

Empowering Real-Time Fraud Prevention with Apache Beam

Hai Sadon



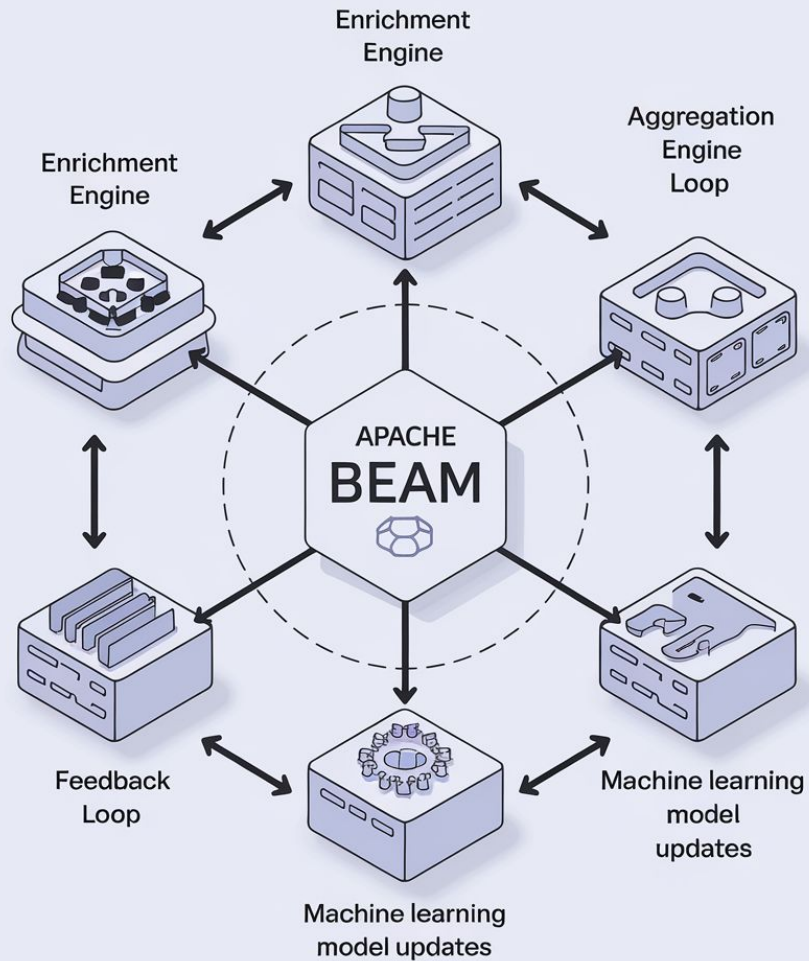
BEAM
SUMMIT

September 4-5, 2024

Sunnyvale, CA. USA

Real-Time Fraud Prevention with Apache Beam

A Deep Dive into Our Modular, High-Performance Fraud Detection System



About me

- Leading the Data Platform Group at Transmit Security
- Previously managed a team overseeing Apache Flink infrastructure for Microsoft engineering teams





