Chapter 4

The need for a theory of event reconstruction

This chapter begins with an explanation why a theory of event reconstruction is needed in digital forensics. Such a theory is needed to reduce reasoning errors, to support automation, and to fulfil legal requirements. This is the subject of Section 4.1.

Currently there is no such theory, but there are semi-formal techniques that structure event reconstruction in "ordinary" investigations. A review of key points of these techniques is given in Section 4.2. It is shown that the reviewed techniques do not fulfil the needs of digital forensics, because they do not facilitate automation, and their event reconstruction process remains informal and incomplete.

It is then argued in Section 4.3.2 that digital forensic investigations are more suitable for formalisation than "ordinary" investigations, and that complete formalisation of certain reconstruction techniques in digital forensics is both possible and desirable.

The chapter concludes with a problem statement, which defines objectives for the rest of the dissertation.

4.1 Why digital forensics need a theory of event reconstruction

As shown in Section 3.2.2, reconstruction of events in computer systems is an important task in digital forensic analysis. The development of a theory supporting event reconstruction techniques is needed in the digital forensic analysis for a number of reasons.

1. To *improve efficiency of analysis.* Formalisation of reconstruction techniques would facilitate their automation, which may reduce time required to perform them.

2. To *improve effectiveness of analysis.* Informal reasoning employed by existing reconstruction techniques increases the possibility of erroneous conclusions. The development of a formal reconstruction procedure based on an established theory of computer science would reduce the possibility of reasoning error, thus increasing effectiveness of the analysis.

3. To *satisfy admissibility requirements.* The presence of a sound theory that explains working of reconstruction techniques is warranted by the legal requirements for admissibility of expert evidence outlined in Section 2.2.1.

4.2 State of the art

Although the field of digital forensics is rapidly evolving, few publications to date explored the theoretical side of analysis and corroboration of digital evidence. The major developments include

- a classification of uncertainties accompanying digital evidence, and a method for reasoning about these uncertainties [24];

- the view of digital forensic tools as translators of information between different layers of abstraction inherent in computer software, and a way of defining such tools by specifying their translation function and error rate [22];
- the analysis of the possibility of using formal description of file systems for extracting data from binary images of disk drives [61];
- a demonstration that it is feasible to describe the outcome of investigation using a rigorous formal notation – colored petri nets [75].

With the exception of [75], none of these works addressed the problem of event reconstruction directly. In [75] it was argued that colored petri nets provide a convenient way to *illustrate* the results of otherwise informal event reconstruction, but no theory of event reconstruction process has been proposed.

Although currently there is no theory of event reconstruction in digital forensics, several attempts to bring structure and formality to the event reconstruction have been made in criminalistics and other branches of forensic science. They are discussed in the following subsections.

4.2.1 Attack trees

An important part of reconstruction process is identification of possible incident scenarios. Attack trees described in [73] is a semi-formal approach to discovery, documentation, and analysis of possible incident scenarios.

An attack tree is a diagram that describes different scenarios achieving some goal. The goal is represented by the root node. Other nodes represent subgoals that must be achieved – either alone, or in conjunction with other subgoals – to achieve the goal.

Figure 4.1 shows an attack tree of opening a safe without authorisation (this example is taken from [73]). The goal of the tree is to open a safe. The goal can be achieved in a number of different ways, such as picking the lock,

