SEAL: Storage-efficient Causality Analysis on Enterprise Logs with Query-friendly Compression

Peng Fei, Zhou Li, Zhiying Wang, Xiao Yu, Ding Li, Kangkook Jee
Background and Motivation
Advanced persistent threat (APT): Long-term presence on a network in order to mine highly sensitive data.

Kaspersky’s Global Research and Analysis Team (GReAT) is well-known for the discovery and dissemination of the most advanced cyberthreats. According to their data, in 2020 the top targets for advanced persistent threats (APT) were governments, and the most significant threat actor was Lazarus.
An advanced persistent threat (APT):

- **Long-term** presence.
  - The latest intrusion prolongs over 188 days before detection. (TrustWave global report, 2015)

- **Highly sensitive data.**
  - Government data.
  - Banks.
  - ...
Background and Motivation

Intrusion happens on one node.

Nodes: files and programs in the servers.

Edges (logs): read, write and execute.
Background and Motivation

Intrusion transfers to another node
Background and Motivation

What can we do?

Use backtrack to find all the affected nodes.

(Casualty Analysis)

Intrusion detected!
Casualty Analysis:
reconstructs information flow by associating interdependent system events.

Intrusion detected!
Background and Motivation

- **High pressure on storage:**
  - e.g. NEC Laboratories America only sustains 3 month log data with more than 5TB from only 100 hosts.

- Compelling need for a solution to scale storage and processing capacity to meet the enterprise-level requirement.
Let's do some reductions for casualty analysis.
Background and Motivation

Let's do some reductions

5 times reduction rate!
However...

- Time constraint
However...

- Time constraint

C can no longer backtrack to A!
However...

- Time constraint
However...

- Time constraint

B can no longer backtrack to D!
Background and Motivation

Current Compaction Techniques:

  - Remove repeated edges under full dependency preservation (FD). [Hossain, M. 2018]
  - Remove nodes that only contain read-only file. [Lee, K. H., et al. 2013]

- No 100% guarantee for the same casualty analysis outcome.
How about **Lossless Compression**?

- **Standard tools** ([Deutsch, Peter. 1996][1], [Bartík, et all, 2015][2])
  - (Gzip, LZ4, etc)
- **DataBase Compression** ([Roth, et all 1993][3])

<table>
<thead>
<tr>
<th></th>
<th>compression rate</th>
<th>decompression overhead</th>
<th>support time constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard tools</td>
<td>high</td>
<td>high</td>
<td>Yes</td>
</tr>
<tr>
<td>DataBase Compression</td>
<td>high</td>
<td>low</td>
<td>No</td>
</tr>
</tbody>
</table>

- **Our Solution**: Query-friendly Lossless compression.

G’ : compressed graph.

Q’ : query result on the compressed graph.
Background and Motivation

Design Architecture
Compression on Logs dependency graph
Compression on Logs dependency graph

Compression on Graph Structure: **Nodes merge**

Node Map

<table>
<thead>
<tr>
<th>New Node</th>
<th>Represented Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>A, B</td>
</tr>
<tr>
<td>b</td>
<td>B, C</td>
</tr>
<tr>
<td>c</td>
<td>G, H</td>
</tr>
</tbody>
</table>
Compression on Logs dependency graph

Compression on Graph Structure: **Nodes merge**
Compression on Logs dependency graph

Compression on Graph Structure: **Nodes merge**

<table>
<thead>
<tr>
<th>New Node</th>
<th>Represented Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>A, B</td>
</tr>
<tr>
<td>b</td>
<td>B, C</td>
</tr>
<tr>
<td>c</td>
<td>G, H</td>
</tr>
</tbody>
</table>

Node Map
Compression on Logs dependency graph

Compression on Graph Structure: Nodes merge

Node Map

<table>
<thead>
<tr>
<th>New Node</th>
<th>Represented Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>A, B</td>
</tr>
<tr>
<td>b</td>
<td>B, C</td>
</tr>
<tr>
<td>c</td>
<td>G, H</td>
</tr>
</tbody>
</table>
Logs dependency graph compression

Compression on Edge properties:

Sorting: ascending for the starttime, descending for the endtime

<table>
<thead>
<tr>
<th>starttime</th>
<th>endtime</th>
<th>srcid</th>
<th>dstid</th>
<th>agentid</th>
<th>accessright</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1562723488971, 1562723488991]</td>
<td>[1562734588984, 1562734588995]</td>
<td>15</td>
<td>27</td>
<td>-582777938</td>
<td>Execute</td>
</tr>
</tbody>
</table>
Compression on edge properties:

- Delta Coding: Represents a sequence of values with the differences(or delta).

Long integer sequences that share a long prefix.

<table>
<thead>
<tr>
<th>starttime</th>
<th>endtime</th>
<th>srcid</th>
<th>dstid</th>
<th>agentid</th>
<th>accessright</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1562723488971, 1562723488991]</td>
<td>[1562734588984, 1562734588995]</td>
<td>15</td>
<td>27</td>
<td>-582777938</td>
<td>Execute</td>
</tr>
</tbody>
</table>
Compression on Edge properties:

- Delta Coding: Represents a sequence of values with the differences (or delta).

<table>
<thead>
<tr>
<th>starttime</th>
<th>endtime</th>
<th>srcid</th>
<th>dstid</th>
<th>agentid</th>
<th>accessright</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1562723488971, 20]</td>
<td>[1562734588984, 11]</td>
<td>15</td>
<td>27</td>
<td>-582777938</td>
<td>Execute</td>
</tr>
</tbody>
</table>
Query and decompression
Query and decompression
select * from id == 27 and endtime < 1562734588996

We can backtrack to node 15 without any decompression.

<table>
<thead>
<tr>
<th>starttime</th>
<th>endtime</th>
<th>srcid</th>
<th>dstid</th>
<th>agentid</th>
<th>accessright</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1562723488971, 20]</td>
<td>[1562734588995, -11]</td>
<td>15</td>
<td>27</td>
<td>-582777938</td>
<td>Execute</td>
</tr>
</tbody>
</table>
Query and decompression

```sql
select * from id == 2 and endtime < 1562734588990
```


<table>
<thead>
<tr>
<th>starttime</th>
<th>endtime</th>
<th>srcid</th>
<th>dstid</th>
<th>agentid</th>
<th>accessright</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1562723488971, 20]</td>
<td>[1562734588995, -11]</td>
<td>15</td>
<td>27</td>
<td>-582777938</td>
<td>Execute</td>
</tr>
</tbody>
</table>
Evaluation
Evaluation

Compression ratio:

- NEC Lab: 7.6GB/20GB (5M events)

- DARPA Transparent Computing program [DARPA/I20, 2020]: 18GB/233GB (1,183M events)
Evaluation

Compression Ratio Comparison:

- **Optimal** ratio for edges compression. Only leave one edge between one pair of nodes.
  - Full dependency (FD) and our edge only method both gain similar results.

- Better Ratio if node merged is added.
Query efficiency:

- 89% cases have less total time than the uncompressed data. (< 100%).

- On average decompression only takes 18.66% of the overall time.
Thank you!