# Categories of Digital Investigation Analysis Techniques Based On The Computer History Model

*By*

**Brian Carrier, Eugene Spafford**

*Presented At*

The Digital Forensic Research Conference

**DFRWS 2006 USA** Lafayette, IN (Aug 14th - 16th)

PURDUE
UNIVERSITY

# Categories of Digital Investigation Analysis Techniques Based on the Computer History Model

Brian D. Carrier
and
Eugene H. Spafford

CERIAS

http://www.cerias.purdue.edu

# DFRWS 2004 Frameworks

- More like process models
- But, there is no unique process for an investigation
- Number of phases were subjective (including ours…)
- Completeness cannot be shown
- Useful for teaching, but not as useful for research and development

# The New Approach

1. Define an investigation model based on a standard computation model.
   - i.e. mathematical model
2. Define analysis technique categories based on the investigation model.

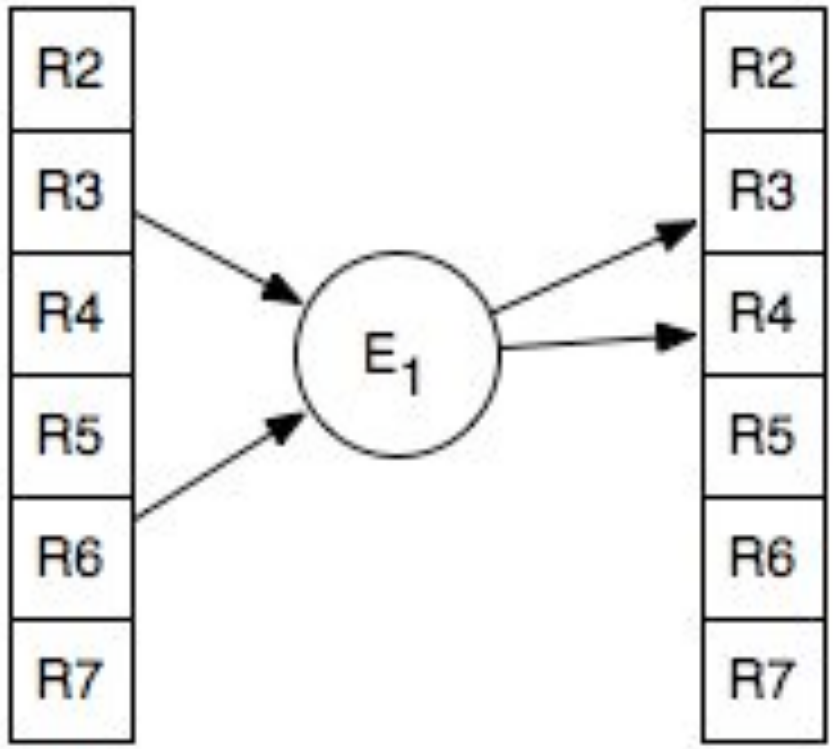
# Finite State Machine

- Finite State Machine (FSM)
  - Set of possible states: Q
  - Set of possible event symbols: $\Sigma$
  - State change function: $\delta$

$$Q \times \Sigma \rightarrow Q$$

- We assume that a computer **CAN** be represented by a FSM
  - Reduction is not performed during an investigation
  - FSM used for hardware / software independence

# Basic Event Visualization



# Computer History

- A computer's history contains the sequence of its previous states and events
- A digital investigation is a process to answer questions about previous and current states and events.
  - Starts with one or more known states
  - Makes inferences about the others
  - Searches the known and inferred states and events
- If you know the history, you can answer any question.

