



AUGUST 6-7, 2025
MANDALAY BAY / LAS VEGAS

FACADE

High-Precision Insider Threat Detection Using Contrastive Learning

Alex Kantchelian, Elie Bursztein, Casper Neo, Ryan
Stevens, Sadegh Momeni, Birkett Huber, Yanis Pavlidis with
the help of many Googlers



Presentation slides:
<https://elie.net/facade>

10 billion+

events processed
annually to protect
Google from insider
threats



Insider attacks threat model



Intentional

attack by a rogue
employee



Unwilling

attack by a deceived or
coerced employee



Accidental

harm by a well
intentioned employee

Example of insider threats

Intentional

access of confidential documents without business justification through access permissions abuse

Unwilling

access made using an employee account compromised by a malware

Accidental

share confidential documents with external party without NDA in good faith



Why detecting insider attacks is hard



Very low incidence

Insider threats incidence events are extremely low volume



Heavily context dependent

Risks depends on user roles and its relation to the resources accessed



Wide attack surface

Insider attackers have a broad access to the enterprise infrastructure via legitimate credentials



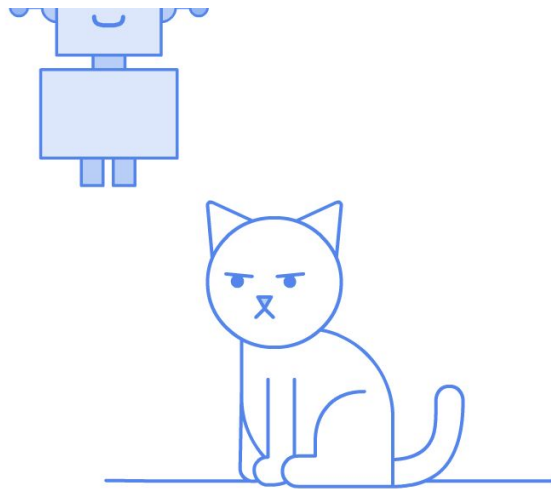
low false alerts

FACADE: A High-Precision Insider Threat Detection Using Deep Contextual Anomaly Detection

Deep
learning
model

User and
resource
aware

How likely is
the access?

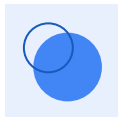


Highly accurate anomaly detection? Really?



Red Team attacks ranked in the top 0.01% of suspicious events and many red team attackers in the top-10 most suspicious users during the attack period, with 10+ millions events ranked by FACADE during that timespan.

Agenda



FACADE Overview



Featurization of Resources and Users



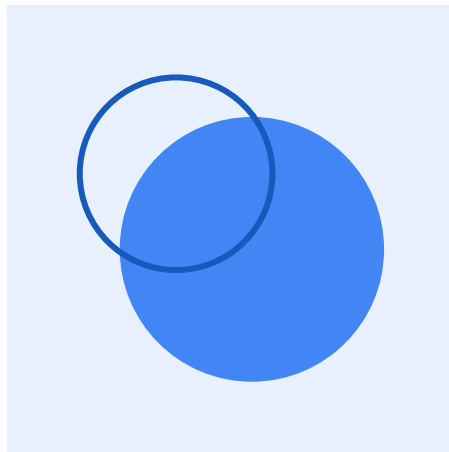
Scoring Arbitrary Time Periods



Finding Insider Attacks with FACADE



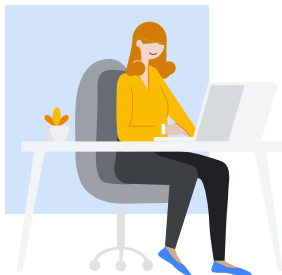
FACADE Overview





Problem formulation

Is it normal for a given user
to access a given resource?



