Provenance-based Intrusion Detection

Thomas Pasquier University of Bristol https://tfjmp.org 12/11/2020

Talk loosely based on following publications

- Han et al. "SIGL: Securing Software Installations Through Deep Graph Learning", USENIX Security 2021
- Han et al. "UNICORN: Revisiting Host-Based Intrusion Detection in the Age of Data Provenance", NDSS 2020
- Pasquier et al. "Runtime Analysis of Whole-System Provenance", ACM CCS 2018
- Pasquier et al. "Practical Whole-System Provenance Capture", ACM SoCC 2017

System Calls







System Calls Identify abnormal patterns Hidden among benign actions Masquerading as benign action

System Calls

[...] Identify abnormal patterns Hidden among benign actions Masquerading as benign action [...] Over a long period of time

What is provenance?

What is provenance?

- From the French "provenir" meaning "coming from"
- Formal set of documents describing the origin of an art piece
- Sequence of
 - Formal ownership
 - Custody
 - Places of storage
- Used for authentication



What is data-provenance?

- Represent interactions between objects of different types
 - Data-items (entities)
 - Processing (activities)
 - Individuals and Organisations (agents)
- Represented as a **directed acyclic graph** (think information flows)
- Edges represent interactions between objects' states as dependencies
- It is a representation of history of a system execution
 - Immutable (unless it's 1984)
 - No dependency to the future

How is this useful?

Provenance-based intrusion detection

 Intuition: provenance graph exposes causality relationships between events



Provenance-based intrusion detection

 Intuition: provenance graph exposes causality relationships between events



Provenance-based intrusion detection

- Related events are connected even across long period of time



How to perform detection?

Assumptions (and limitations)

Runtime detection

- We target environment with minimal human intervention
 - relatively consistent behaviour
 - e.g. web servers, CI pipelines etc...
- Build a model of system behaviour (unsupervised training)
 - in a controlled environment
 - from a representative workload (this is hard!)
- Detect deviation from the model
- Several approaches being explored...



 Han et al. "UNICORN: Runtime Provenance-Based Detector for Advanced Persistent Threats", NDSS 2020



1) Graph streamed in, converted to histogram, labelled using (modified) **struct2vec**



2) At regular interval, histogram converted to a fixed size vector using **similarity preserving graph sketching**



3) Feature vectors are **clustered**



4) Cluster forms "meta-state", transitions are modelledIn deployment, anomaly detected via clustering and "meta-state" model

Relatively simple

- Labelled directed acyclic graph
 - node/edge types
 - security context (when available)
- Modification and combination of existing algorithms
 - struct2vec
 - similarity preserving hashing
 - clustering
- Right combination + domain knowledge

Some insights from this work

We can build practical provenance-based IDSs

- We can detect intrusion out of graph structure with little metadata
 - Vertex type (thread, file, socket etc...)
 - Edge type (read, write, connect etc...)
- Processing speed
 - Current prototype
 - Data generation speed < processing speed!</p>

Proper evaluation is hard!

- Dataset are hard to generate

- What is a good quality dataset?
- Hard to compare across papers, a lot is not available
 - Experiments (i.e. attacks)
 - Capture Mechanisms
 - Analysis pipelines
- Leads to unsatisfactory evaluation
 - I may be able to compare to similar techniques (may reuse dataset)
 - ... very hard for unrelated one (i.e. ingest different data type)
- Adversarial ML?

Identifying threats: explainability is a problem

- There is a problem within the last batch of X graph elements
 - 2,000 in previous figures
- Good luck finding out what went wrong
- Provenance forensic is an active field of research
 - Promising work out of the DARPA programme
- ... but could we do better during detection?

Ongoing projects

Towards more interpretable provenance-based IDSs

- PhD student project (Xueyuan "Michael" Han)
- Collaborators
 - Harvard University
 - UBC
 - NEC Labs America
- Deep graph learning techniques
- Precisely identifying attacks within a provenance-graph
- Generating actionable reports

A framework for Provenance-based forensics

- PhD student project (**Priyanka Badva**)
- Collaborators
 - SRI International
- Provenance graphs are large and complex (several millions nodes)
- Designing tools and techniques to identify/explain attacks
- Working with my colleague **Ryan**

Distributed IDS

- Edge network
- Collaboration with Toshiba (£4M)
- Exploring distributed learning
 - Poisoning
 - Mechanism
 - Etc.
- Large testbed planned (work starting January)
- Hiring 2 postdocs at Bristol
- Money available for an intern short term (+-covid)

Kernel partitioning

- PhD student project (Soo Yee Lim)
- Collaborators
 - HP Labs Bristol
 - Royal Holloway, University of London
 - University of Otago
- Leveraging CHERI/ARM Morello hardware
 - Hardware capabilities
- Implement kernel partitioning in the Linux OS

Thank you! Questions?

https://tfjmp.org thomas.pasquier@bristol.ac.uk

How to evaluate?