# Computer History Model

- Goal is to mathematically represent the computer's history.
- Define a set T with the times that the history exists.
- Map times in set T to the states in Q and events in ∑ that occurred.

$$h_{ps}: T \rightarrow Q$$

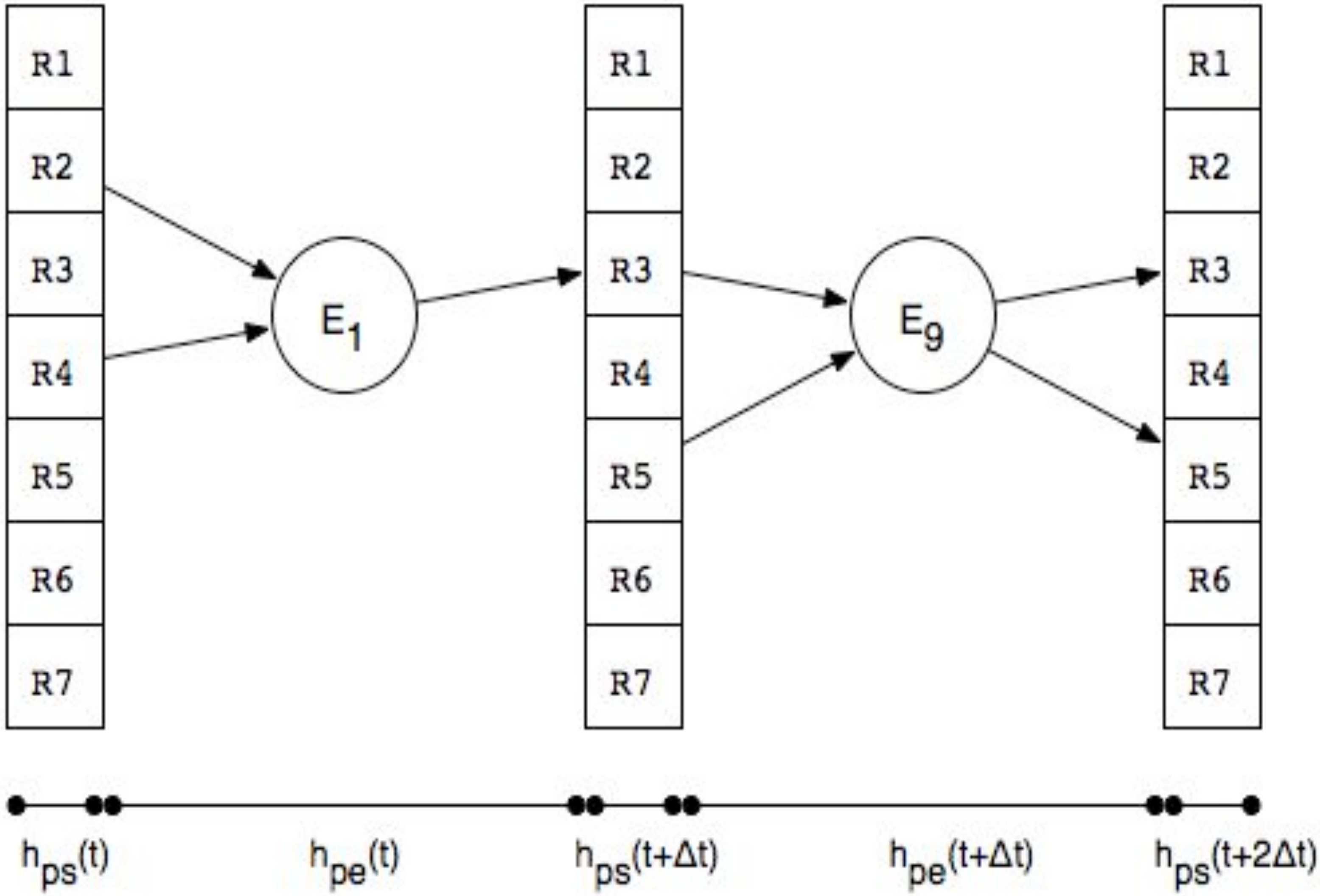
$$h_{pe}: T \rightarrow \Sigma$$

# Dynamic FSM

- Problem: The set of possible states and events at 2 times can be different in real systems.  Why?