Transmit Security Mosaic Platform



Orchestration

Simplified integrations, policy & decisioning, journey workflow



Identity Management

User profiles, authorization, SSO



Authentication

Passkeys, passwordless MFA, magic links



Identity Verification

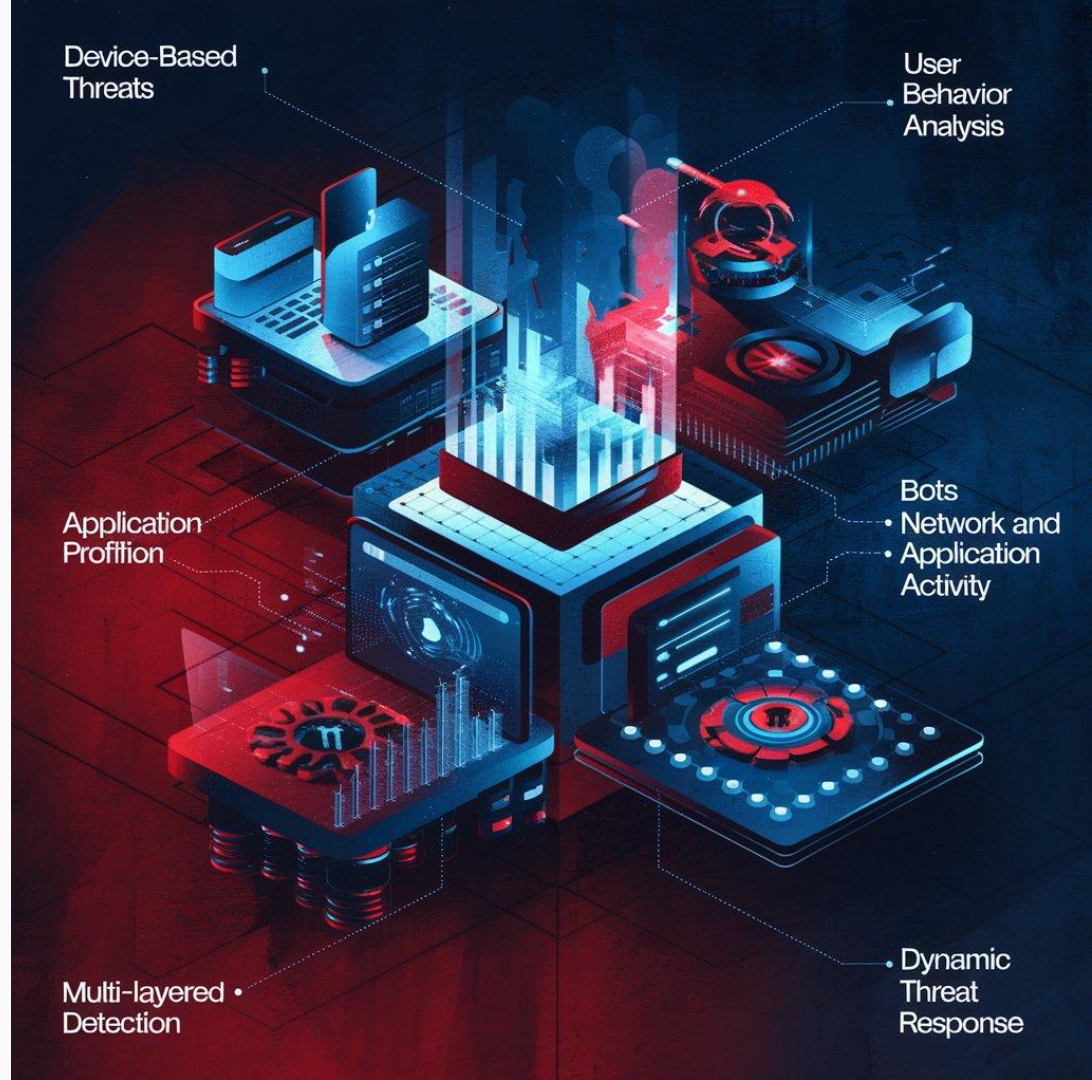
Documentation and database checks, liveness and selfie analysis, embedded fraud intelligence



Detection and Response Service

Multi-mode, real-time and post detection, ML and AI driven

The Challenge



The Challenge of Real-Time Fraud Prevention

Device-Based Threats

Detecting compromised or spoofed devices used for fraud.

Identifying device anomalies in real time.

User Behavior Analysis

Differentiating between normal and suspicious activities.

Continuous monitoring and adaptive detection.

Bot Activity

Automated attacks mimicking human behavior.

Advanced detection of sophisticated bot activities.

Network and Application Activity

Identifying coordinated attacks across networks and apps.

Real-time monitoring for unusual patterns.

User Profiling

Differentiating legitimate users from fraudsters.

Maintaining accurate, dynamic user profiles.

Multi-Layered Detection

Integrating various detection methods like device fingerprinting, biometrics, and network analysis.

Creating a unified system to prevent fragmented decision-making.



