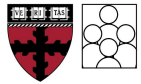
Provenance-based Intrusion Detection: Opportunities and Challenges

Xueyuan “Michael” Han*, **Thomas Pasquier***+, Margo Seltzer*

*Harvard University, +University of Cambridge

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

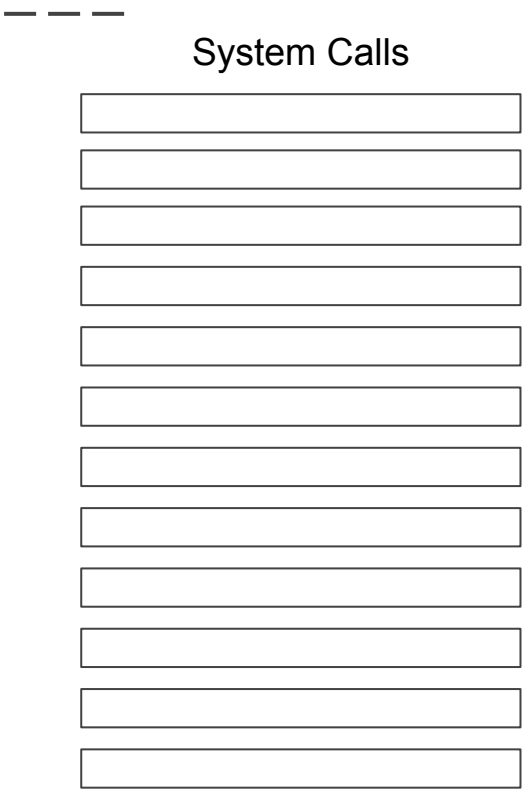


CRCS Center for Research on Computation and Society
at Harvard John A. Paulson School of Engineering and Applied Sciences