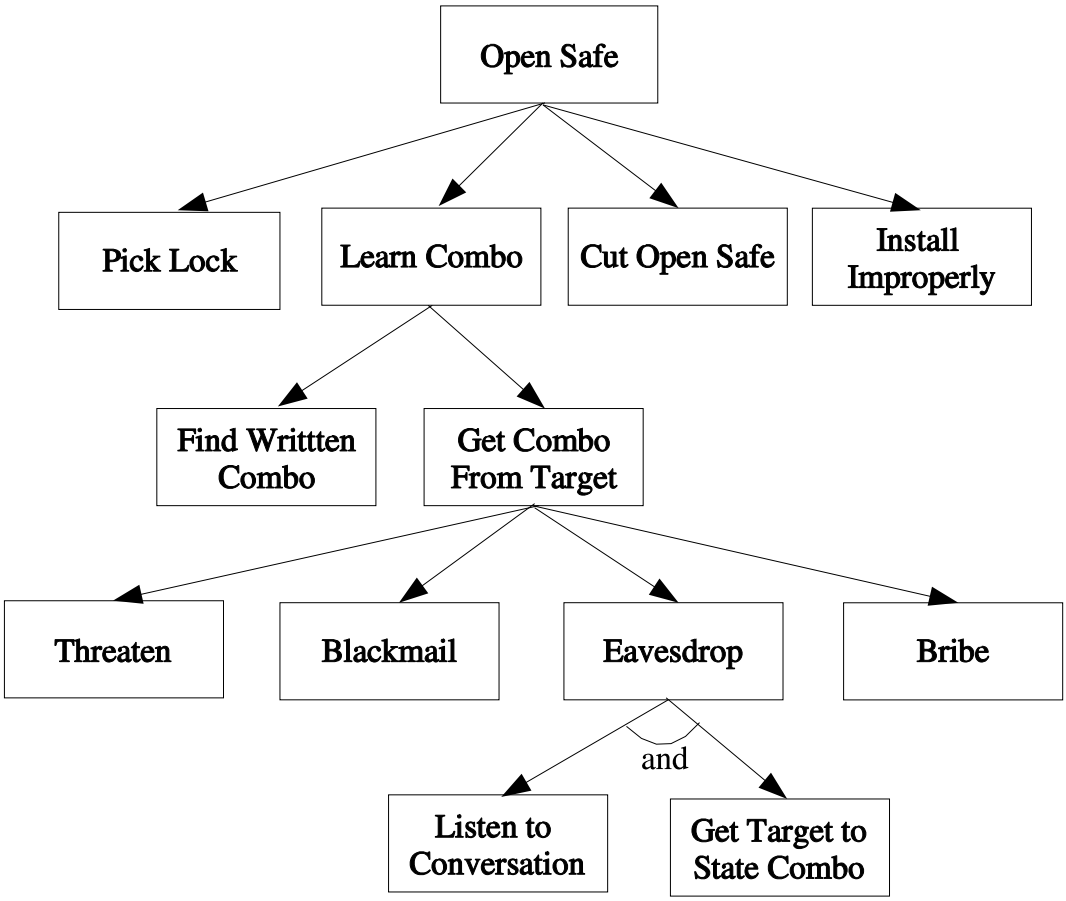

Figure 4.1: Attack tree describing different ways to open a safe

cutting through the safe wall, learning the combo, or making a security hole when the safe is installed. Different ways of learning a combo are elaborated under the "learn combo" node.

Basic attack tree is built from two types of node: AND nodes, and OR nodes. In this dissertation, AND node is graphically distinguished from an OR node by an arc that joins arrows coming out of the AND node. Both AND and OR nodes describe sub-goals that need to be fulfilled to fulfil the main goal of attack. To fulfil an OR node, any one of its child nodes must be fulfilled. To fulfil an AND node, all of its child nodes must be fulfilled. A possible scenario corresponds to every group of leaf nodes, whose joint fulfilment results in the fulfilment of the tree's root node.

To build an attack tree for a given goal, the analyst repeatedly splits the tree's goal into subgoals until the required level of detail is achieved. The process of splitting the goal into subgoals is informal. The analyst must use his or her intuition and common sense.

Once the attack tree is built, the analyst can calculate various properties of the discovered scenarios. The simplest property to calculate is whether a scenario is possible. Finding possible scenarios is a three-step process. In the first step, the analyst assigns a value "possible" or "impossible" to each leaf node. To decide whether a particular leaf node is possible or impossible, the analyst uses available evidence. In the second step, the analyst decides for every other node whether it is possible or not using the following rules

- an OR node is possible if *at least one* of its child nodes is possible;
- an AND node is possible if *all* of its child nodes are possible.

In the third step, all possible scenarios are identified by tracing them from root to leafs along chains of "possible" nodes.

Attack trees permit other types of scenario comparison. In particular, they can be used to calculate probability and cost of different scenarios.