BEAM
SUMMIT

Our Fraud Detection System Architecture



Data Capture



Enrichment Engine



Aggregation Engine



Feature Engine



Machine Learning Models Engine



Rule Engine



Feedback Loop



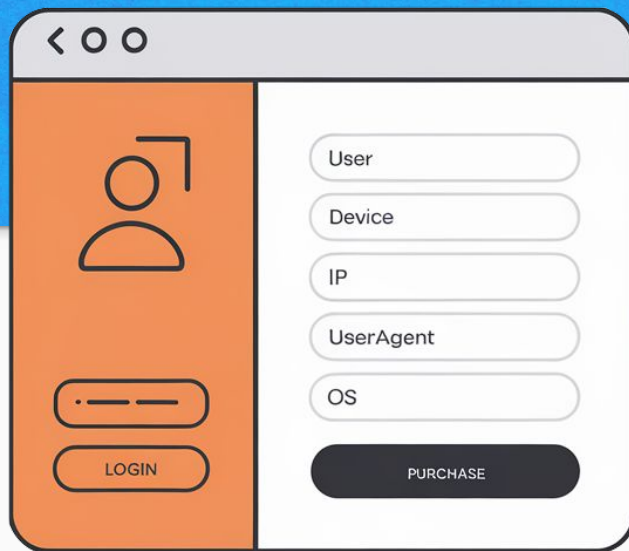
Data Skew Management











The Data Capture



Captures events
in real-time

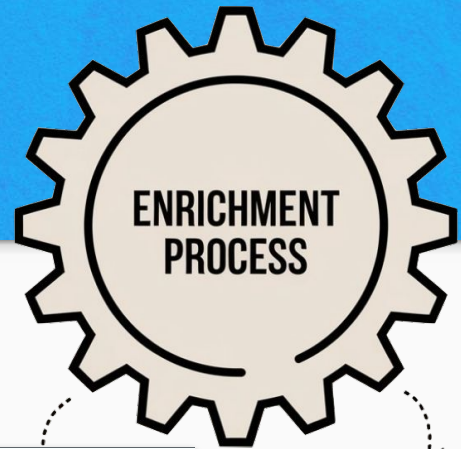


-  **Capture**
-  **Enrichment**
-  **Aggregation**
-  **Feature**
-  **ML**
-  **Rule**
-  **Feedback**
-  **Data Skew**



BEAM
SUMMIT

The Enrichment Engine



Process raw events



Enhances data with external sources



Utilizes Apache Beam's flexibility for integration



Extract data from external sources



Transform it to unified data model



Load it to external state (database)

```
{  
  "actionType": "login",  
  "ip": "192.168.1.1",  
  "userId": "user123",  
  "deviceId": "device456"  
}
```

```
{  
  "actionType": "login",  
  "ip": "192.168.1.1",  
  "userId": "user123",  
  "deviceId": "device456",  
  "geoLocation": "New York, USA",  
  "asn": "AS12345",  
  "knownProxies": [  
    "proxy1",  
    "proxy2"  
  ],  
  "hostedServices": [  
    "service1",  
    "service2"  
  ]  
}
```



Capture



Enrichment



Aggregation



Feature



ML



Rule



Feedback



Data Skew



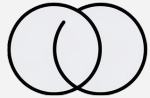
BEAM
SUMMIT

Split and Merge Approach



THE CONCEPT

When dealing with multiple stages or processes that can operate independently, this approach can be applied to parallelize processing.



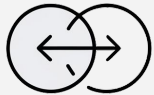
SPLIT

The event is cloned for each key, allowing parallel processing in different stages or processors.



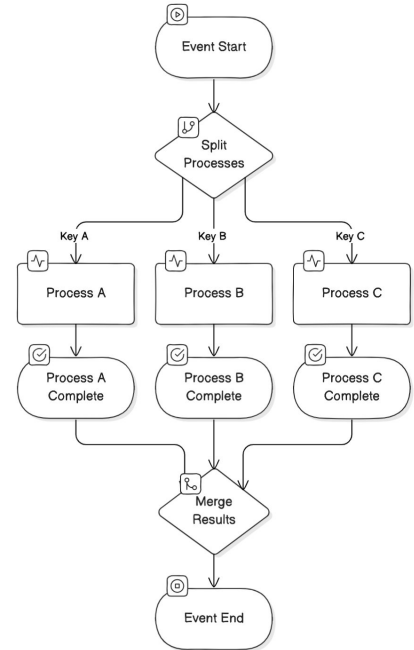
PROCESS INDEPENDENTLY

Each processor handles its assigned key without interference from other stages.



MERGE

Once all the stages are completed, the results are merged back into a single event, integrating the outputs from each stage.



