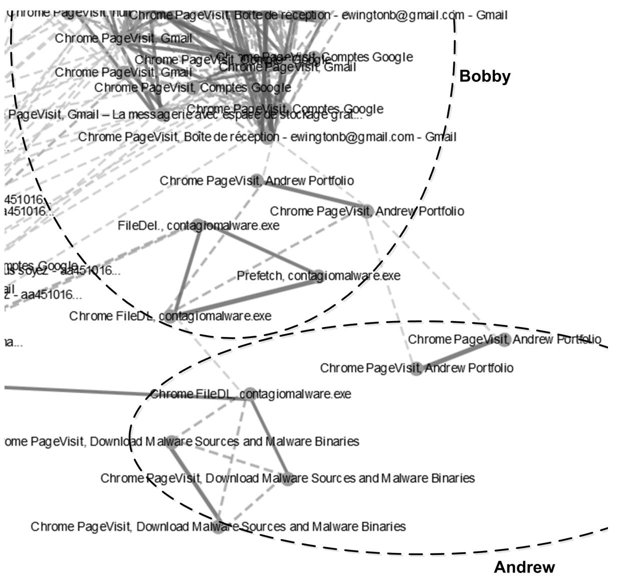
Chrome PageVisit, Boîte de réception - ewingtonb@gmail.com - Gmail
Chrome PageVisit, Gmail
Chrome PageVisit, Comptes Google
Bobby
Chrome PageVisit, Boîte de réception - ewingtonb@gmail.com - Gmail
Chrome PageVisit, Andrew Portfolio
FileDel., contagiomalware.exe
Prefetch, contagiomalware.exe
Chrome FileDL, contagiomalware.exe
Chrome PageVisit, Andrew Portfolio
Chrome FileDL, contagiomalware.exe
Chrome PageVisit, Download Malware Sources and Malware Binaries
Andrew

Figure 11: Correlation graph viewer: close view on the chain of events related to *contagiomalware.exe*

ontology, deducing new knowledge and analysing it. *SADFC* provides answers to the three issues identified in the state of the art: the cognitive overload due to the large volumes of data to be processed, the heterogeneity of the data and the legal requirements that every evidence has to meet in order to be admissible in court. Regarding the data volumes, our tools are automatic which enables efficient data processing. In addition, we develop enhancement and analysis tools to perform reasoning tasks allowing the investigators to focus on the most complex parts of the investigation where their experience, expertise and skills are the most needed. Visualisation tools have been also introduced to provide an intuitive and clear way to access the knowledge.
Regarding heterogeneity of data, *SADFC* is based on *ORD2I* which is composed of three layers for representing knowledge common to all events, specialised information and knowledge about the investigation. The use of an ontology allows to have a unified model to represent knowledge, allowing to easily build analysis processes. Finally, the proposed ontology allows to meet the legal requirements and especially the need for reproducibility and traceability (Chabot et al., 2014).

Despite the mentioned qualities, *SADFC* has several limitations that will be studied in future works. The performance of the tool can still be improved, especially the execution time of the analysis phase. The benchmarks show that it takes approximatively three hours to handle a timeline with 20000 entries which is a small timeline compared to the timeline handled in real cases. The optimisation of the analysis is, therefore, a priority in future versions of the tool.
Regarding the extraction and the instantiation of the ontology, the tool can presently handle fifteen sources of information which is still not enough to have a complete view of an incident. In future work, new sources will therefore be integrated. In particular, information sources related to Android and iOS must be integrated to handle mobile devices. At the same time, the ontology *ORD2I* must be validated by experts.
For the analysis, we plan to integrate new tools besides the event correlation algorithm. 1) We will develop a pattern matching algorithm to detect illegal actions by identifying specific event sequences (an illegal action can be composed of several authorised events, hence the need to use a pattern-based system to correctly detect illicit actions). 2) Based on the work of (Hargreaves & Patterson, 2012), we will implement a tool to find event composition in order to allow the summarisation of the timeline.
Concerning the interface, several visualisation tools will be implemented in future works. First, more features will be integrated in the correlation graph viewer including search functions. Second, an advanced timeline viewer will be proposed. This viewer will allow on the one hand to display conventionally the timeline and in the other hand to enrich this timeline with knowledge from the ontology including information about objects and subjects interacting with each event and information produced by the enhancement phase and the analysis phase. We also plan to allow the investigators to enrich the knowledge of the ontology with information they found on their own on the digital crime scene in order to improve interactivity and collaboration between *SADFC* and the investigators. The implementation of such functionality requires mechanisms to check the consistency of the ontology and to update it frequently. Indeed, adding new information may conflict with the existing knowledge about the incident or enable new deductions. Furthermore, if several investigators work on the same case, we must provide collaborative tools to manage the points of view of all members of the team.