4.2.2 Visual investigative analysis

Another semi-formal technique used for event reconstruction is Visual Investigative Analysis (VIA). It is a charting technique that

> uses a network approach to display graphically the sequences of occurrences and the relationships of all the elements of a criminal incident [65].

VIA emerged from the need to visualise complex crimes, in which many criminal activities go in parallel and interact with each other. Examples of such crimes are organised crime business manipulations and planned bankruptcies.

Graphical notation

VIA chart is a directed acyclic graph whose arrows represent activities and whose nodes represent states of the world in which activities start and finish. The following graphical notation is used to draw VIA charts.

- *Solid line arrows.* A solid line arrow represents single activity. The description of the activity is placed above the arrow. Activity always start at one node and ends at another. To improve clarity of the chart, only one solid line arrow is permitted between any two nodes.

- *Circles and triangles.* Circles and triangles are nodes. A node represents the world state in which one or more activities start or finish. *Terminal* is the last node in a sequence of activities. There is usually only one terminal on a chart. Terminals are drawn as triangles. All other nodes are drawn as circles.

- *Dotted lines and arrows.* A dotted line or arrow denotes a *dummy* activity. Dummy activity consumes no time and does nothing. It simply says that two nodes denote the same world state. Dotted lines are used for drawing parallel activities whose starting and ending world states are the same.

All nodes in VIA chart are given reference numbers. An activity is referred to by its starting and ending node numbers.

An example VIA chart is given in Figure 4.2. It shows a bank robbery at a very coarse level of detail.

Creation of VIA chart

Creation of VIA chart is a highly informal process. It consists of three interleaving stages: identification of activities, ordering of activities, and additional investigation.

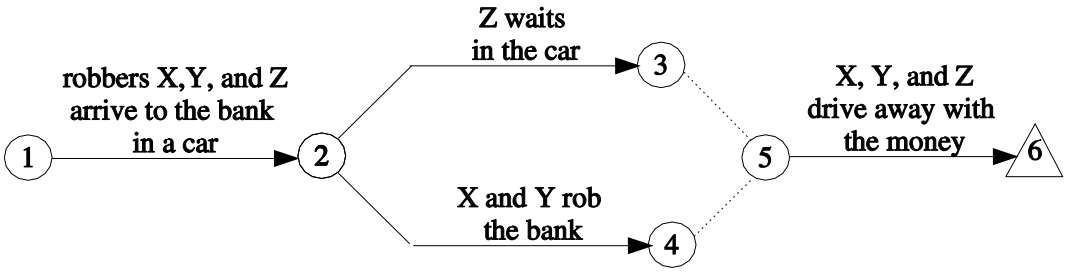

Figure 4.2: Example VIA chart

First stage is identification of activities. It consists in studying investigation reports and making a list of activities to be portrayed on a VIA chart.

Second stage is ordering of activities. The ordering is determined by answering three questions about each activity:

1. what activities precede this one;
2. what activities are concurrent with this one;
3. what activities follow this one.

During the second stage it may become clear that additional investigation is necessary to answer some of the questions. Such investigation is the task of the third stage. Several iterations of activity identification, activity ordering, and additional investigations may be required before VIA chart is complete. The ability to suggest additional investigations is a major advantage of the visual investigative analysis over unstructured investigations.

Visual investigative analysis does not provide a way to portray possible scenarios. Instead, the process of charting is interleaved with additional investigation until all possible scenarios except one are eliminated.

4.2.3 Multilinear events sequencing

Multilinear event sequencing (MES) is a set of semi-formal techniques for conducting investigations [13]. It is based on the same ideas as visual investigative

analysis and attack trees.

Multilinear event sequencing diagrams

The heart of MES are Multilinear event sequence diagrams (MES-diagrams). A MES-diagram portrays the crime or accident being investigated as a sequence of causally connected *events*, which represent activities.

MES-diagram shows events as rectangular *event blocks*. Each event block contains the following information about its event

- description of the action;
- actor – the object or person that performed the action;
- time of event.

Event blocks are connected via arrows that represent causation. If event X was necessary for event Y to occur, then MES-diagram contains an arrow from X to Y.

