Eidetic Systems

David Devecsery, Michael Chow, Xianzheng Dou, Jason Flinn, Peter Chen
University of Michigan
What is an Eidetic System?

_Eidetic_ – Having “Perfect memory” or “Total Recall”

_Eidetic System_ – A system which can recall and trace through the lineage of any past computation
Motivation - Heartbleed

- Was Heartbleed exploited?
- What data was leaked?
Motivation - Heartbleed

- Was Heartbleed exploited? - Yes
- What data was leaked?

Heartbleed Message

Leaked Data
Motivation - Heartbleed

- Was Heartbleed exploited? - Yes
- What data was leaked?

Leaked Database Rows

Heartbleed Message

Leaked Data

Leaked Database Rows

Heartbleed Message

Leaked Data
Motivation – Wrong Reference

• How did I get the wrong citation?
• How did I get the wrong citation?
Motivation – Wrong Reference

• How did I get the wrong citation?
How did I get the wrong citation?
What else did this affect?
Motivation

• How did I get the wrong citation?
• What else did this affect?
Arnold

- First practical eidetic computer system
  - Efficiently records & recalls all user-space computation
    - Process register/memory state
    - Inter-process communication
  - Handles lineage queries
    - What data was affected?
    - What states and outputs were affected?
- Targeted towards desktop/workstation use
- Reasonable overheads
  - Record 4 years of data on $150 commodity HD
  - Under 8% performance overhead on most benchmarks
Overview

• Introduction
• Motivation
• How Arnold remembers all state
• How Arnold supports lineage queries
• Conclusion
Remembering State

• Requirements:
  • Store years of state on a single disk
    • Memory/register space within a process
    • Inter process communication
    • File state
  • Recall any state in reasonable time

• Solution:
  • Deterministic record & replay
    • “Process group” based replay
    • “Process graph” to track inter-process lineage
  • Log compression
Recording Granularity

• What granularity is best to record our system?
Recording Granularity

- Whole system recording
  - ✔ Low space overhead
  - × Costly to replay

David Devecserny
Recording Granularity

• Process level recording
  ✓ Efficient to replay
  ✗ Uses extra disk space
  ✗ No inter-process tracking

External Inputs

David Devecsey
Recording Granularity

- Process group recording
  - Efficient to replay
  - Reasonable disk space
  - No Inter-process tracking

External Inputs
Implementation – Process Graph

Record Log

IPC

Read

Pipe 1

Pipe 2

Read 1

David Devecsery
Implementation – Process Graph

Record Log

IPC

Read

Pipe 1

Pipe 2

David Devecserny
Recording

• Process group recording + process graph
  ✓ Efficient to replay
  ✓ Reasonable disk space
  ✓ Inter-process tracking
Space Optimizations

Log Compression vs Baseline

- Baseline
- Model-Based Compression
- Deduplicated File Cache
- X Server Compression
- Semi-Deterministic Time
- +Gzip

David Devecsery
Space Optimizations

Log Compression vs Baseline

<table>
<thead>
<tr>
<th>Treatment</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>1</td>
</tr>
<tr>
<td>+Model-Based Compression</td>
<td>411:1</td>
</tr>
<tr>
<td>+Deduplicated File Cache</td>
<td></td>
</tr>
<tr>
<td>+X Server Compression</td>
<td></td>
</tr>
<tr>
<td>+Semi-Deterministic Time</td>
<td></td>
</tr>
<tr>
<td>+Gzip</td>
<td></td>
</tr>
</tbody>
</table>

David Devecsey
Space Optimizations

![Bar Chart]

- Baseline vs. Compression
- Model-Based Compression vs. Baseline: 411:1 Ratio
- Deduplicated File Cache vs. Baseline: 6:1 Ratio
- X Server Compression vs. Baseline
- Semi-Deterministic Time vs. Baseline
- Gzip Compression vs. Baseline

David Devecsery
Space Optimizations

4 years of data on a $150 4TB commodity HD

Log Compression vs Baseline

- Baseline
- +Model-Based Compression
- +Deduplicated File Cache
- +X Server Compression
- +Semi-Deterministic Time
- +Gzip
- Only Gzip

411:1 Ratio
6:1 Ratio
Model-Based Compression

• Formulate a model of a typical execution
  • Only record deviations from that model
    
    \[
    \text{ret\_val} = \text{sys\_read} (\text{fd}, \text{buffer}, \text{count});
    \]
    
    \[
    \text{usually equal}
    \]

• Idea: Partial determinism
  • Encourage the program to conform to the model
Semi-Deterministic Time

- Frequent time queries are non-deterministic
- Use partially deterministic clock at record time
  - Real time clock & deterministic clock
  - Bound deviation

```java
if (deterministic_clock - real_time_clock < threshold) {
    adjust deterministic_clock
    record deviation
}
return deterministic_clock
```
Performance Evaluation

![Bar chart showing normalized runtime comparison between Baseline and Arnold for various applications. The chart includes applications like kernel copy, cvs checkout, make, latex, apache, gedit, facebook, spreadsheet.]
Overview

• Introduction
• Motivation
• How Arnold remembers all state
• How Arnold supports lineage queries
• Conclusion
Querying Lineage

• Two types of queries:
  • Reverse: Where did this data come from?
  • Forward: What did this data affect?

