FAuST: Striking a Bargain between Forensic Auditing's Security and Throughput

Muhammad Adil Inam, Akul Goyal, Jason Liu, Jaron Mink, Noor Michael, Sneha Gaur, Adam Bates, Wajih Ul Hassan

Annual Computer Security Applications Conference (ACSAC) 2022
December 9, 2022
Logs are Critical but Expensive

- High-profile APT attacks last for years [1]

[4] Shiqing Ma et al., USENIX ATC ‘18
[5] Kyu Hyung Lee et al., CCS ’13
Logs are Critical but Expensive

• High-profile APT attacks last for years [1]

[4] Shiqing Ma et al., USENIX ATC ’18
[5] Kyu Hyung Lee et al., CCS ’13
Logs are Critical but Expensive

- High-profile APT attacks last for years [1]
- Most organizations store logs for a few months [2,3]

[4] Shiqing Ma et al., USENIX ATC ’18
[5] Kyu Hyung Lee et al., CCS ’13
Logs are Critical but Expensive

- High-profile APT attacks last for years [1]
- Most organizations store logs for a few months [2,3]
- Each machine can generate 400-1200 GB per year [4,5]
- Log analysis often costs at least $1500 per GB [6]

[4] Shiqing Ma et al., USENIX ATC ’18
[5] Kyu Hyung Lee et al., CCS ’13
Keeping only Important Logs

• Data compression only gets so far [1]

• Investigation tools need to be able to search for key events

• Researchers instead leverage data provenance to reduce logs

[1] https://www.elastic.co/blog/filebeat-modiles-access-logs-and-elasticsearch-storage-requirements
Provenance graphs allow us to reason about causality relationships
Provenance graphs allow us to reason about causality relationships.

**Audit Log**

\[ t1, \text{open}, \text{firefox, /tmp/foo} \]

**Provenance Graph**

\[ t1 \]

\[ /tmp/foo \]

\[ \text{firefox} \]
Provenance graphs allow us to reason about causality relationships.

Audit Log

- t1, open, firefox, /tmp/foo
- t2, write, firefox, /tmp/foo

Provenance Graph

- /tmp/foo
- firefox
- t1
- t2
Provenance graphs allow us to reason about causality relationships.

Audit Log

- t1, open, firefox, /tmp/foo
- t2, write, firefox, /tmp/foo
- t3, clone, firefox, bash

Provenance Graph

- Graph showing the relationships between Firefox, /tmp/foo, and Bash.
Provenance graphs allow us to reason about causality relationships.

Audit Log

- t1, open, firefox, /tmp/foo
- t2, write, firefox, /tmp/foo
- t3, clone, firefox, bash
- t4, read, firefox, /tmp/foo

Provenance Graph
• Provenance graphs allow us to reason about causality relationships

**Audit Log**

- t1, open, firefox, /tmp/foo
- t2, write, firefox, /tmp/foo
- t3, clone, firefox, bash
- t4, read, firefox, /tmp/foo
- t5, socket, bash, X.X.X.X

**Provenance Graph**

```
firefox
/ tmp/foo
bash
X.X.X.X
```
• Provenance graphs allow us to reason about causality relationships

Audit Log

- t1, open, firefox, /tmp/foo
- t2, write, firefox, /tmp/foo
- t3, clone, firefox, bash
- t4, read, firefox, /tmp/foo
- t5, socket, bash, X.X.X.X
- t6, write, bash, X.X.X.X

Provenance Graph

- firefox
- /tmp/foo
- bash
- X.X.X.X
- t1 → t2 → t3
- t4 → t1
- t5 → t6
Provenance graphs allow us to reason about causality relationships.

Audit Log

- t1, open, firefox, /tmp/foo
- t2, write, firefox, /tmp/foo
- t3, clone, firefox, bash
- t4, read, firefox, /tmp/foo
- t5, socket, bash, X.X.X.X
- t6, write, bash, X.X.X.X

Provenance Graph

- firefox
  - t1
  - t2
  - t4
- /tmp/foo
- bash
  - t5
  - t6
- X.X.X.X
Provenance graphs allow us to reason about causality relationships.

Audit Log

- t1, open, firefox, /tmp/foo
- t2, write, firefox, /tmp/foo
- t3, clone, firefox, bash
- t4, read, firefox, /tmp/foo
- t5, socket, bash, X.X.X.X
- t6, write, bash, X.X.X.X

Provenance Graph

- firefox
- /tmp/foo
- bash
- X.X.X.X
Reduction Techniques

• LogGC: remove temporary file I/O that cannot affect other parts of the graph [1]
  • Claimed ~93-97% reduction

• CPR: remove parallel edges that do not add any new causal information [2]
  • Claimed ~56% reduction, and that it can be combined with LogGC

[1] Kyu Hyung Lee et al., CCS ‘13
[2] Zhang Xu et al., CCS ‘16
Reduction Techniques

- **LogGC**: remove temporary file I/O that cannot affect other parts of the graph [1]
  - Claimed ~93-97% reduction