MES-diagrams have two axes. The horisontal axis represent time. The vertical axis lists actors involved in the accident. Event blocks are placed on the diagram according to their time and actor.

Any relevant information that does not fit event block structure can be placed on MES-diagram as a *condition*. Condition is an oval with some text inside. A condition is linked to an event block via an arrow.

Figure 4.3 shows MES-diagram of the bank robbery example from the previous section.

Creation of MES-diagrams

The process of creating MES-diagrams is similar to the process of creating VIA charts. First, events are identified. Second, causal connections between events are determined. Two techniques are defined to make creation of MES-diagrams more formal:

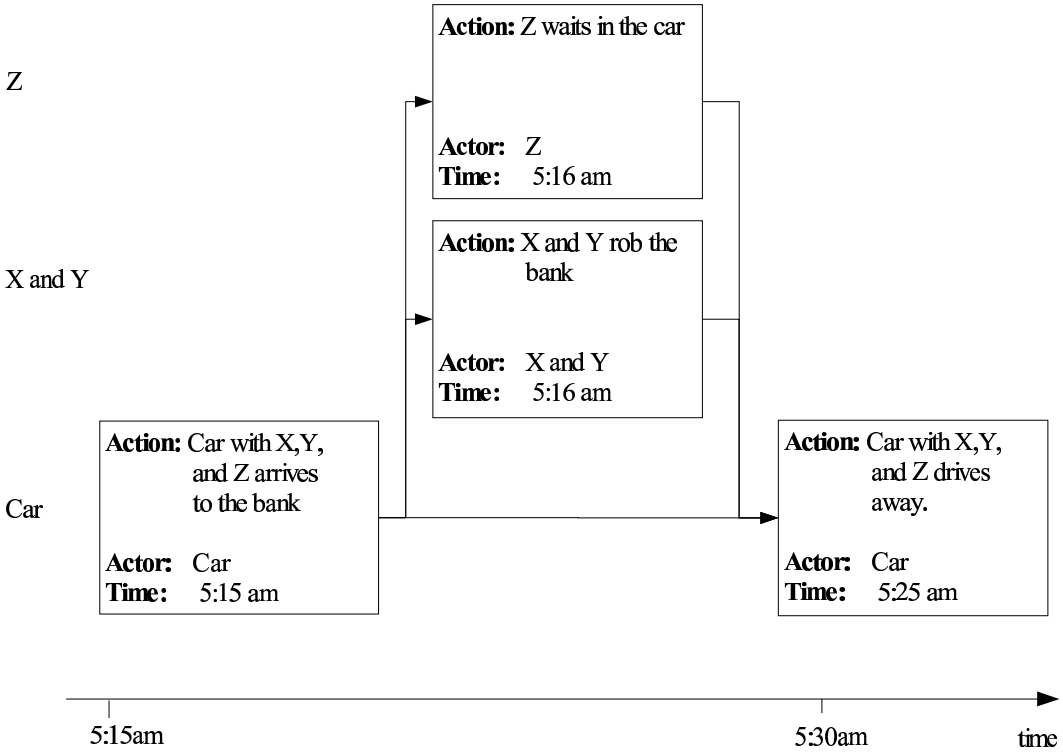

Figure 4.3: Example MES-diagram

1. *MES-trees.* Identification of events in MES is informal, but MES defines a technique called MES-trees [14] for elaborating possible scenarios of the crime or accident. MES-trees are very similar to attack trees described in Seton 4.2.1.

2. *Counterfactual reasoning.* Counterfactual reasoning is a way to establish causal dependency between events. It refers to the following philosophical argument due to David Hume [45]:

   > We may define a cause to be an object followed by another, and where all the objects, similar to the first, are followed by objects similar to the second. Or, in other words, where, if the first object had not been, the second never had existed.

   A mathematical formalisation of counterfactual reasoning has been given in [55] and [56].

Counterfactual reasoning is used on MES-diagram as follows. After events are identified, counterfactual reasoning is applied to every pair of events X and Y on the diagram. If X is a causal factor of Y, then an arrow is drawn from X to Y.