# Dynamic FSM

- Problem: The set of possible states and events at 2 times can be different in real systems. Why?
- Sets Q and $\Sigma$ can change based on:
  - The devices that were connected.
  - The possible states for each device
    - Number of addresses
    - Domain of each address
  - The possible events for each device


R1
R2
R3
R4
R5
R6
R7
$E_1$
$E_9$
$h_{ps}(t)$
$h_{pe}(t)$
$h_{ps}(t+\Delta t)$
$h_{pe}(t+\Delta t)$
$h_{ps}(t+2\Delta t)$

# Summary (thus far)

- We assume a computer CAN be represented as a FSM.
- FSM must be dynamic and account for removable devices.
- We can represent the primitive history of the computer as a mapping from times to the FSM.

# Complex Systems

- Modern computers operate at “complex” levels
- Complex states: Defined by virtual storage locations that map and transform to primitive and lower-level storage locations.
  - Files, process memory, data structures…
- Complex events: A single event that causes multiple lower-level events to occur.
  - User-level events, buttons, system calls..
- A history exists for complex states and events

# Dynamic Complex Systems

- Number of possible complex events and states is based on:
  - The primitive devices connected
  - The programs on each device
  - The capabilities of each program
- A file exists only if programs on the computer supports the file system….
- We can map between the different layers (file type rules, event decomposition…)