TPU
schematics

Normal pattern

Legitimate user
Hardware division

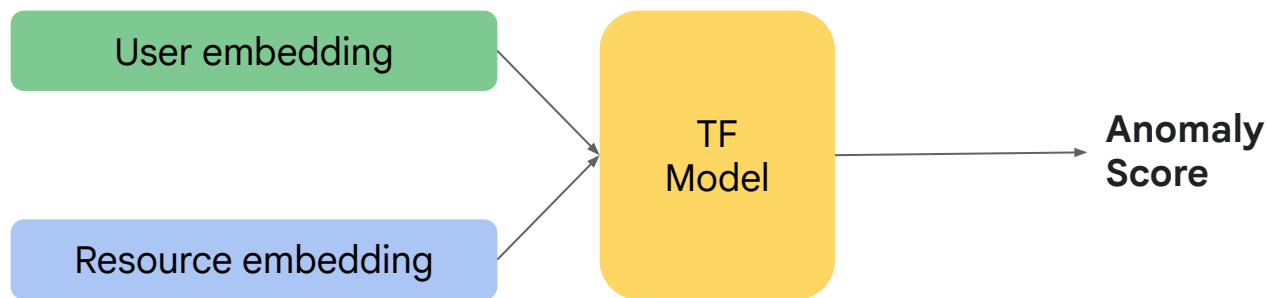


TPU
schematics

Abnormal pattern

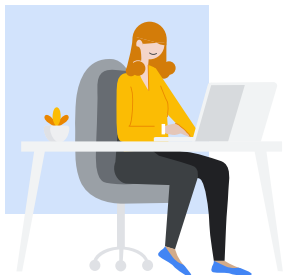
Rogue actor
Ads Sales

FACADE model architecture





How do we **train** such
model **with little to no**
insider attacks examples?



