Intrusion Detection in Structured Event Logs Using Statistical Tools

Corentin Larroche

corentin.larroche@ssi.gouv.fr

Agence Nationale de la Sécurité des Systèmes d'Information

LTCI, Télécom Paris, IP Paris

November 19th, 2021

Agenda

1. Events, logs, and intrusions
2. Anomalous user detection using numeric features
3. Anomalous user detection using authentication graphs
4. Anomaly detection for user-host access matrices
5. Anomaly detection for higher-order interactions
6. Conclusion

Agenda

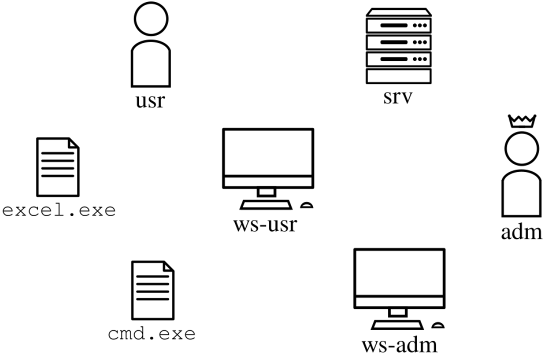
Computer network monitoring and intrusion detection

Monitoring activity inside a computer network

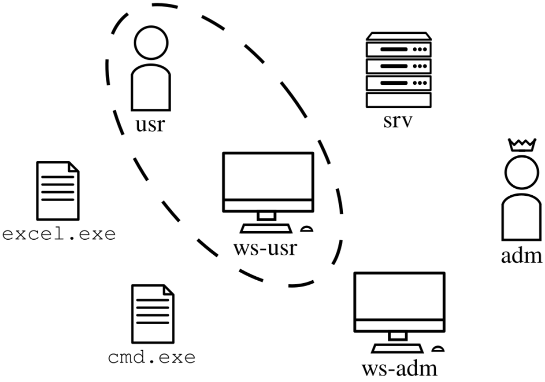
- Various data sources: NetFlows, system logs, endpoint protection software, etc.
- Huge data streams, processed in either an online or batch setting
- Idea: if an intrusion occurs, it should leave a trace in these logs

Remark: we will not discuss the (many) operational problems which occur when operating such a logging system.

Event logs: entities and interactions



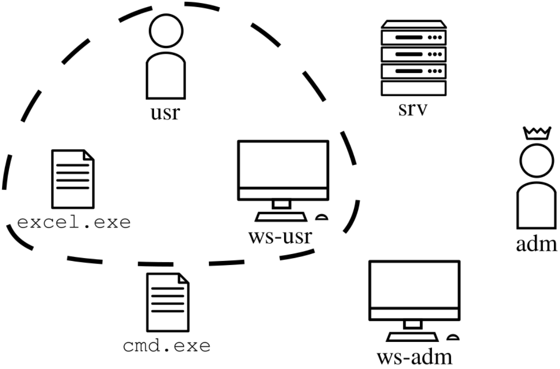
Event logs: entities and interactions



Event logs:

- Interactions between entities

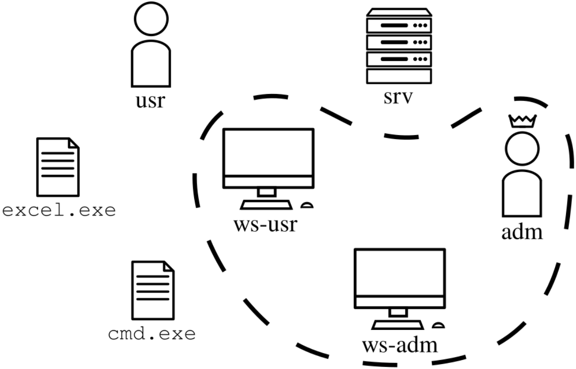
Event logs: entities and interactions



Event logs:

- Interactions between entities
- Various event types

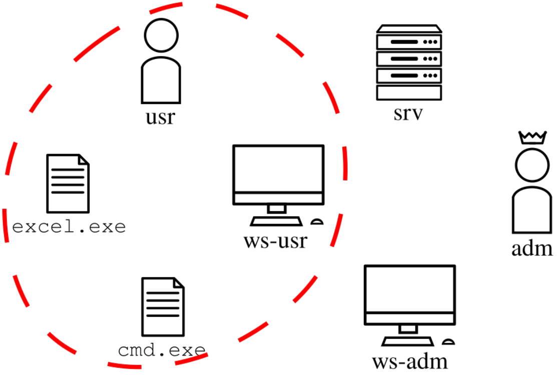
Event logs: entities and interactions



Event logs:

- Interactions between entities
- Various event types
- Complex association patterns

Event logs: entities and interactions



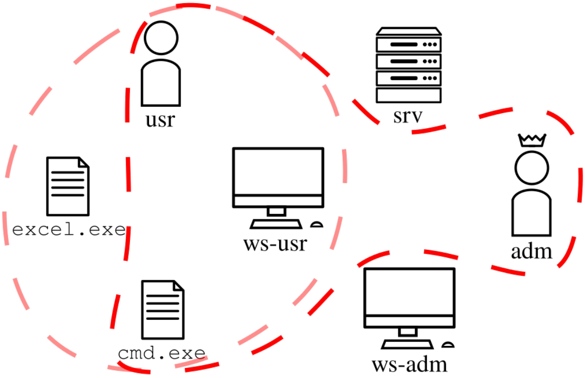
Event logs:

- Interactions between entities
- Various event types
- Complex association patterns

Malicious activity:

- Unusual events

Event logs: entities and interactions

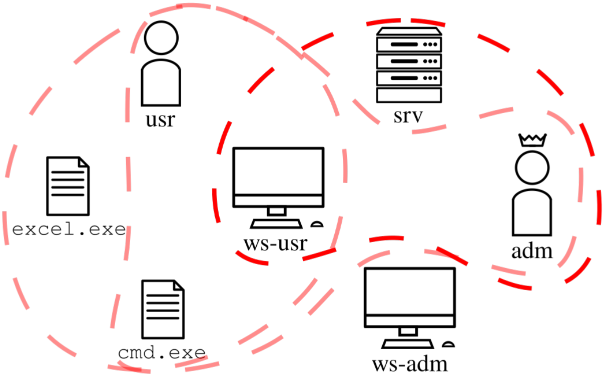


Event logs:

- Interactions between entities
- Various event types
- Complex association patterns

Malicious activity:

- Unusual events
- Intricate action sequences



Event logs:

- Interactions between entities
- Various event types
- Complex association patterns

Malicious activity:

- Unusual events
- Intricate action sequences
- Involving shared entities

Event logs and intrusion detection: definitions

Definition

An **event** is the conjunction of a **timestamp**, an **event type**, a set of **involved entities** and some **optional additional information**.

Event logs and intrusion detection: definitions

Definition

An **event** is the conjunction of a **timestamp**, an **event type**, a set of **involved entities** and some **optional additional information**.

Problem statement

Given a sequence of events, intrusion detection consists in **finding a subset** of this sequence corresponding to malicious activity. Malicious events (or event sets) are assumed to be **scarce**, **distinguishable** from benign activity and involving some **shared entities**.

Event logs and intrusion detection: definitions

Definition

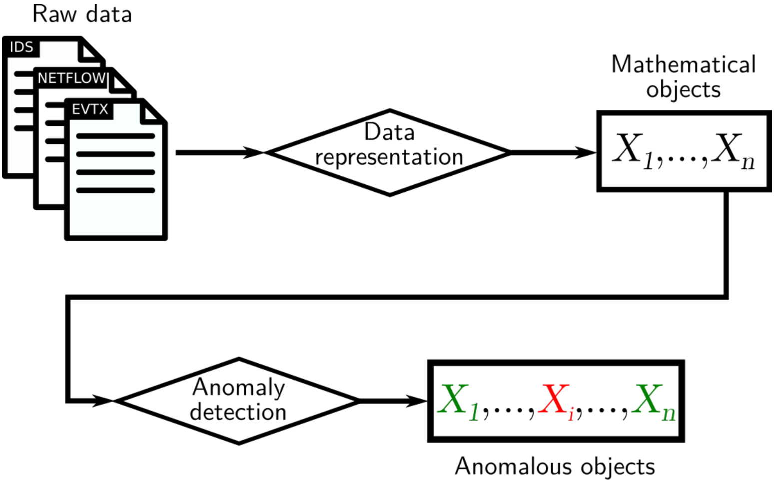
An **event** is the conjunction of a **timestamp**, an **event type**, a set of **involved entities** and some **optional additional information**.