6. Acknowledgements

The above work is a part of a collaborative research project between the CheckSem team (Le2i UMR CNRS 6306 University of Burgundy) and the UCD School of Computer Science and Informatics. This project is supported by a grant from University College Dublin and the Burgundy region (France). The authors would like to thank Séverine Fock for the advices and the proofreading of this manuscript.

References

Abbott, J., Bell, J., Clark, A., De Vel, O., & Mohay, G. (2006). Automated recognition of event scenarios for digital forensics. In *Proceedings of the 2006 ACM symposium on Applied computing* (pp. 293–300). ACM.

Allen, J. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM*, *26*, 832–843.

Barnum, S. (2011). Cyber observable expression (cybox) use cases.

Baryamureeba, V., & Tushabe, F. (2004). The enhanced digital investigation process model. In *Proceedings of the Fourth Digital Forensic Research Workshop*. Citeseer.

Buchholz, F., & Falk, C. (2005). Design and implementation of zeitline: a forensic timeline editor. In *Digital forensic research workshop*.

Case, A., Cristina, A., Marziale, L., Richard, G. G., & Roussev, V. (2008). Face: Automated digital evidence discovery and correlation. *digital investigation*, *5*, S65–S75.

Casey, E., Back, G., & Barnum, S. (2015). Leveraging cybox to standardize representation and exchange of digital forensic information. *Digital Investigation*, *12, Supplement 1*, S102 – S110. {DFRWS} 2015 Europe Proceedings of the Second Annual {DFRWS} Europe.

Chabot, Y., Bertaux, A., Nicolle, C., & Kechadi, T. (2014). A complete formalized knowledge representation model for advanced digital forensics timeline analysis. *Digital Investigation (Fourteenth Annual DFRWS Conference)*, *11*, S95–S105.

Chen, K., Clark, A., De Vel, O., & Mohay, G. (2003). Ecf-event correlation for forensics. In *First Australian Computer Network and Information Forensics Conference* (pp. 1–10). Perth, Australia: Edith Cowan University.

Farmer, D., & Venema, W. (2004). The coroner's toolkit (tct).

Gil, Y., & Miles, S. (2013). Prov model primer. *W3C Working Group Note*, .

Gladyshev, P., & Patel, A. (2004). Finite state machine approach to digital event reconstruction. *Digital Investigation*, *1*, 130–149.

Gruber, T. R. et al. (1993). A translation approach to portable ontology specifications. *Knowledge acquisition*, *5*, 199–220.

Gudhjonsson, K. (2010). Mastering the super timeline with log2timeline. *SANS Reading Room*, .

Hargreaves, C., & Patterson, J. (2012). An automated timeline reconstruction approach for digital forensic investigations. *Digital Investigation*, *9*, 69–79.

James, J., Gladyshev, P., Abdullah, M., & Zhu, Y. (2010). Analysis of evidence using formal event reconstruction. *Digital Forensics and Cyber Crime*, (pp. 85–98).

Khan, M., Chatwin, C., & Young, R. (2007). A framework for post-event timeline reconstruction using neural networks. *Digital Investigation*, *4*, 146–157.

Lebo, T., Sahoo, S., McGuinness, D., Belhajjame, K., Cheney, J., Corsar, D., Garijo, D., Soiland-Reyes, S., Zednik, S., & Zhao, J. (2013). Prov-o: The prov ontology. *W3C Recommendation, 30th April*, .

Motik, B., Grau, B. C., Horrocks, I., Wu, Z., Fokoue, A., & Lutz, C. (2009). Owl 2 web ontology language: Profiles. *W3C recommendation*, *27*, 61.

Olsson, J., & Boldt, M. (2009). Computer forensic timeline visualization tool. *Digital Investigation*, *6*, 78–87.

Schatz, B., Mohay, G., & Clark, A. (2004). Rich event representation for computer forensics'. *Proceedings of the Fifth Asia-Pacific Industrial Engineering and Management Systems Conference (APIEMS 2004)*, *2*, 1–16.

Turnbull, B., & Randhawa, S. (2015). Automated event and social network extraction from digital evidence sources with ontological mapping. *Digital Investigation*, *13*, 94 – 106.