- **CPR**: remove parallel edges that do not add any new causal information [2]
  - Claimed ~56% reduction, and that it can be combined with LogGC

---

[1] Kyu Hyung Lee et al., CCS ’13
[2] Zhang Xu et al., CCS ’16
Reduction Techniques

- LogGC: remove temporary file I/O that cannot affect other parts of the graph [1]
  - Claimed ~93-97% reduction
- CPR: remove parallel edges that do not add any new causal information [2]
  - Claimed ~56% reduction, and that it can be combined with LogGC

[1] Kyu Hyung Lee et al., CCS ’13
[2] Zhang Xu et al., CCS ’16
Reduction Techniques

- LogGC: remove temporary file I/O that cannot affect other parts of the graph [1]
  - Claimed ~93-97% reduction
- CPR: remove parallel edges that do not add any new causal information [2]
  - Claimed ~56% reduction, and that it can be combined with LogGC

[1] Kyu Hyung Lee et al., CCS ‘13
[2] Zhang Xu et al., CCS ‘16
Comparing Reduction Techniques

• To what degree are different techniques actually orthogonal?

  - LogGC
  - CPR

• Particular log datasets may be better suited to certain techniques

• Is ~95% reduction for LogGC and ~56% for CPR a fair comparison?
Comparing Reduction Techniques

• To what degree are different techniques actually orthogonal?

• Particular log datasets may be better suited to certain techniques

• Is ~95% reduction for LogGC and ~56% for CPR a fair comparison?
Comparing Reduction Techniques

• To what degree are different techniques actually orthogonal?

• Particular log datasets may be better suited to certain techniques

• Is ~95% reduction for LogGC and ~56% for CPR a fair comparison?
FAuST: Transparent & Modular Reduction

- Implement reduction techniques in an extensible modular framework
- Combine multiple reduction techniques simultaneously
- Evaluate and compare reduction performance and throughput for any combination
FAuST Design Overview

Streaming Audit Logs

Downstream Tools

Streaming Audit Logs
FAuST Design Overview

Downstream Tools

Streaming Audit Logs

FAuST

auditd

Streaming Audit Logs
FAuST Design Overview

- auditd
  - Streaming Audit Logs
- splunk
  - Streaming Audit Logs
FAuST Design Overview

Streaming Audit Logs

auditd

FAuST

splunk

Streaming Audit Logs
FAuST Design Overview

Streaming Audit Logs

Log Parser

FAuST

 auditd

Splunk

Streaming Audit Logs
FAuST Design Overview

- **Log Parser**
- **Provenance Graph**
- **Log Event Buffer**
- Streaming Audit Logs
- **FAuST**
- **splunk**
- **Auditd**
- **Streaming Audit Logs**
FAuST Design Overview

FAuST

Provenance Graph

Reduction Filters

Log Parser

Log Event Buffer

auditd

Streaming Audit Logs

Splunk

Streaming Audit Logs
FAuST Design Overview

FAuST

Provenance Graph

Reduction Filters

Reduction Decisions

Log Event Buffer

Reduction Engine

Log Parser

Streaming Audit Logs

auditd

Splunk

Streaming Audit Logs
Local and Batch Filters

- Local techniques: analyze subgraphs in response to certain events
- Global techniques: analyze entire graph in offline setting
• Local techniques: analyze subgraphs in response to certain events
• Global techniques: analyze entire graph in offline setting
Local and Batch Filters

- Local techniques: analyze subgraphs in response to certain events
- Global techniques: analyze entire graph in offline setting
Local and Batch Filters

- Local techniques: analyze subgraphs in response to certain events
- Global techniques: analyze entire graph in offline setting

<table>
<thead>
<tr>
<th>Local</th>
<th>Batch</th>
</tr>
</thead>
<tbody>
<tr>
<td>LogGC</td>
<td>NodeMerge</td>
</tr>
<tr>
<td>Kyu Hyung Lee et al.</td>
<td>Yutao Tang et al.</td>
</tr>
<tr>
<td>CCS ‘13</td>
<td>CCS ‘18</td>
</tr>
<tr>
<td>CPR and PCAR</td>
<td>F- and S-DPR</td>
</tr>
<tr>
<td>Zhang Xu et al.</td>
<td>Md Nahid Hossain et al.</td>
</tr>
<tr>
<td>CCS ‘16</td>
<td>USENIX Security ‘18</td>
</tr>
<tr>
<td></td>
<td>Winnower</td>
</tr>
<tr>
<td></td>
<td>Wajih Ul Hassan et al.</td>
</tr>
<tr>
<td></td>
<td>NDSS ‘18</td>
</tr>
<tr>
<td></td>
<td>LogApprox</td>
</tr>
<tr>
<td></td>
<td>Noor Michael et al.</td>
</tr>
<tr>
<td></td>
<td>ACSAC ‘20</td>
</tr>
</tbody>
</table>
Reduction Filters: CPR

- t1, clone, ...
- t2, read, ...
- ...
- t4, read, ...
- t5, read, ...
- ...
- t11, read, ...
- t12, clone, ...
- t13, read, ...
- t14, clone, ...
- t15, send, ...
- ...
- t21, recv, ...
- t22, connect, ...
- t23, clone, ...
- t24, send, ...
- ...
- t30, recv, ...
- t31, write, ...
- t32, exec, ...
- t33, connect, ...