Problem statement

Given a sequence of events, intrusion detection consists in **finding a subset** of this sequence corresponding to malicious activity. Malicious events (or event sets) are assumed to be **scarce**, **distinguishable** from benign activity and involving some **shared entities**.

- Abstract definition meant to encompass multiple concrete instances
- Designing a detection algorithm requires a more practical representation

A (simple) processing pipeline



The LANL dataset

- "Comprehensive, Multi-Source Cyber-Security Events" dataset[1] released by the Los Alamos National Laboratory
- 58 days of activity in a large network ($\sim$ 12 000 users, 17 000 hosts) recorded in several types of event logs
- Red team exercise with labelled authentication events
- Here, we consider a subset of the data:
  - Logon events only
  - No computer and built-in accounts
  - First 8 days for training, next 5 days for testing

[1] https://csr.lanl.gov/data/cyber1/

The LANL dataset – Fields of a logon event

68,U1167@DOM1,U1167@DOM1,C473,C529,Kerberos,Network,LogOn,Success

① ② ③ ④ ⑤ ⑥ ⑦ ⑧ ⑨

(1) Timestamp
(2) Source user: usually irrelevant, but can be useful for logons with explicit credentials (e.g. `runas`)
(3) Destination user: account under which the new session runs
(4) Source host
(5) Destination host (can be the same as the source)
(6) Authentication Package (AP) used to verify the user credentials
(7) Logon type (usual ones: Interactive, Network, RemoteInteractive)
(8) Event type (always LogOn here)
(9) Authentication status (success or failure)

The LANL dataset – Descriptive statistics

A brief description of the reduced dataset:

	Train	Test
#Events (total)	16 623 950	10 488 336
#Events (malicious)	50	483
#Users (total)	12 123	10 700
#Users (compromised)	7	71
#Hosts (total)	12 257	11 928
#Hosts (compromised)	27	240

The LANL dataset – Red team activity

Main goal: lateral movement detection.

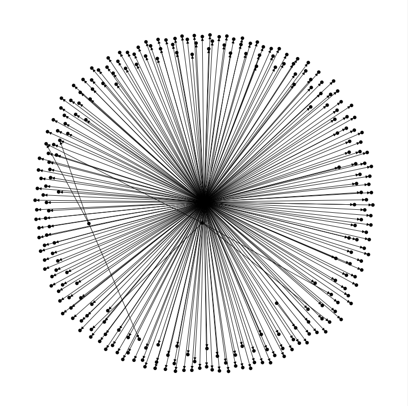


Figure: Red team events as a host-host directed graph.

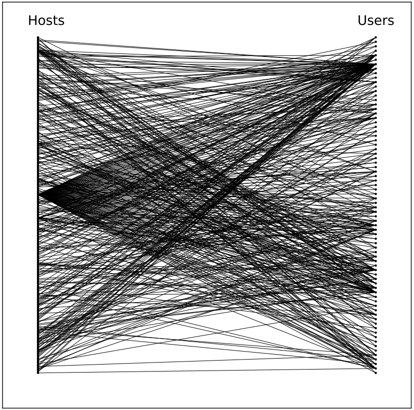


Figure: Red team events as a user-host bipartite graph.