4.2.4 Why-because analysis

Why-Because Analysis (WBA) is a method for determining causes of complex accidents [53]. Like MES, it uses counterfactual reasoning to establish causality, but the approach taken by WBA is more formal.

Why-because graph

The graphical notation used by WBA to represent accidents is Why-because graph (WB-graph). It is a directed acyclic graph whose nodes represent elements of the accident, and whose arrows represent causal relationship.

Four kinds of nodes are defined for building WB-graphs. A node can be either of the following

- system state
- event – change from one state to another
- process – undifferentiated mix of events and states
- non-event – causally important absence of some event

Figure 4.4 shows WB-graph for the tarts rhyme from [23]:

> The Queen of Hearts, she made some tarts, All on a summer's day:
> The Knave of Hearts, he stole those tarts, And took them quite away!

Five nodes of WB-graph represent respectively:

1. Process: The queen of hearts makes tarts

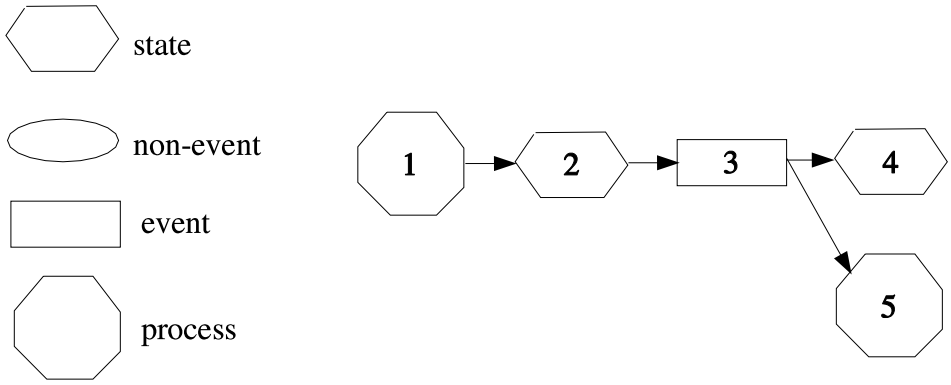

Figure 4.4: WB-graph for the tarts rhyme

2. State: the tarts are present

3. Event: Knave of hearts steels the tarts

4. State: the tarts are missing

5. Process: Knave of hearts takes the tarts away

The process of creating a WB-graph is similar to the process of creating MES-diagram. First, nodes of the graph are identified. Second, causal relationship between nodes is established using informal counterfactual reasoning.

Formal verification of WB-graph

After the WB-graph is constructed, it can be formally verified using *explanatory logic* defined by WBA.

Explanatory logic. Explanatory logic (EL), which is described in [57] and [58], has been purposefully developed for formal verification of WB-graphs. To achieve necessary expressiveness, EL combines three distinct formalisms: The Temporal Logic of Actions, the Standard Deontic Logic, and VCU logic.

- The Temporal Logic of Actions (TLA) described in [54] is a well known formalism for describing state based systems. TLA is used by EL for

modeling the real world. World states, events, processes, and required scientific knowledge are all described by TLA formulae.

- Standard Deontic Logic (SDL), which is described in [63], formalises "reasoning pertaining to obligation, permission, prohibition, and other normative matters [42]." SDL is used by EL to formalise laws and regulations. TLA is used in conjunction with SDL to formalise actions and objects mentioned in laws and regulations.

- VCU logic of D.K.Lewis [56] formalises counterfactual reasoning.

A well-formed EL formula is a VCU formula, whose basic propositions are TLA or SDL/TLA formulae.

Causal sufficiency of WB-graph. Formal verification of WB-graph consists in proving that every node in the graph (except for the nodes that represent root causes) is given a *causally sufficient* explanation.

Let $A_1, A_2, \ldots, A_n,$ and B be nodes in WB-graph, and let $A_1, A_2, \ldots, A_n$ be the nodes linked to B via an arrow (the arrow goes from A_i to B). Then the set $A_1, A_2, \ldots, A_n$ is a *causally sufficient* explanation of B if

1. $A_1, A_2, \ldots, A_n$ are *necessary* causal factors of B, which means that B cannot happen without prior happening of $A_1, A_2, \ldots, A_n$.