Zhang Xu et al., CCS '16
Reduction Filters: CPR

- t1, clone, ...
- t2, read, ...
- ...
- * t4, read, ...
- t5, read, ...
- ...
- * t11, read, ...
- t12, clone, ...
- t13, read, ...
- t14, clone, ...
- t15, send, ...
- ...
- * t21, recv, ...
- t22, connect, ...
- t23, clone, ...
- t24, send, ...
- ...
- t30, recv, ...
- t31, write, ...
- t32, exec, ...
- t33, connect, ...

Zhang Xu et al., CCS ‘16
Reduction Filters: NodeMerge

Yutao Tang et al., CCS ’18

t1, clone, ...
t2, read, ...
* ...
* t4, read, ...
t5, read, ...
* ...
* t11, read, ...
t12, clone, ...
t13, read, ...
t14, clone, ...
t15, send, ...
* ...
* t21, recv, ...
t22, connect, ...
t23, clone, ...
t24, send, ...
...
t30, recv, ...
t31, write, ...
t32, exec, ...
t33, connect, ...

/ home/admin/clean

146.153.68.151

161.116.88.72

cert8.db

128.55.12.110

libdl-2.15.so
libtinfo.so.5.9
org.chromium.iyhyah
Reduction Filters: NodeMerge

Yutao Tang et al., CCS ’18
Reduction Filters: S-DPR

- t1, clone, ...
- t2, read, ...
- t4, read, ...
- t5, read, ...
- t11, read, ...
- t12, clone, ...
- t13, read, ...
- t14, clone, ...
- t15, send, ...
- t21, recv, ...
- t22, connect, ...
- t23, clone, ...
- t24, send, ...
- t30, recv, ...
- t31, write, ...
- t32, exec, ...
- t33, connect, ...

Md Nahid Hossain et al., USENIX Security ’18
Reduction Filters: S-DPR

Md Nahid Hossain et al., USENIX Security ’18
Final Reduction

t1, clone, ...
t2, read, ...

*  *  *  ...
*  *  t4, read, ...
*  t5, read, ...
***  *  *
***  t11, read, ...
t12, clone, ...
t13, read, ...
t14, clone, ...
t15, send, ...

*  *  *
*  *  t21, recv, ...
t22, connect, ...
t23, clone, ...
*  t24, send, ...
*  *
*  *  ...
*  *  t30, recv, ...
t31, write, ...
t32, exec, ...
t33, connect, ...

firefox

146.153.68.151

fluxbox

firefox

t1

firefox

t2-4

t5-11

t5-11

t5-11

firefox

t12

t13

t14

t15-21

t15-21

t15-21

firefox

t22

t23

t24-31

t24-31

t24-31

/home/admin/clean

clean

161.116.88.72

128.55.12.110

cert8.db

libdl.2.15.so

libtinfo.so.5.9

org.chromium.iyhyah

CPR
NodeMerge
S-DPR

ILLOIS
t1, clone, ...
t2, read, ...
t5, read, ...
t12, clone, ...
t13, read, ...
t14, clone, ...
t15, send, ...
t22, connect, ...
t23, clone, ...
t31, write, ...
t32, exec, ...
t33, connect, ...

Node Template

Final Reduction
Reduction Evaluation

• Diminishing returns for increasing number of filters

• 2-3 filters is generally a decent tradeoff of reduction to performance
Reduction Evaluation

- Diminishing returns for increasing number of filters
- 2-3 filters is generally a decent tradeoff of reduction to performance
Reduction Evaluation

- Diminishing returns for increasing number of filters
- 2-3 filters is generally a decent tradeoff of reduction to performance
Reduction Evaluation

- Diminishing returns for increasing number of filters
- 2-3 filters is generally a decent tradeoff of reduction to performance
Reduction Evaluation

- Diminishing returns for increasing number of filters
- 2-3 filters is generally a decent tradeoff of reduction to performance
Performance Evaluation

Reduction performance largely levels off by 100k logs per batch.
Local filters have much higher throughput than batch filters.
Conclusion

• FAuST: easily implement and evaluate log reduction techniques

• Available open-source at https://bitbucket.org/sts-lab/faust

• Transparent log reduction tool for any log analysis project or workflow

• Easy baseline comparison with 8 existing techniques for new reductions

• We use FAuST to enable our SoK on log reduction techniques [1]

Muhammad Adil Inam <mainam2@illinois.edu>
Jason Liu <jdliu2@illinois.edu>

[1] Muhammad Adil Inam et al., IEEE SP’23