ParDo



BEAM
SUMMIT

The Aggregation Engine



Processes enriched events



Splits events by aggregation key



Leverages Beam's stateful processing



Split and merge approach for parallel processing



Aggregation abstract interface for simplicity



Two phase process for each aggregation - update and serve



Capture



Enrichment



Aggregation



Feature



ML



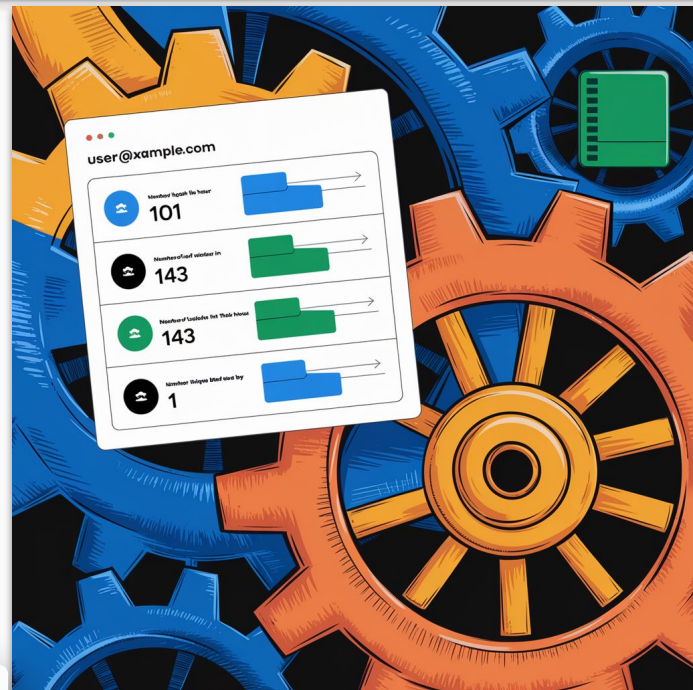
Rule



Feedback



Data Skew



BEAM
SUMMIT

Virtual Aggregations



Used the aggregation interface



Splits events by aggregation key



Leverages Beam's stateful processing



Get its state from an external stream



An integral part of the aggregation engine



Capture



Enrichment



Aggregation



Feature



ML



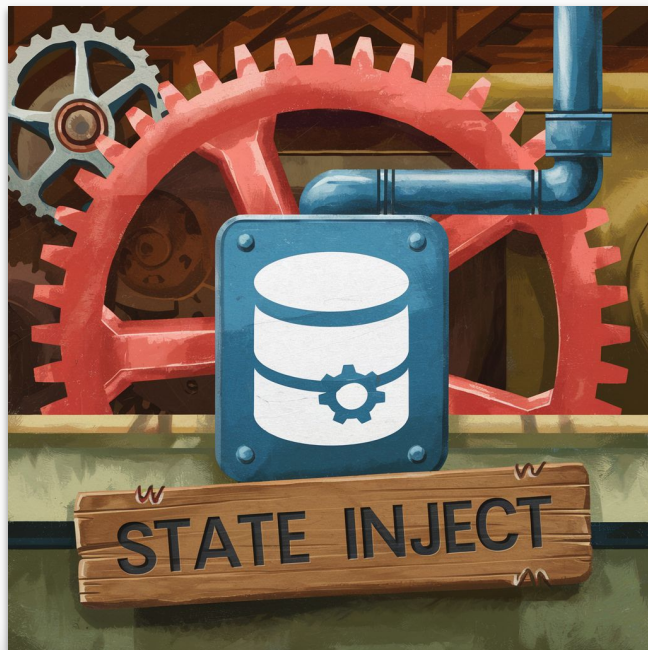
Rule



Feedback



Data Skew



BEAM
SUMMIT

The Feature Engine



Processed aggregated data



Key less processing for high distribution



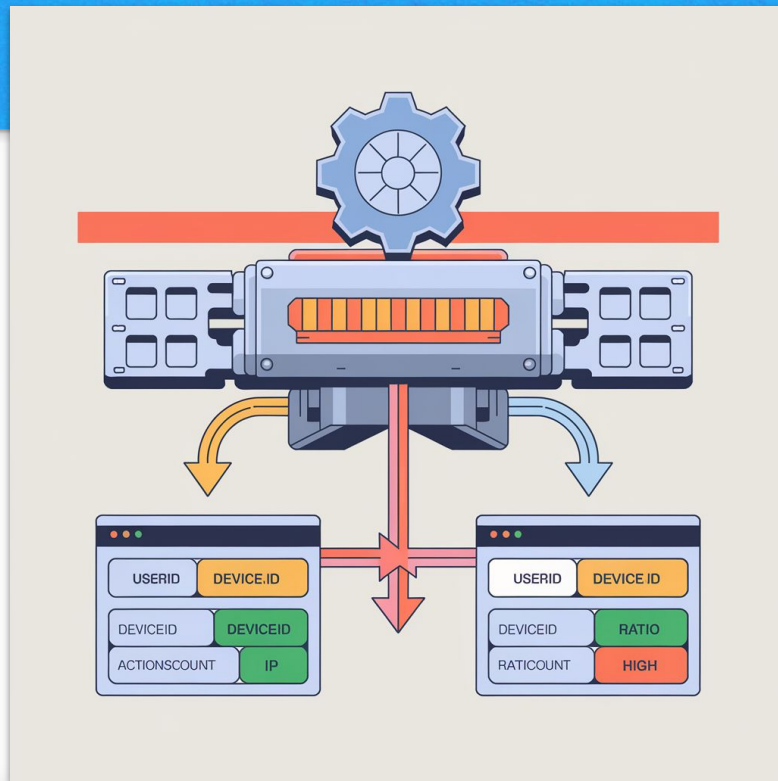
Abstract Interface for feature extractor