2. The happening of all $A_1, A_2, \ldots,$ and A_n will cause B to happen under any circumstances.

EL defines a set of proof rules for proving causal sufficiency. For example, proof rule (4.7) of [58] proves causal sufficiency through procedural necessity. It says that to prove that *circumstances* and *procedures* are causally sufficient explanation of event E, it suffices to prove that

1. Event E happened, and

2. *circumstances* occurred shortly before the event E, and

3. *procedures* were followed at all times, and

4. The *procedures* always lead to E if *circumstances* occur

The truth of items 1, 2, and 3 is usually assumed. The assumptions are motivated by available evidence. The correctness of item 4 is proved formally from formal definition of events and procedures.

EL and reconstruction process. EL formalises verification of reconstruction results, but it does not attempt to formalise the reconstruction process itself. The following observations support this conclusion.

First of all, the existence of WB-graph nodes (events, states, processes, and non-events) is *assumed* when formal verification starts. The initial discovery of the nodes is informal, and is not automated in WBA.

Second, WBA does not attempt to explore all possible causes of an event. It is explicitly stated in [58] that

> It would be a logical problem, interesting perhaps, but not always directly relevant to an incident investigation, to determine precisely which of the procedures might have been necessary to the occurrence of the event. We would not wish to enforce this investigation in all circumstances.

As a result, EL is designed to prove causal sufficiency of given set of nodes $A_1, A_2, \ldots, A_n$ with respect to another node B, rather than to explore all possible causes of B.

4.3 Summary and research problem statement

4.3.1 Analysis of the state of the art

Techniques described in this section can be readily used in digital forensics, but they do not *fulfil* the demand for a reconstruction theory. They reduce

reasoning errors by structuring the reconstruction process, but they neither automate reconstruction, nor ensure completeness of reconstruction. There are several reasons for that insufficiency.

- *Techniques described in this section were developed to assist human investigators rather than to create automated tools.* They do organise reconstruction process into a sequence of steps, but the investigator is expected to use his or her intuition to perform individual steps. Formal notation is used only to present and verify reconstruction results.

- *Techniques described in this section do not formalise the* entire *knowledge used by investigators.* The knowledge used by investigators in "ordinary" investigations is difficult to formalise in its entirety. First of all, it is a lot of knowledge – apart from "common sense" knowledge, it includes the laws of physics, engineering, psychology, medicine, etc. Second, much of that knowledge is too vague to be expressed and manipulated in formal logic.

4.3.2 Research problem statement

In some respects event reconstruction in digital forensics is easier than event reconstruction in "ordinary" investigations. The domain of digital forensics — computers — is only a part of the domain of "ordinary" investigations. In addition, computer functionality can be adequately described using mathematics and formal logic. However, formalisation of the entire knowledge used by digital forensic scientists is still impractical because of its continuing enlargement.

Some reconstruction techniques of digital forensics require only limited knowledge of computers. For example, identification of deleted and moved files on a disk volume requires only the knowledge of the file system, it does not require the knowledge of Internet protocols or anything else. Formalisation of such techniques is practically possible, because the body of knowledge to be formalised is limited and well defined.

This research does not aim to formalise all reconstruction techniques of digital forensic analysis. *The aim of this research is (1) to formalise event reconstruction in a general setting, that is, assuming nothing specific about the digital system under investigation or about the purpose of event reconstruction, and (2) to show that this formalisation can be used to describe and automate selected examples of digital forensic analysis.*

The rest of this dissertation describes an attempt to achieve these aims. After the necessary theoretical backgorund is defined in Chapter 5, Chapter 6 develops a mathematical definition of event reconstruction problem. Chapters 7 and 8 then demonstrate that the developed formalisation can be used to describe and automate specific techniques of digital forensic analysis. Chapter 7 constructs and implements a generic event reconstruction algorithm, which is based on the developed formalisation of event reconstruction. Chapter 8 uses that algorithm to formalise and automate examples of file system analysis and event time bounding analysis.