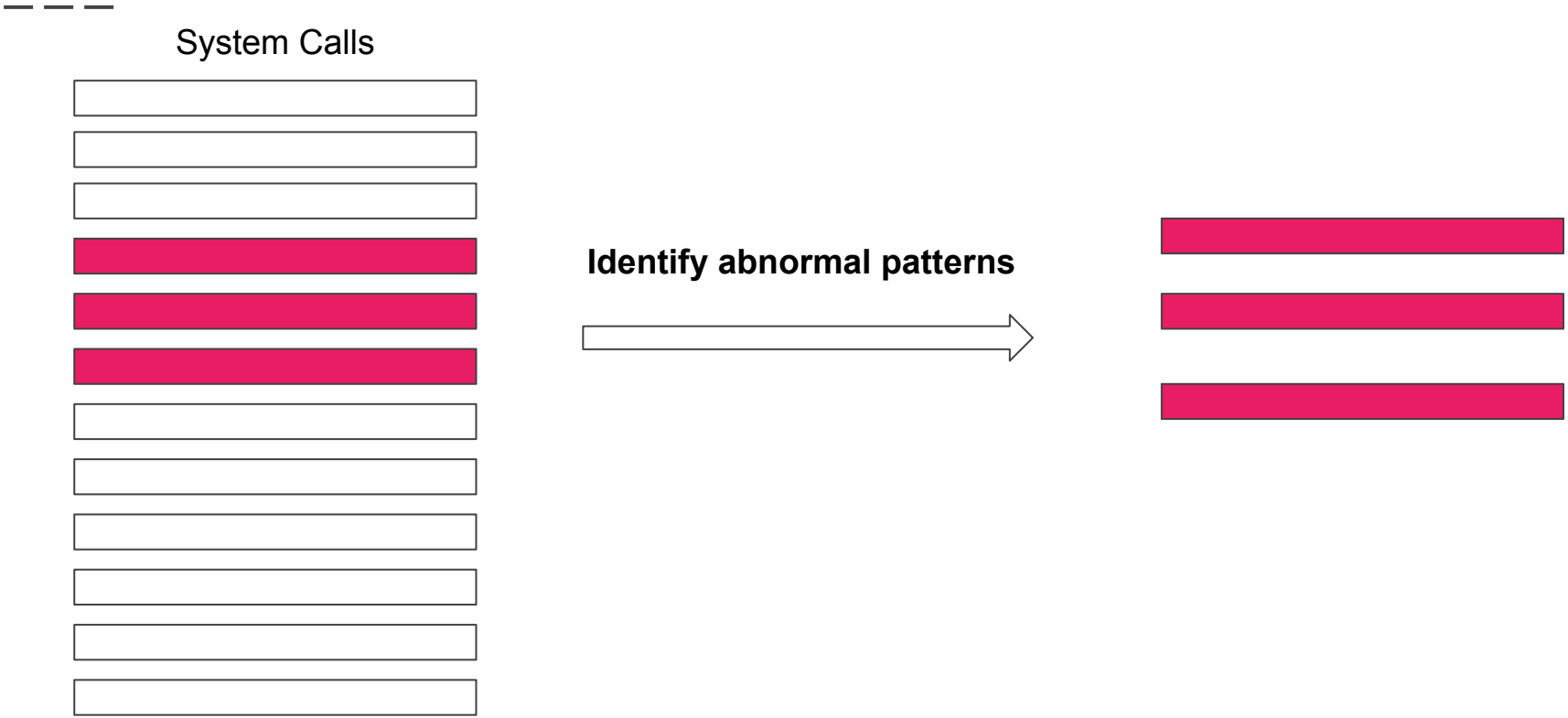
Discussion paper

- (quick) problem description
- (some) interesting challenges

System call based intrusion detection

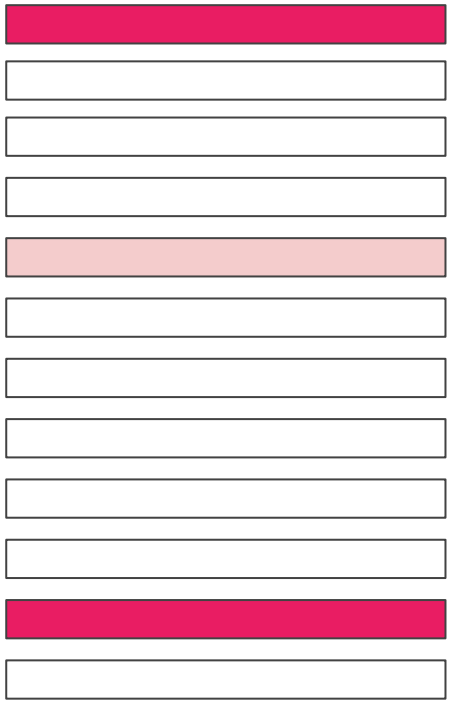


System call based intrusion detection



System call based intrusion detection

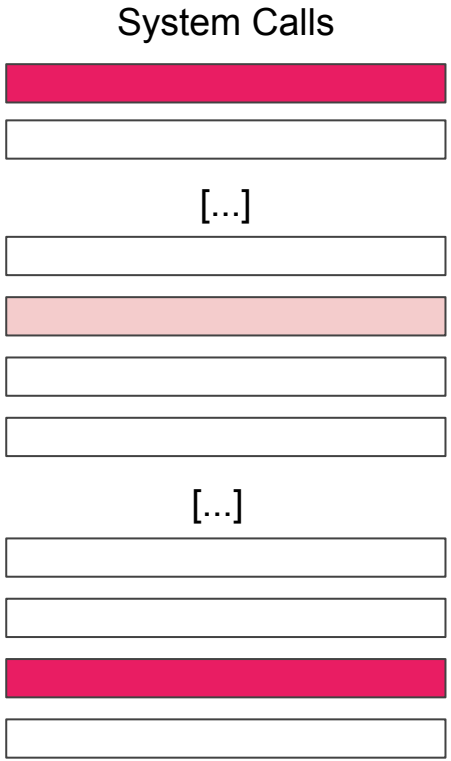
System call based intrusion detection



Identify abnormal patterns

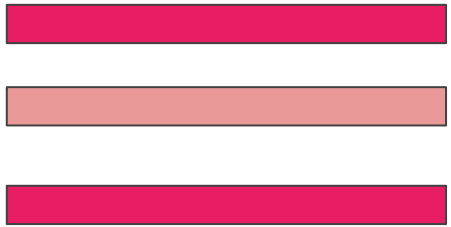
Hidden among benign actions
Masquerading as bening action