Leverages Beam's parallelism



Maintained by the security researcher team



Capture



Enrichment



Aggregation



Feature



ML



Rule



Feedback



Data Skew



BEAM
SUMMIT

Machine Learning Models



Processed featured data



Key less processing for high distribution



Interface for ML model serving



Leverages Beam's parallelism



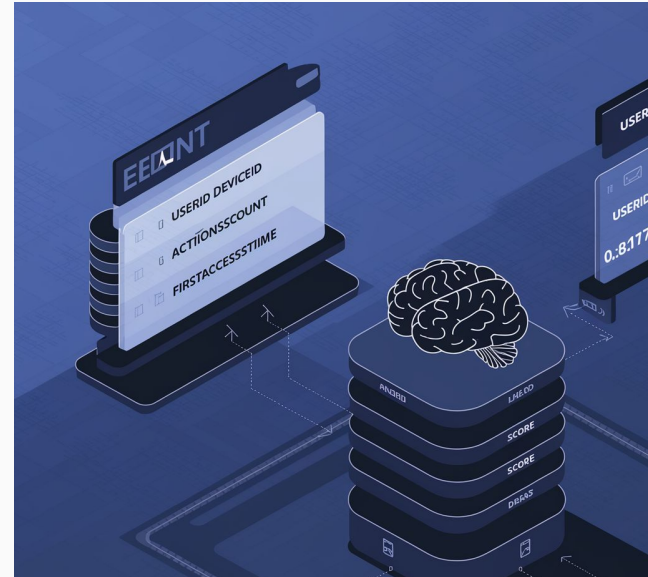
Ability to run multiple versions of models and compare results