User A
Hardware division

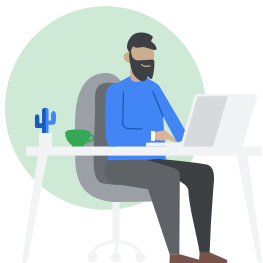


TPU
schematics



User A embedding

TPU doc embedding



User B
Google DeepMind



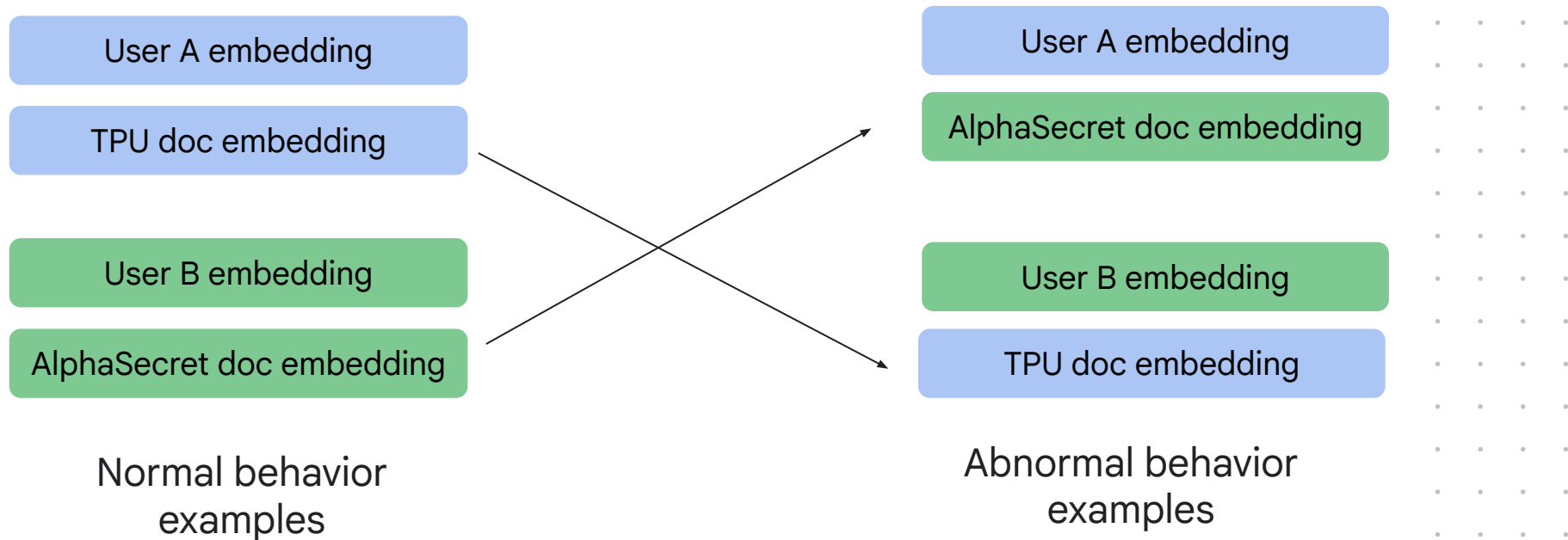
AlphaSecret
Model results



User B embedding

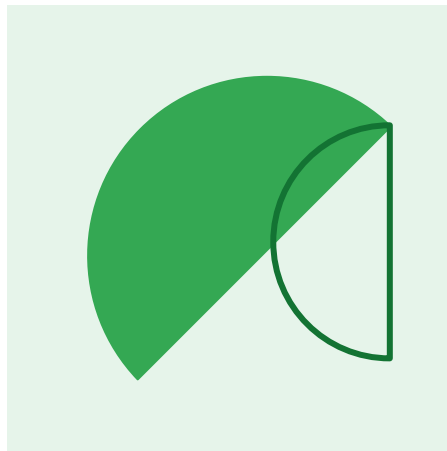
AlphaSecret doc embedding

Unsupervised Training dataset construction



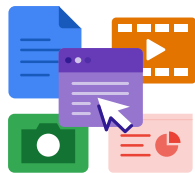


Featurization of Resources and Users



Featurization of Resources

Resource Featurization Challenges



Must handle massive, heterogeneous resources

billions of distinct items

document, spreadsheet, video, data table, RPC endpoint, URL, ...



Content-based features impractical

case-by-case development & maintenance cost
computationally expensive at inference time



Large distribution drift

new resources at inference time *is the norm*, e.g., documents
inherent difficulty in predicting appearance of novel topics in content



How to turn the open
space of resources into a
dense representation
suitable to deep-learning
training?

Intuition



If the following held, could treat resource as categorical feature:
the set of resources is mostly constant
the set of resources is not too large
each resource keeps a stable meaning throughout its existence



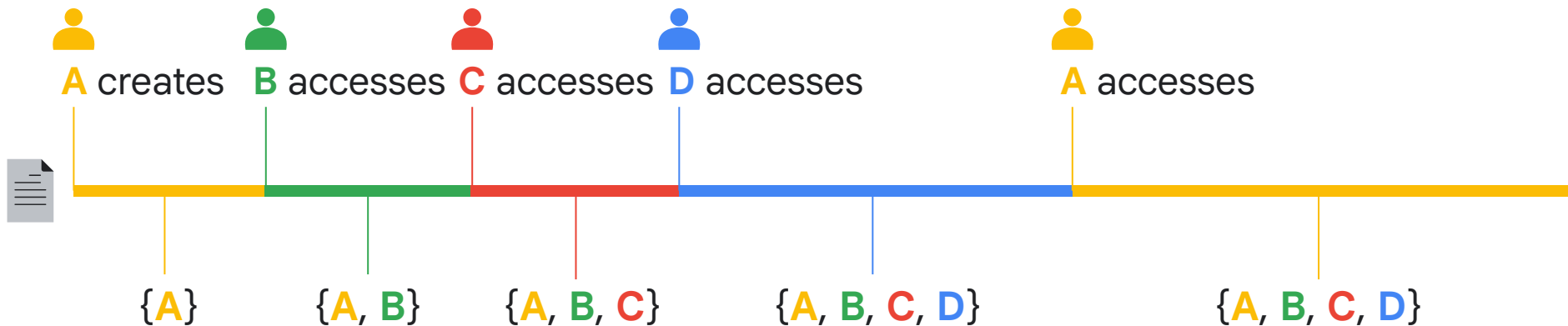
Idea: project resources into a more stable set of opaque identifiers
the set of user ids on a corporate system is a good candidate



History-based Featurization
Bag-of-words of user ids who have previously accessed it

History-based Featurization

Resource Access History



Resource featurizations for given time periods

Handles distribution drift (changing content)