System call based intrusion detection



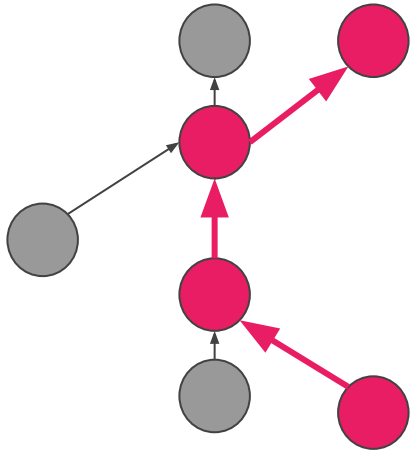
Identify abnormal patterns

Hidden among benign actions
Masquerading as bening action
Over a long period of time



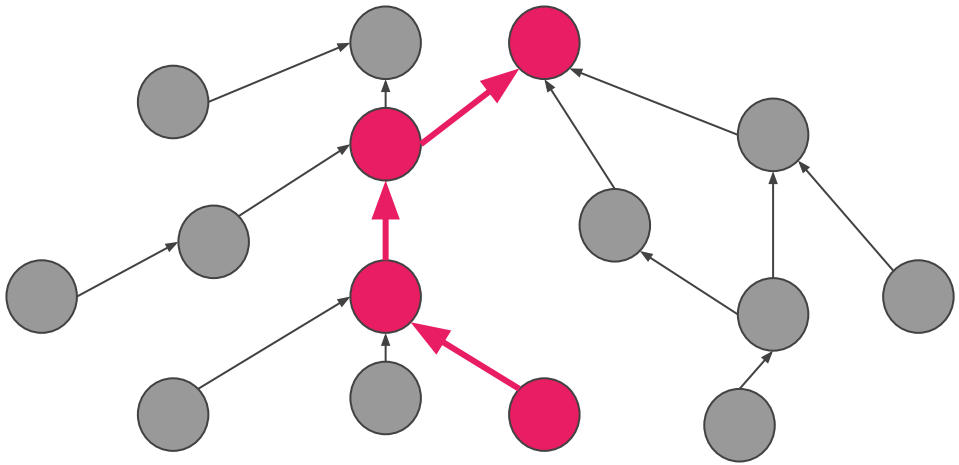
Provenance-based intrusion detection

- **Intuition**: provenance graph **exposes causality relationships** between events