# Summary (thus far)

- The computer history model can represent complex states and events.
- Complex capabilities are based on the devices and programs that exist.
- There is (at least) one mapping between the primitive and complex histories.

# Analysis Technique Categories

- If the computer history is known, we can answer any question.
- Our Hypothesis: The techniques required to define the computer history model are the same as required for a digital investigation.

# Category Overview

- Eight categories and each defines specific variables (27 variables total)
- Organizing into eight is intuitive, but not required
  - There is still subjectivity
- Each category has at least one class of techniques defined based on model and practice
  - Classes may increase over time

# History Duration Category (#1)

- Defines the set T of times when the computer has a history.
- When did the computer first exist?
  - Did the computer exist during the timeframe being investigated?
- Examples:
  - Manufactured date
  - OS Install date
  - Earliest MAC time

# Primitive Storage Capabilities Category (#2)

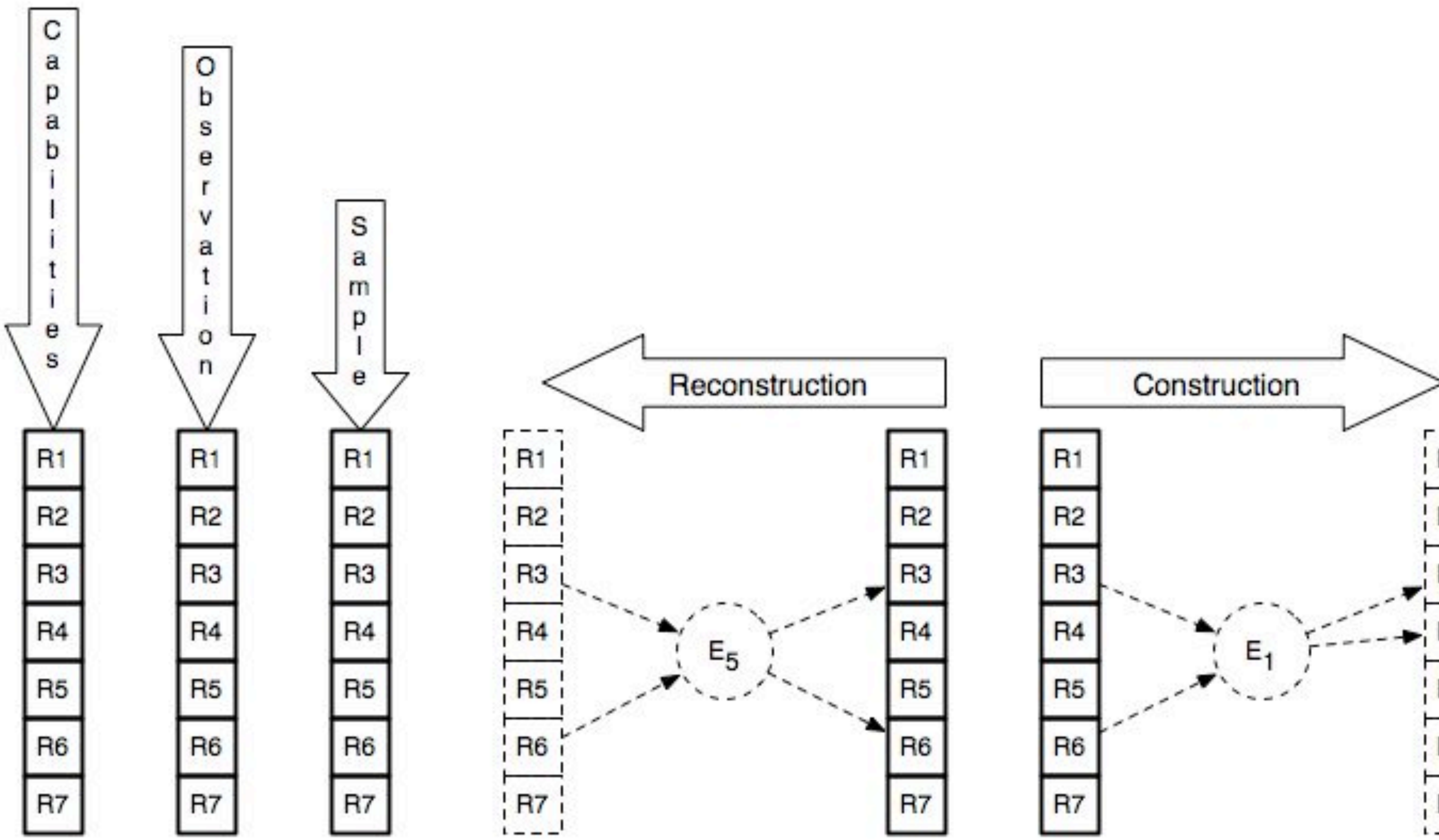
- Defines the possible states of the system at each time.
  - Which storage devices existed.
  - The possible states of each device.
  - When each device was connected.
- Examples:
  - Hard disk spec or query commands
  - Logs that record connected devices