Detection methodologies

We're going to look at several methods:

- Aggregate events by user and day, then:
  - Extract a feature vector for each user-day and run standard anomaly detection algorithms
  - Represent each user-day as a graph, then use graph-oriented tools
- Aggregate events by user-host pair, then look for anomalous pairs using matrix factorization tools
- Use higher-order aggregation keys and corresponding models

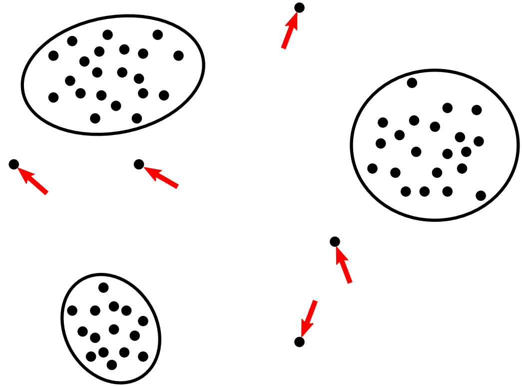
Agenda

1. Events, logs, and intrusions
2. Anomalous user detection using numeric features
3. Anomalous user detection using authentication graphs
4. Anomaly detection for user-host access matrices
5. Anomaly detection for higher-order interactions
6. Conclusion

Motivation: reverting to the standard setting

Anomaly detection – Standard setting

Given n data points $x_1, \ldots, x_n \in \mathbb{R}^d$, estimate the underlying probability density function p and find points x_i such that $p(x_i)$ is low.



Data preprocessing

Instance delimitation and feature extraction

For each user u and day d, let $\mathcal{E}_{u,d}$ be the set of events involving u on day d. Feature extraction maps each event subset $\mathcal{E}_{u,d}$ to a fixed-size vector $x_{u,d} \in \mathbb{R}^d$.

Which features should we use?

- Essentially count-based: number events, number of visited hosts...
- Some fields can be used as filters: AP, logon type

After this preprocessing step, standard anomaly detection algorithms can be used to detect anomalous user-days.

Demo time

`user_features.ipynb`

Agenda

Motivation: building a richer representation

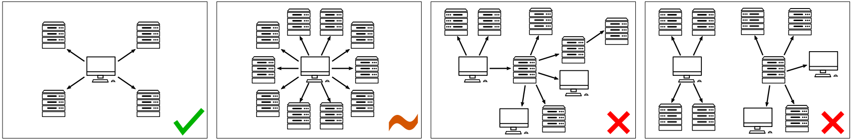
- User-based aggregation and numeric feature extraction seem to give decent results
- However, we would like to do better
- Natural lead for improvement: data representation
  - Count-based features capture a small fraction of the information contained in the logs
  - In particular, the global patterns formed by a user's authentications are mostly lost
  - A graph-based representation could be more adequate

User authentication graphs

Authentication graph

For each user u and day d, the authentication graph $\mathcal{G}_{u,d} = (\mathcal{V}_{u,d}, \mathcal{A}_{u,d}, w_{u,d})$ is a weighted directed graph, where

- $\mathcal{V}_{u,d}$ = set of hosts from or to which u has authenticated on day d
- $(h_1, h_2) \in \mathcal{A}_{u,d}$ if u has authenticated from h_1 to h_2 on day d
- $w_{u,d}(h_1, h_2)$ = number of times u has authenticated from h_1 to h_2 on day d



Demo time

`user_graphs.ipynb`

Agenda

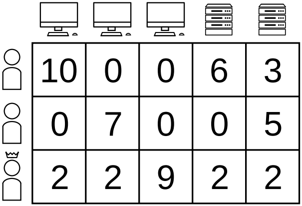
Motivation: analyzing activity at a finer granularity

- Aggregating events by user-day is too coarse
  - Malicious activity drowned in benign events
  - Hardly interpretable predictions
- We need to split the dataset at a finer granularity
- We thus aggregate events by user-host pair
  - Malicious events should be more isolated
- Authentication logs are now represented as a user-host access matrix

User-host access matrices

User-host access matrix

Given n users and m hosts, the user-host access matrix for a given time window is the matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$ whose coefficient at location (i, j) is the number of times user i has authenticated to host j within this time window.