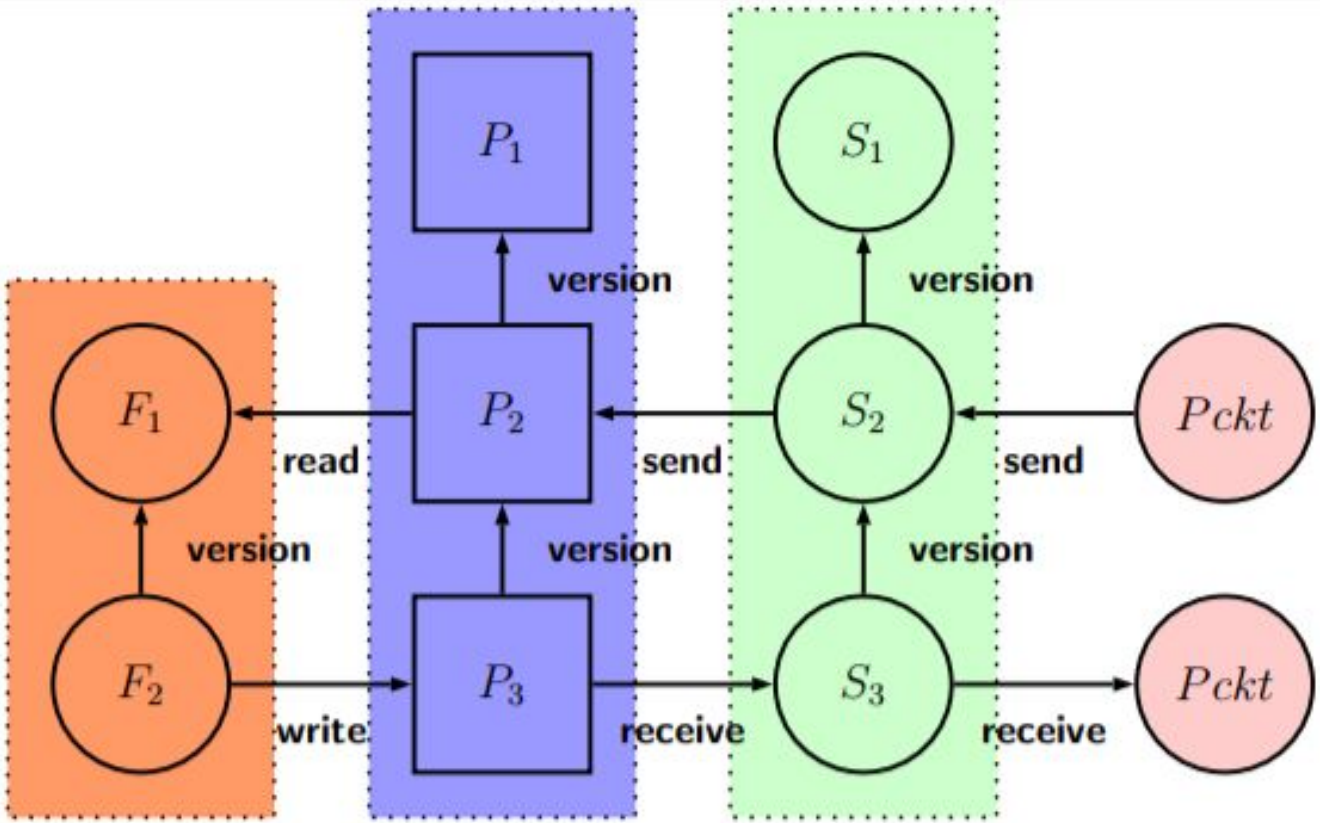


Provenance-based intrusion detection

- Intuition: provenance graph **exposes causality relationships** between events



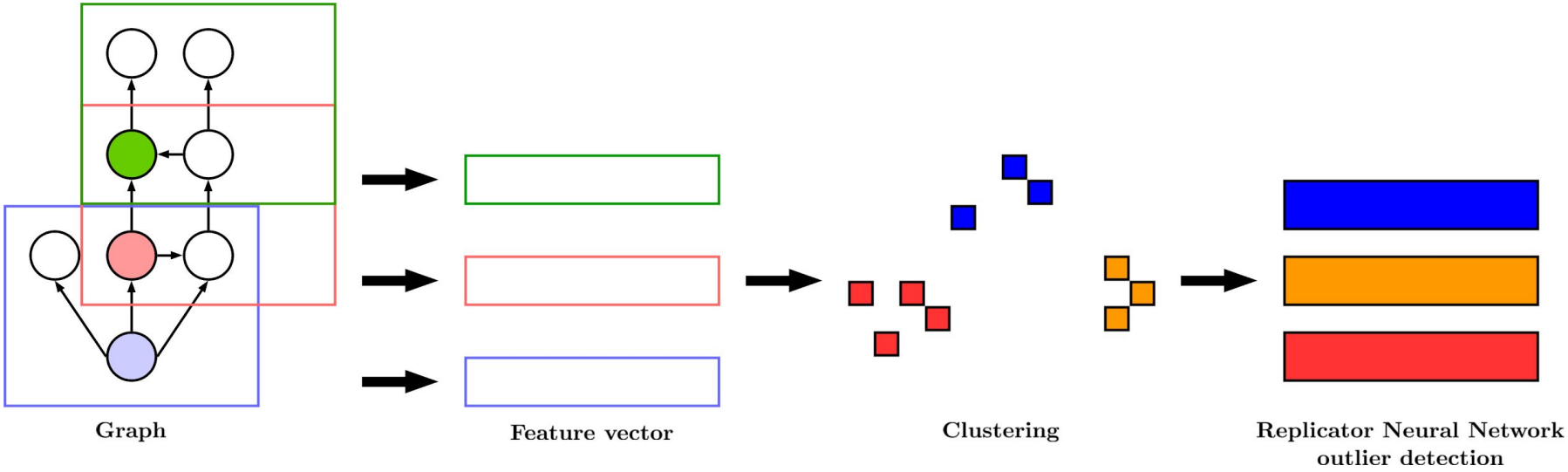
Example: what provenance?



