Towards Scalable Cluster Auditing through Grammatical Inference over Provenance Graphs

Wajih Ul Hassan, Mark Lemay, Nuraini Aguse, Adam Bates, Thomas Moyer

NDSS Symposium 2018
Feb 20, 2018
Notable Data Breach in 2017

Equifax Data Breach Impacts 143 Million Americans

Equifax, Apache Struts, & CVE-2017-5638 Vulnerability

We recently discovered a cybersecurity incident involving consumer information. Once discovered, we acted immediately to stop the intrusion.
Notable Data Breach in 2017

Equifax Data Breach Timeline 2017

<table>
<thead>
<tr>
<th>Date</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apr</td>
<td>Breached</td>
</tr>
<tr>
<td>May</td>
<td>Detected</td>
</tr>
<tr>
<td>Jun</td>
<td>Hackers in Equifax Servers</td>
</tr>
<tr>
<td>Jul</td>
<td>Patched</td>
</tr>
<tr>
<td>Aug</td>
<td>Breached</td>
</tr>
<tr>
<td>Sep</td>
<td>Announced</td>
</tr>
</tbody>
</table>
Notable Data Breach in 2017

Equifax Data Breach Timeline 2017

3 Months of crucial attack audit logs

Hackers in Equifax Servers

Patched
Notable Data Breach in 2017

Equifax Data Breach Timeline 2017

Breached

Detected

Breached

Announced

3 Months of crucial attack audit logs

Are current auditing systems scalable?

Hackers in Equifax Servers

Patched
Data Provenance aka Audit log

- Lineage of system activities
- Represented as Directed Acyclic Graph (DAG)
- Used for forensic analysis

**Code Execution**
- **Bash:**
  ```bash
eexec("./NGINX");
  ```
- **NGINX:**
  ```c
  recv(...,"abc.com");
  fread("index.html");
  ```

**Audit log**
1. **Bash**, Spawns **NGINX**
2. **NGINX**, Receives from **abc.com**
3. **NGINX**, Reads File **index.html**
4. ...........

**Provenance Graph**
- **Bash**
- **abc.com**
- **NGINX**
- **index.html**
Data Provenance in a Cluster

Centralized auditing not practical due to two limitations
Limitation#1: Graph Complexity

- NGINX and MySQL running for 5 mins on a single machine
Limitation#2: Storage overhead

- Leads to network overhead as logs are transferred to master node

![Audit Log Size Growth for a Single NGINX Server](image)

- Uncompressed
- Compressed

- 2.54 GB/Day on a single machine
- With compression >1GB/Day
Winnower

- Cluster applications are replicated in accordance with microservice architecture principle
- Replicated apps produce highly homogeneous provenance graphs
  - core execution behaviour is similar

**Key Idea:**
Remove redundancy from provenance graphs across cluster before sending to master node
Master Node View with Winnower

Before

After

Bash

mysql

mysqlld

NGINX

/uptime

/uptime/

/db/*
• Build consensus model across cluster using graph grammars
• Like string grammar, graph grammars provide rule-based mechanisms
  • For generating, manipulating and analyzing graphs
  • *Induction* – produce grammar from a given graph
  • *Parsing* – membership test of a given graph is in a grammar

\[
S \equiv A \rightarrow T
\]
\[
A \equiv a \rightarrow B
\]
\[
T \equiv t \rightarrow S
\]
\[
B \equiv b
\]

Graph

Graph Grammar
Architecture

- Master Node
  - Model Aggregator
  - Winnower Agent

- Worker Nodes

- Audit Module
  - Prov. Graph
  - Fetch graph at each epoch

- Model graphs/grammars from cluster

- Fine-grained Graph
Architecture

Master Node

Model Aggregator

Worker Nodes

Audit Module

Worker Node

Prov. Graph

Fetch graph at next epoch

Winnower Agent

Aggregated Model

Model updates

Fine-grained Graph

Graph Abstraction

Abstracted Graph

Graph Induction

Model Graph
Architecture

Model Aggregator

Query part of Provenance graph

High fidelity Provenance graph

Master Node

Worker Nodes

Worker Node

Audit Module

Winnower Agent
Provenance Graph Abstraction

- Graph Induction process builds a model/grammar that concisely describe the whole graph
- However, instance-specific fields frustrate any attempts to build a generic application behaviour model
Provenance Graph Abstraction

- Provenance graph vertices have well defined fields
  - E.g. $pid:1234$, $FilePath:/etc/ld.so$
- Defined rules manually that remove or generalize these fields
Provenance Graph Induction

- Deterministic Finite Automata (DFA) Learning to generate grammar
  - Encodes the causality in generated models
- In DFA learning the present state of a vertex includes the path taken to reach the vertex (provenance ancestry)
  - Winnower extends it to remember descendants (provenance progeny)

- State of each vertex consist of three items:
  1. Label
  2. Provenance ancestry
  3. Provenance progeny
Provenance Graph Induction

- Finds repetitive patterns using standard implicit and explicit state merging algorithm
- Implicit state merging combines two subgraphs if states of each vertex are same in both subgraphs

![Graph Induction Diagram]
Explicit State Merging

- At high-level explicit state merging
  - Picks two nodes and make their states same
  - Check if subgraph can be merged implicitly
- Consider a chained map reduce job

Graph Grammar

```plaintext
:= S
S := A
S := T | V
T := A -> X | A -> Y
X := B -> W
Y := C -> W
W := D | D -> S
A := data
B := java mapper
C := java reducer
D := java
```
Provenance Graph Induction

- Consider a graph with a malicious activity
- Malicious behavior is visible in the final model
Evaluation Setup

• Setup
  • 1 VM as master node, 4 VMs as worker nodes
  • SPADE and Docker Swarm
  • Epoch size 50 sec

• Metrics
  • Storage Overhead
  • Computational Cost
  • Effectiveness
Storage Overhead on Master Node

98.7% decrease

LOG SIZE IN MB

HTTPD

ProFTPD

MySQL

98.7% decrease
Storage Reduction on Master Node

- Apache Webserver with moderate workload
- Note the log scale on y-axis

7z compression is not suitable:
- No global view of cluster
- Oblivious to previous batch
Evaluation: Computation Cost

• Average time spent in induction and membership test at each epoch
Case Study: Ransomware Attack

- Attacker exploits Redis database server vulnerability version < 3.2
- Vulnerability allows attacker to change SSH key and log in as Root
- Attacker deletes the database and left a note using vim to send bitcoins get database back
Traditional Graph of Attack

- 10 instances of redis running in the cluster
- ~80k vertices and ~83K edges with 161 MB size
- Part of provenance graph shown below
Winnower Generated Provenance graph

- 54 vertices and 68 edges with 0.7 MB size
- Part of graph is shown below:
Winnower Generated Provenance graph

• What happens if we attack all the nodes in the cluster
Conclusion

• Winnower is the first practical system for provenance-based auditing of clusters at scale with low overhead

• Winnower significantly improves attack identification and investigation in a large cluster
Thank you for your time.

whassan3@illinois.edu
Backup Slides
Threat model

- **Assumptions**
  - Winnower only tracks user-space attacks i.e. trusts the OS
  - Log integrity is maintained

- **Attack surface**
  - Distributed application replicated on Worker nodes

- **Attacker’ motive**
  - Gain control over worker node by exploiting a software vulnerability in the distributed application
Online Learning

Abstracted graph → DFA Learning → Graph Grammar → Model Aggregator → Aggregated Model