Comparison state of the art

Experiment	Dataset	# of Graphs	Avg. V	Avg. E	Preprocessed Data Size (GiB)
	YouTube	100	8,292	113,229	0.3
C	Gmail	100	6,827	37,382	0.1
	Download	100	8,831	310,814	1
Streamspor	VGame	100	8,637	112,958	0.4
	CNN	100	8,990	294,903	0.9
	Attack	100	8,891	28,423	0.1

TABLE I: Characteristics of the StreamSpot dataset. The dataset is publicly available only in a preprocessed format.

Experiment	Precision	Recall	Accuracy	F-Score
StreamSpot (baseline)	0.74	N/A	0.66	N/A
R = 1	0.51	1.0	0.60	0.68
R = 3	0.98	0.93	0.96	0.94

TABLE II: Comparison to StreamSpot on the StreamSpot dataset. We estimate StreamSpot's average accuracy and precision from the figure included in the paper [83], which does not report exact values. They did not report recall or F-score.

Manzoor et al. "Fast memory-efficient anomaly detection in streaming heterogeneous graphs" ACM KDD, 2016.

R -> neighborhood size for struct2vec algorithm

Evaluation with DARPA datasets

Experiment	Dataset	# of Graphs	Avg. V	Avg. E	Raw Data Size (GiB)
DARPA	Benign	66	59,983	4,811,836	271
CADETS	Attack	8	386,548	5,160,963	38
DARPA	Benign	43	2,309	4,199,309	441
ClearScope	Attack	51	11,769	4,273,003	432
DARPA	Benign	2	19,461	1,913,202	4
THEIA	Attack	25	275,822	4,073,621	85

Experiment	Precision	Recall	Accuracy	F-Score
DARPA CADETS	0.98	1.0	0.99	0.99
DARPA ClearScope	0.98	1.0	0.98	0.99
DARPA THEIA	1.0	1.0	1.0	1.0
	• • 1	1. 6.1		

TABLE V: Experimental results of the DARPA datasets.

TABLE IV: Characteristics of graph datasets used in the DARPA experiments.

Evaluation with DARPA datasets

Experiment	Dataset	# of Graphs	Avg. V	Avg. E	Raw Data Size (GiB)
DARPA	Benign	66	59,983	4,811,836	271
CADETS	Attack	8	386,548	5,160,963	38
DARPA	Benign	43	2,309	4,199,309	441
ClearScope	Attack	51	11,769	4,273,003	432
DARPA	Benign	2	19,461	1,913,202	4
THEIA	Attack	25	275,822	4,073,621	85

Experiment	Precision	Recall	Accuracy	F-Score
DARPA CADETS	0.98	1.0	0.99	0.99
DARPA ClearScope	0.98	1.0	0.98	0.99
DARPA THEIA	1.0	1.0	1.0	1.0
		1. 0.1	DIDDI 1	

TABLE V: Experimental results of the DARPA datasets.

TABLE IV: Characteristics of graph datasets used in the DARPA experiments.

SUCH GOOD RESULTS ARE NOT NORMAL

Building our own dataset

Experiment	Dataset	# of Graphs	Avg. V	Avg. E	Raw Data Size (GiB)
	Benign	125	265,424	975,226	64
SC-1	Attack	25	257,156	957,968	12
	Benign	125	238,338	911,153	59
SC-2	Attack	25	243,658	949,887	12

Experiment	Precision	Recall	Accuracy	F-Score
SC-1	0.85	0.96	0.90	0.90
SC-2	0.75	0.80	0.77	0.78

TABLE VIII: Experimental results of the supply-chain APT attack scenarios.

TABLE VI: Characteristics of the datasets used in the supply-chain APT attack experiments.

Attack designed to look similar to background activity

Building our own dataset

Experiment	Dataset	# of Graphs	Avg. V	Avg. E	Raw Data Size (GiB)
	Benign	125	265,424	975,226	64
SC-1	Attack	25	257,156	957,968	12
	Benign	125	238,338	911,153	59
SC-2	Attack	25	243,658	949,887	12

Experiment	Precision	Recall	Accuracy	F-Score
SC-1	0.85	0.96	0.90	0.90
SC-2	0.75	0.80	0.77	0.78

TABLE VIII: Experimental results of the supply-chain APT attack scenarios.

TABLE VI: Characteristics of the datasets used in the supply-chain APT attack experiments.

Attack designed to look similar to background activity

Is that enough?

Runtime performance

Fig. 4: Total number of processed edges over time (in seconds) in the SC-1 experimental workload with varying batch sizes (Fig. 4(a)), sketch sizes (Fig. 4(b)), hop counts (Fig. 4(c)), and intervals of sketch generation (Fig. 4(d)). Dashed blue line represents the speed of graph edges streamed into UNICORN for analysis. Triangle maroon baseline has the same configurations as those used in our experiments and indicates the values of the controlled parameters (that remain constant) in each figure.



R = 1

560

Runtime performance



Runtime performance

Memory usage: ~500MB CPU usage 15% on 1 core



Fig. 4: Total number of processed edges over time (in seconds) in the SC-1 experimental workload with varying batch sizes (Fig. 4(a)), sketch sizes (Fig. 4(b)),