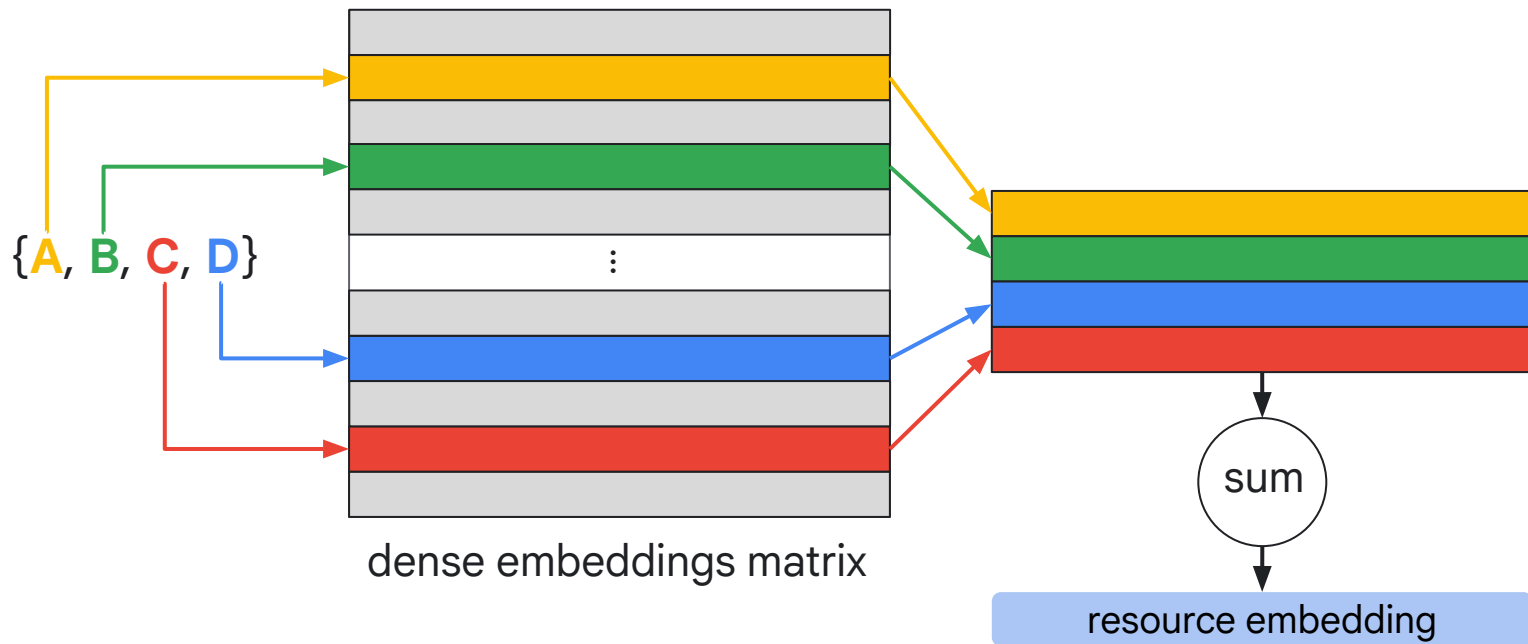
Access Event Input Example

```
{  
  id: "e767..."           # An unique identifier for this access event  
  occurred_at: 1710...       # Timestamp of the event  
  principal: "A"             # The id of the user performing the access  
  type: "doc_access"         # Resource type (doc, db table, hostname, ...)  
  resource_id: "8bca..."    # Resource identifier, e.g., document id  
}
```



**You only need to choose a stable-in-time resource identifier
Facade takes care of the rest (history-based featurization)**

History Set to Dense Vector



Featurization of Users

Two Types of User Attributes



Low cardinality, stable attributes

E.g. Job title (receptionist, software engineer, hardware engineer, etc)