10	0	0	6	3
0	7	0	0	5
2	2	9	2	2

Demo time

`user_host_matrices.ipynb`

Agenda

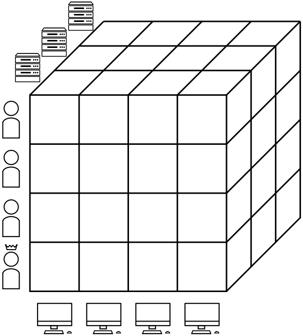
Motivation: including more information in our representation

- Despite higher ratios of red team events for malicious user-host pairs, access matrix modelling does not yield higher detection performance than user-based aggregation
- Possible cause: data representation not rich enough
  - Only the user and destination host are extracted from each event
  - Potentially relevant information is lost: source host, AP, logon type
- Incorporating more information requires a different mathematical framework
  - Natural extension of matrices to higher-order interactions: tensors

Representing categorical datasets as tensors

Authentication tensor

Assume that each authentication event is represented by m categorical variables, with respective arities $n_1, \ldots, n_m$. We then define the tensor $\mathcal{Y} \in \mathbb{R}^{n_1 \times \ldots \times n_m}$, whose coefficient at location $(i_1, \ldots, i_m)$ is the number of events involving entities $(i_1, \ldots, i_m)$.

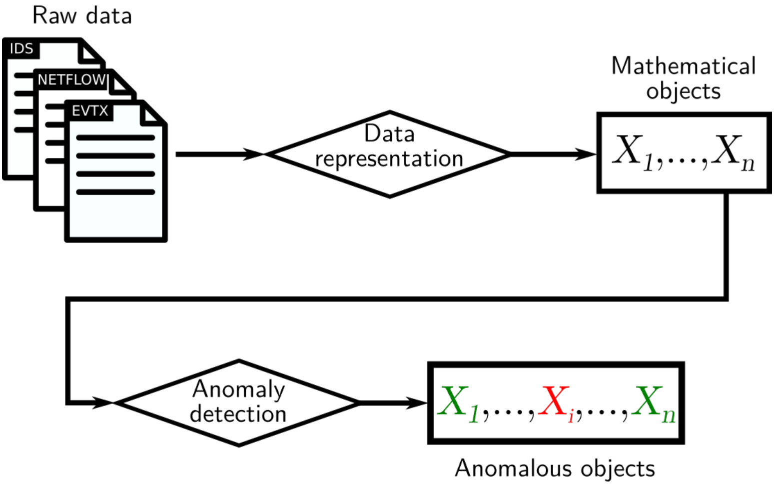


Demo time

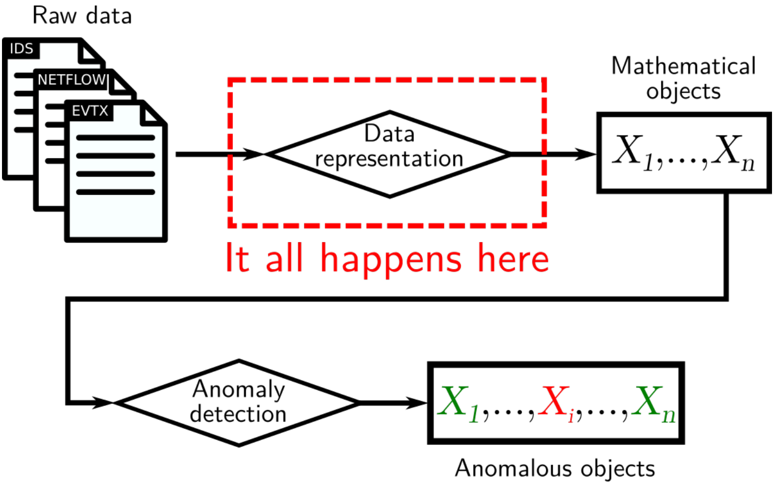
event_tensors.ipynb

Agenda

What have we learned?



What have we learned?



A few insights

- Globally, fine-grained segmentation of the data leads to better detection methodologies
  - More accurate models
  - More interpretable predictions
- Finding the right amount of information to include is hard
  - Domain knowledge can help
  - Extensive evaluation is necessary (but not easy to perform)
- Ideally, several algorithms should be applied sequentially
  - In particular, some postprocessing is required to make anomaly detection more reliable