Example: How?

- Principles first introduced in Han et al.(USENIX HotCloud 2017)
- In a cloud computing context (self-contained VMs)
- Capture uncompromised behaviour
  - in a controlled environment
  - from a representative workload
- Build a model of system behaviour
  - Unsupervised learning
  - Neural Network, Statistical Model, etc.
- Detect deviations from the model
- Several approaches explored
  - e.g., see Ghita talk on Friday

Example: How?


Graph
Feature vector
Clustering
Replicator Neural Network
outlier detection

(some) challenges

- Capture
- Analysis
- Take away

Capture: Provenance, but what provenance?

- **Whole-system** provenance (defined quite broadly)
- A number of implementations
  - e.g., SPADE, HiFi, LPM, CamFlow, CADETS, etc.
  - similarity in capture mechanisms…
  - … yet very different semantics.
- This has been driven by systems research
  - … but maybe it is time for some formalism?
  - early exploration ProvMark to compare the produced provenance.
- What are we trying to represent?
- How do we represent it?
- More than syntax! (see James/Bertram talks)

Capture: open source (and maintained) capture mechanism

SPADE: http://spade.csl.sri.com

CADETS: https://github.com/cadets

CamFlow: http://camflow.org

Capture: is the provenance correct/complete/accurate?

- Need to solve the previous questions!
- Proof?
  - Operating systems are very complex
- Smaller steps?
  - CamFlow: we did some static analysis to show what is captured and how it is represented…
  - ProvMark: runtime equivalent (multiple systems) [talk to James]
- Nothing quite satisfactory yet (need formalisation)
- … security/ threat model

Analysis: machine learning on structured graphs

- A much harder problem than on flat data pattern discovery
  - Nodes/edges have different numbers of attributes
  - Need to learn various relations (i.e., graph structures) among them
  - **But most ML algorithms learn from fixed-length, real-valued vectors**
  - What information matter? How to represent it?
- Online deployment requires learning and detection in a streaming fashion
  - Analysing the entire graph is impossible
    - **What's the trade-off?**
  - Complexity vs Performance
  - Complexity vs Interpretability

Analysis: comparing results

- Abnormal patterns are not known
  - Machine Learning
  - **Unsupervised learning**
- What model, based on what properties?
  - Neural networks
  - Statical models, etc…
  - What assumptions made about the provenance?
- What dataset?
  - i.e., graph representing “uncompromised” executions
  - … and “similar” compromised executions
  - Where do we get the datasets?
  - DARPA TC? Unicorn effort(https://github.com/crimson-unicorn/dataset)
  - Towards a community effort?

Analysis: constraints

- Runtime vs Post-mortem
- Detecting vs Explaining
  - Frappuccino (HotCloud 2017 Han et al.)
  - Backtracking intrusions (SOSP 2003 King and Chen)
- More complex algorithms **may** do better…
- …but they may not scale
- Systems engineering constraints
  - Need to work on real data
  - On real sized problems
  - There is no solution if the idea cannot get deployed
- **Graph structure/properties matters**

Take away

- Interesting, but complex problem
- Community effort to improve reproducibility/comparison
- Provenance “solved” problems need to be re-explored
  - different scale
  - different threat model
  - different performant constraints
  - **system engineering and security matters**
- Need a venue to discuss engineering/security problems

Thank you!

tfjmp2@cam.ac.uk
http://tfjmp.org/
hanx@g.harvard.edu
https://scholar.harvard.edu/han

Questions?

http://camflow.org