→ Direct categorical featurization



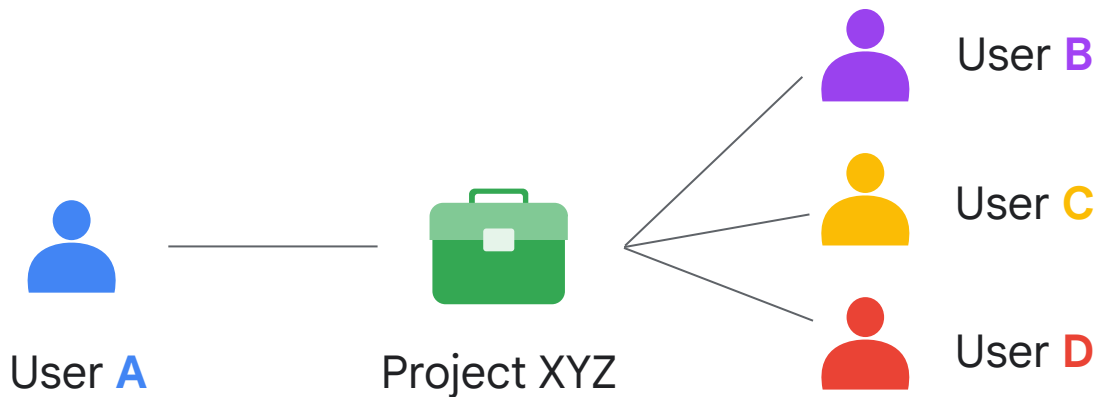
High cardinality, unstable attributes

E.g. team, projects assigned, meetings attended, PRs reviewed, ...

Large distribution drift (re-orgs, new projects, employees, etc)

→ Implicit social network featurization

Implicit Social Network Featurization

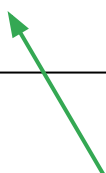


“project” feature for user **A** is bag-of-words {**B**, **C**, **D**}

Any user is featurized by a set of sets of other user ids
one set per attribute type (team, project, department, etc)

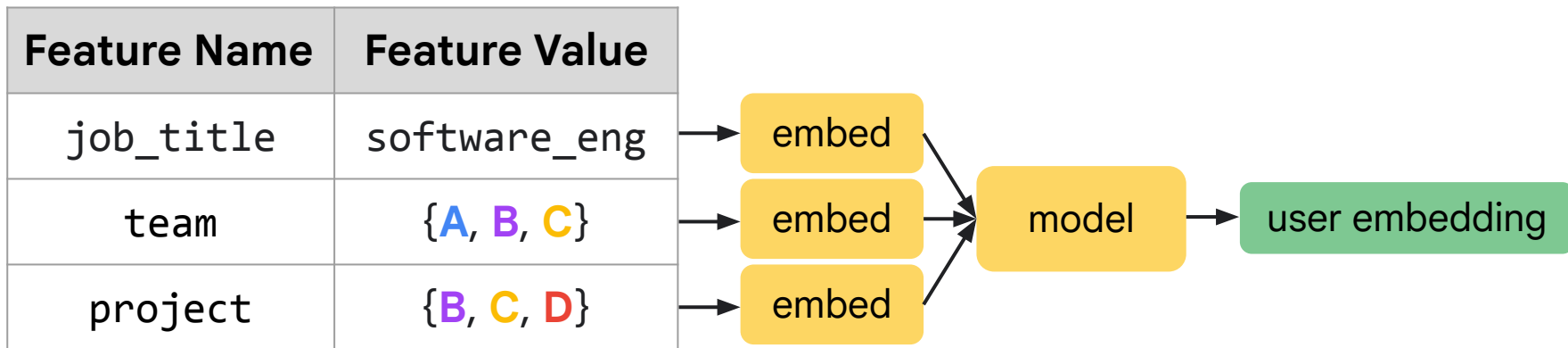
User Context Event Input Example

```
{  
  valid_from: 1700... # Start of validity for this context fragment  
  principal: "A"      # The id of the user this pertains to  
  name: "project"     # User attribute (team, project, meetings, ...)  
  value: "XYZ"        # Opaque identifier  
}
```



**You only need to choose the user attributes you want to use
Facade takes care of the rest (implicit social network featurization)**

User features to dense vector



Featurization Takeaways

1

**Universal
featurization method**

2

**Robust to
distribution drift**

3

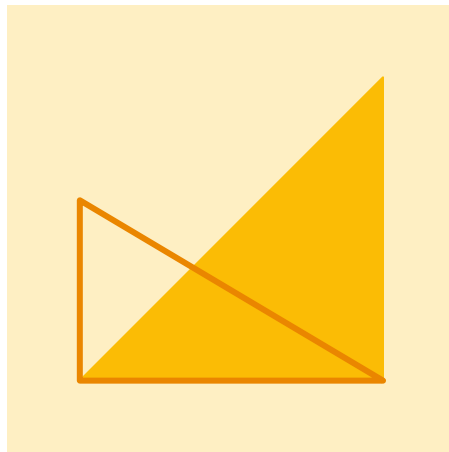
**New resources and
users w/o retraining**