# Primitive Event Capabilities Category (#3)

- Defines the possible events that could have occurred at each time.
  - The event devices that existed.
  - The possible events and state change functions for each event device.
  - When each device was connected.
- Examples:
  - Processor spec or query commands
  - Logs that record connected devices

# Primitive State and Event Definition Category (#4)

- Defines the states and events that are believed to have occurred
  - Observed states
  - Event and state reconstruction
  - Event and state construction
  - Sampling
  - Capabilities
- Can use one technique for defining a state or event and others for testing.


Capabilities
Observation
Sample
Reconstruction
Construction
R1
R2
R3
R4
R5
R6
R7
$E_5$
$E_1$

# Layers of Abstraction Definition Category (#5)

- Defines the layers of event abstractions
- Nearly impossible to determine
  - Requires knowledge about development process over lifetime of programs on system
  - Multiple equivalent layers exist
- In practice, make assumptions:
  - User-level events
  - File systems

# Complex Storage Capabilities Category (#6)

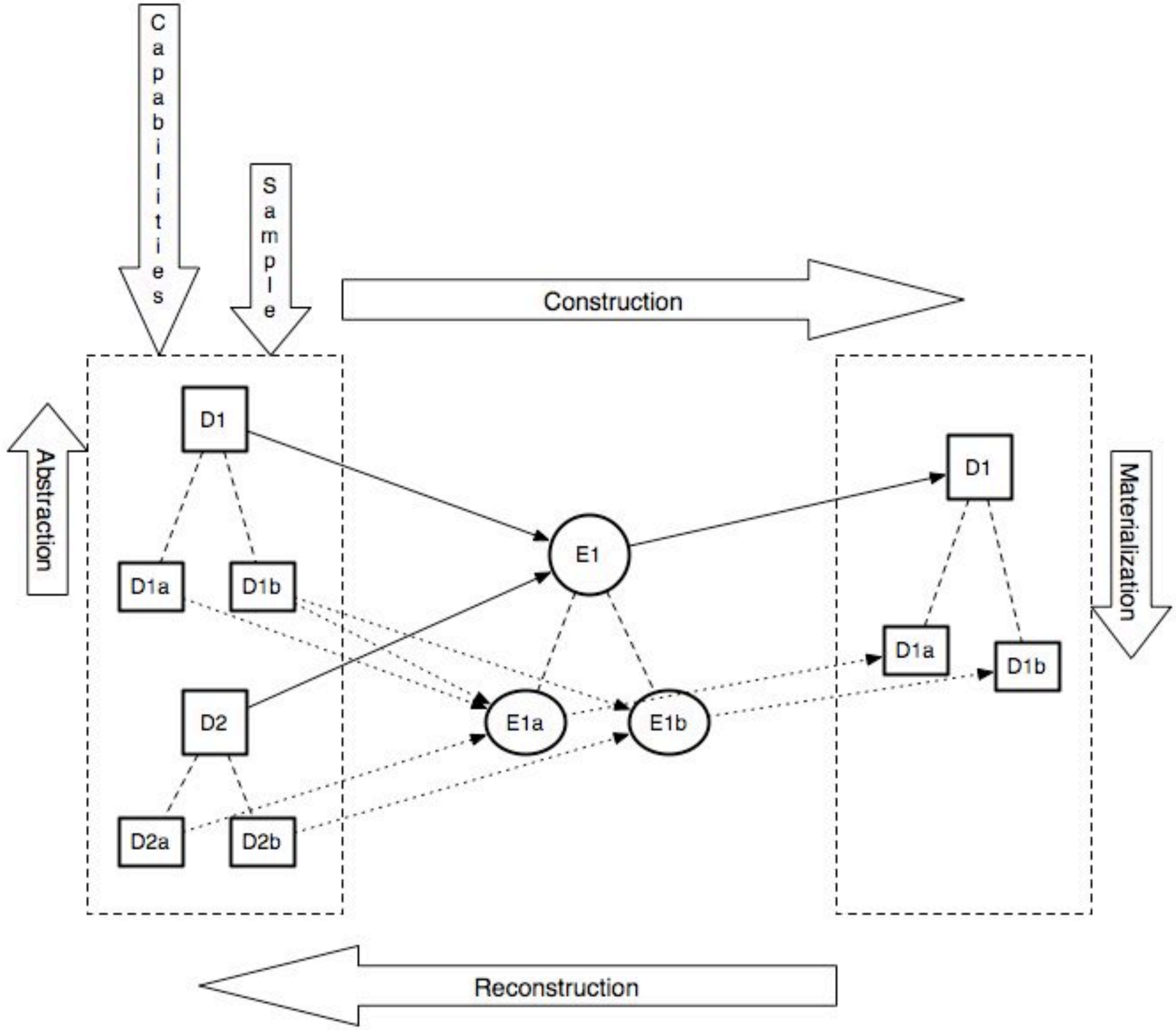
- Defines the possible complex storage states
  - Identify the programs that exist at each time (in theory)
  - Identify the complex storage types for each time.
- Examples:
  - Reverse engineer stored data
  - Static / dynamic analysis of programs
  - Program specifications

# Complex Event Capabilities Category (#7)

- Defines the possible complex events at each time.
  - Identify the programs that exist at each time (in theory)
  - Identify the complex events defined by each program
- Examples:
  - Static / dynamic analysis of programs
  - Program specifications

# Complex State and Event Definition Category (#8)

- Defines complex states and events that are believed to have occurred.
- Make inferences about previous events and states.
- Examples:
  - Event and state abstraction
  - Event and state materialization
  - Event and state reconstruction
  - Event and state construction


Capabilities
Sample
Construction
Abstraction
D1
D1a
D1b
D2
D2a
D2b
E1
E1a
E1b
D1
D1a
D1b
Materialization
Reconstruction

# Summary

- Previous frameworks / classifications not based on mathematical models.
- This work defined an investigation model based on a standard computation model.
- Categories of techniques can be shown to be complete, but structure is still subjective.
- The difference between previous frameworks is how they organize these categories.

# Questions?


Brian Carrier

carrier@digital-evidence.org

Eugene Spafford

spaf@cerias.purdue.edu