• How does Arnold support these queries?
  • User specifies initial state
  • Trace the lineage of the computation
    • Intra-process tracking
    • Inter-process tracking
Intra-Process Lineage

• Use taint tracking for intra-process causality
  • Run retroactively, on recorded execution
  • Parallelizable

• Arnold supports several notions of causality:

<table>
<thead>
<tr>
<th>Copy Only</th>
<th>Data Flow</th>
<th>Data+Index Flow</th>
<th>Control Flow</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strong input/output relation</td>
<td>Precision</td>
<td>Recall</td>
<td>Weak input/output Relation</td>
</tr>
<tr>
<td>May miss relations</td>
<td></td>
<td></td>
<td>Misses few relations</td>
</tr>
</tbody>
</table>

David Devecsery
Intra-Process Lineage

Which linkage tool should Arnold use?

Inputs

\textsc{BibTeX}

\textsc{LaTeX} → \textsc{LaTeX}

Program

\textsc{Vim}
Intra-Process Lineage

- Strong input/output relation
  - May miss relations

- Weak input/output Relation
  - Misses few relations

- Precision
- Recall

Copy

Data Flow

Data+Index Flow

David Devecsery
Intra-Process Lineage

Strong input/output relation
May miss relations

Precision
Recall

Weak input/output Relation
Misses few relations

Copy

Data Flow

Data+Index Flow

BIBTEX

LATEX

LATEX

Vim

David Devecsery
Intra-Process Lineage

Strong input/output relation
May miss relations

Precision
Recall

Weak input/output Relation
Misses few relations

Copy
Data Flow

BibTeX
LaTeX
Vim

BibTeX
LaTeX
Vim

BibTeX
LaTeX
Vim

Data+Index Flow
Intra-Process Lineage

Strong input/output relation
May miss relations

Precision
Recall
Weak input/output Relation
Misses few relations

Copy
Data Flow

Data+Index Flow

BibTeX

LaTeX

Vim

LaTeX

LaTeX

LaTeX

LaTeX

LaTeX

LaTeX

LaTeX

LaTeX

LaTeX

LaTeX

LaTeX

LaTeX
Intra-Process Lineage

Strong input/output relation
May miss relations

Precision

Weak input/output Relation
Misses few relations

Recall

Copy

BibTeX

LaTeX

Vim

Data Flow

LaTeX

LaTeX

LaTeX

Vim

Data+Index Flow

LaTeX

LaTeX

LaTeX

Vim

David Devecsery
Intra-Process Lineage

Strong input/output relation
May miss relations

Precision
Recall

Weak input/output Relation
Misses few relations

Data Flow

Arnold selects the most precise tool with at least one result
Inter-Process Lineage

• Two notions of inter-process linkage
• Process graph
  • Tracks lineage through inter-process communication
  • Precise
  • Captures group to group communication
• Human linkage
  • Handles relations between user inputs and outputs
  • Infers linkages based on data content and time
  • Imprecise – may have false negatives and false positives
  • Can capture linkages the process graph can miss
Evaluation – Wrong Reference

- Few false positives (font files, latex sty files, libc.so, libXt.so)
- No false negatives

<table>
<thead>
<tr>
<th>Record Time</th>
<th>Replay Time</th>
<th>Replay + Pin Time</th>
<th>Query Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>96.1s</td>
<td>2.2s</td>
<td>70.0s</td>
<td>209.5s</td>
</tr>
</tbody>
</table>
Evaluation – Heartbleed

- No false positives or negatives

<table>
<thead>
<tr>
<th></th>
<th>Record Time</th>
<th>Replay Time</th>
<th>Replay + Pin Time</th>
<th>Query Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data + Index</td>
<td>230.3s</td>
<td>0.4s</td>
<td>139.5s</td>
<td>235.1s</td>
</tr>
<tr>
<td>Data + Index</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data + Index</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Conclusion

• Eidetic Systems are powerful tools
  • Complete vision into past computation
  • Answer powerful queries about state’s lineage

• Arnold – First practical Eidetic System
  • Low runtime overhead
  • 4 years of computation on a commodity HD
  • Supports powerful lineage queries

• Code is released
  https://github.com/endplay/omniplay
Questions?
Backup Slides
Cloud Storage

• Future work
• Two approaches:
  • Statically served content
  • Distributed replay system
Related Works

• Execution Mining (Tralfamadore)
• DejaVu
• RAIL
## User Study Log-Sizes

<table>
<thead>
<tr>
<th>Users</th>
<th>Days</th>
<th>Groups Per Day</th>
<th>RAW File Cache</th>
<th>Logs</th>
<th>Filemap</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>25</td>
<td>995</td>
<td>475</td>
<td>267</td>
<td>36</td>
<td>779</td>
</tr>
<tr>
<td>B</td>
<td>24</td>
<td>475</td>
<td>1095</td>
<td>936</td>
<td>339</td>
<td>2064</td>
</tr>
<tr>
<td>C</td>
<td>21</td>
<td>26122</td>
<td>869</td>
<td>350</td>
<td>690</td>
<td>1910</td>
</tr>
<tr>
<td>D</td>
<td>16</td>
<td>3339</td>
<td>1675</td>
<td>838</td>
<td>838</td>
<td>2594</td>
</tr>
</tbody>
</table>