4

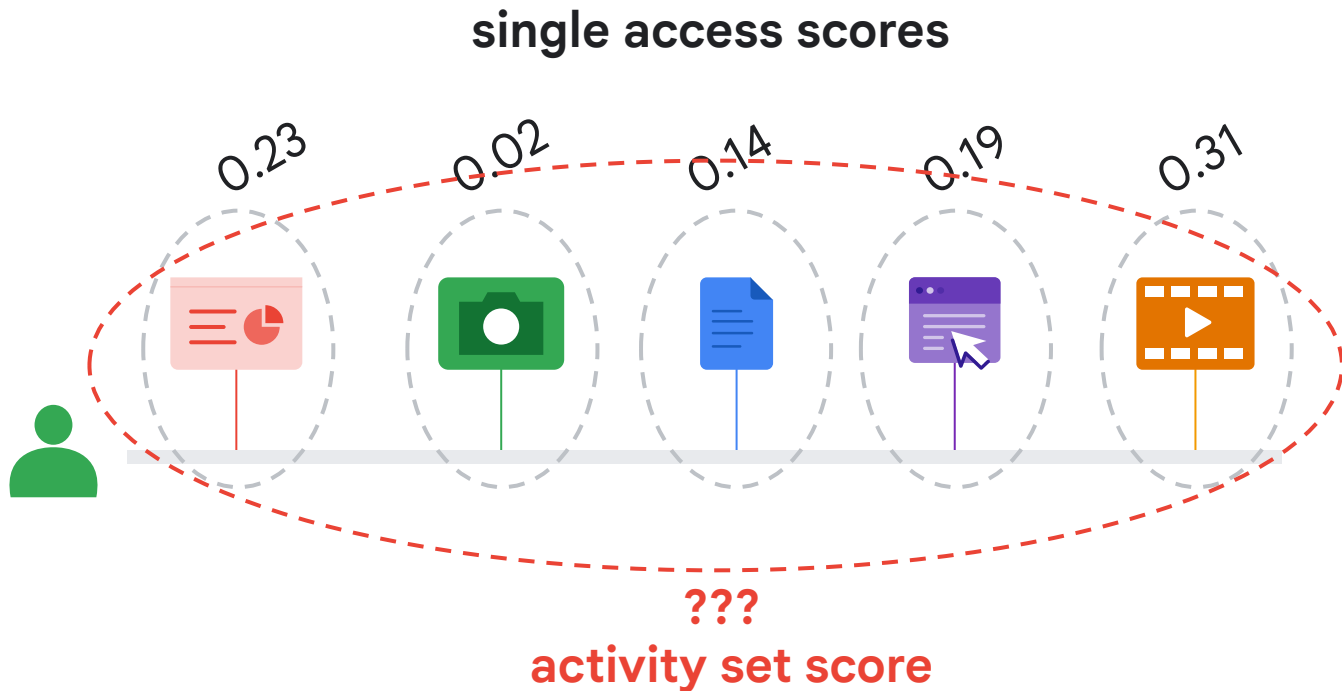
Fast and efficient



Scoring Arbitrary Time Periods



Pointwise VS Activity Set Scoring



A Simple Problem?



Average of scores

attacker can decrease score by adding benign accesses



Sum of scores

users with more activity will be more anomalous



Max score

ignores all but one access

Scoring Diversity of Anomalous Activity

Eliminate redundant and repetitive anomalies

Use the resource embeddings for similarity

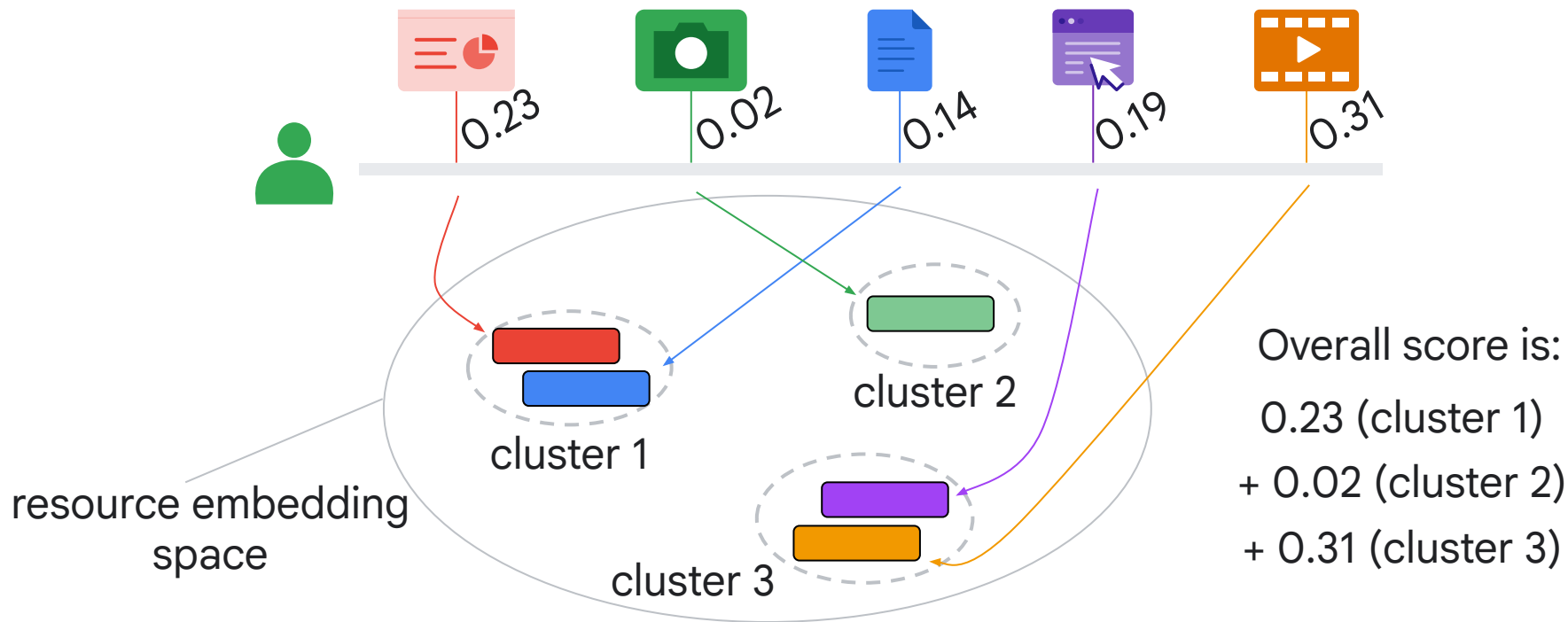


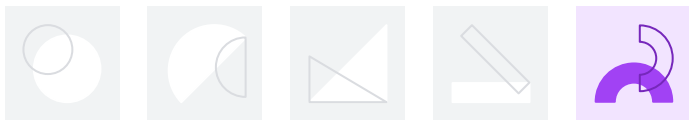
1. **Cluster** **similarly-anomalous** accesses together
2. **Sum** together **max** score of each cluster

Prevent attacker from hiding malicious activity

More diversely-anomalous sets score higher

Example





Finding Insider Attacks with FACADE



Red Team Insider Threat Scenarios



Media Sharing Platform

Attackers seek corporate financial data, individual creators earnings, ...



Hardware Product

Attackers seek next gen device design, timelines, pictures, schematics, ...



AI Research

Attackers seek next gen AI: unpublished papers, code, model weights, ...

Operational Setup



15 participants

Full-time employees with interest in cyber security



High-level playbooks provided

attackers seek to discover and access sensitive information
attackers not provided detailed attack plans or target resources



Various levels of attack success per participant

Evaluation results

~180,000+ user accounts

Triaging budget: top 10 users/day

Detects 4 out of 15 attackers

More details in

<https://arxiv.org/abs/2412.06700>



Try it yourself



Reference implementation

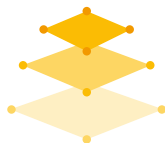
<https://github.com/google/facade>

Note: as mentioned Facade is meant to work on large scale data and requires you bring your own modeling. Using it on small datasets won't work well.

Takeaways



Insider threats: low incidence high impact attacks
Detection requires contextual analysis



FACADE: high-precision contextual anomaly detection
Works for single-access *and* activity set



Adaptable to many systems and use-cases
Open-source model and featurizer code available

Slides:

<https://elie.net/facade>

Code

<https://github.com/google/facade>



#BHUSA @BlackHatEvents