Maintained by the data science team



Split and merge approach for parallel processing



Capture



Enrichment



Aggregation



Feature



ML



Rule



Feedback



Data Skew



BEAM
SUMMIT

Rule Engine



Processed models output data



Keyless processing for high distribution



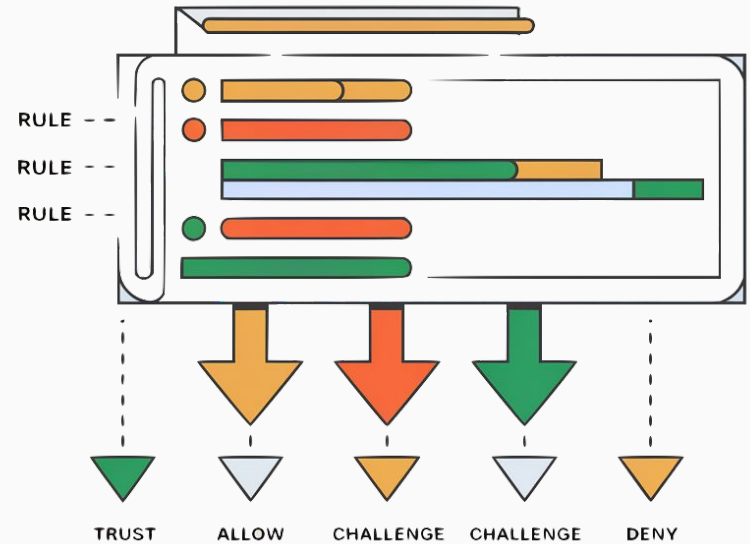
Interface for rule registration



Leverages Beam's parallelism



Maintained by the security researcher team



Capture



Enrichment



Aggregation



Feature



ML



Rule



Feedback



Data Skew

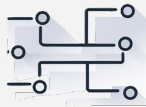


BEAM
SUMMIT

Post Processing Aggregations



Processes evaluated records data



Splits events by aggregation key



Leverages Beam's stateful processing



Served as a virtual aggregations



Split and merge approach for parallel processing



Aggregation abstract interface for simplicity



Two phase process - update and export state



Capture



Enrichment



Aggregation



Feature



ML



Rule



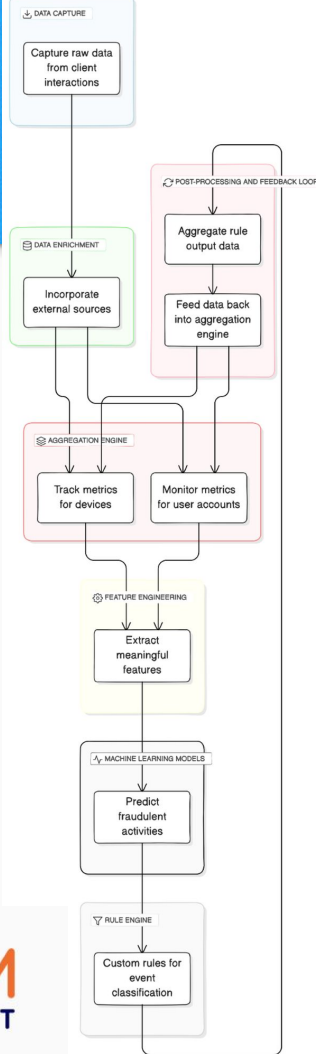
Feedback



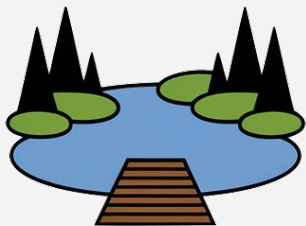
Data Skew



BEAM
SUMMIT

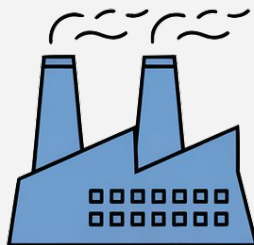


Data Skew Management



Store the data for batch processing

DATA LAKE



Process the data offline

BATCH PROCESS



Ingest back aggregation results

INGESTION



Serve the skewed keys (virtual aggregations)

AGGREGATIONS SERVING



Capture



Enrichment



Aggregation



Feature



ML



Rule



Feedback



Data Skew



Combiners



BEAM
SUMMIT

Challenges and Workarounds

Lack of structure object support in state

Like dictionaries.
Also states are must be declared and are not dynamic.

Used naive serializer

No built-in state TTL mechanism

No offering

Using timer for TTL

Key iterations is not possible

No offering

Create a key base store

Cannot query the state outside the application

No offering

Pushed the state into BigTable

Ordering

There is no low latency solution for ordering.

Custom logic (no real solution)

Cannot clear window state

No offering

Create job from snapshot from time to time



Collaboration and Modularity

MODULAR

Modular design supporting cross-team collaboration

INDEPENDENT

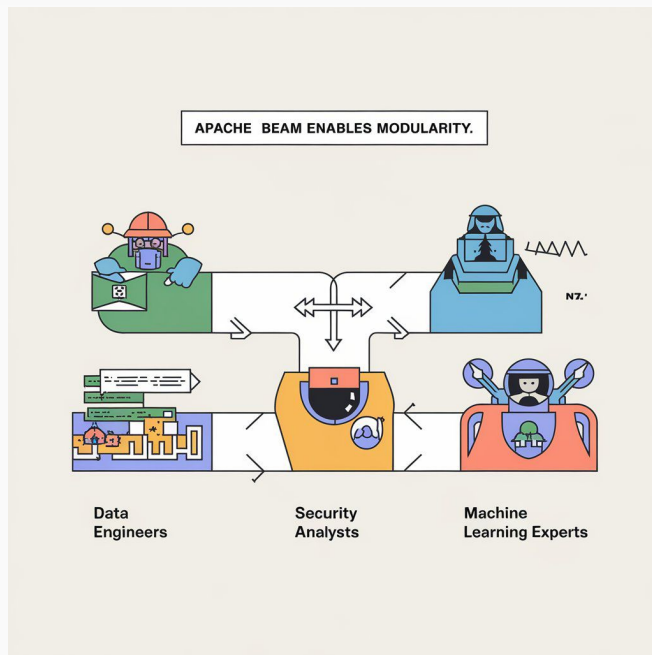
Independent development of pipeline stages

MODULAR EXECUTION

Ability to run stages independently or end-to-end

FLEXIBILITY

Once all the stages are completed, the results are merged back into a single event, integrating the outputs from each stage.



BEAM
SUMMIT

Conclusion

Recap of
challenges and
solutions

Importance of
Apache Beam in
our success

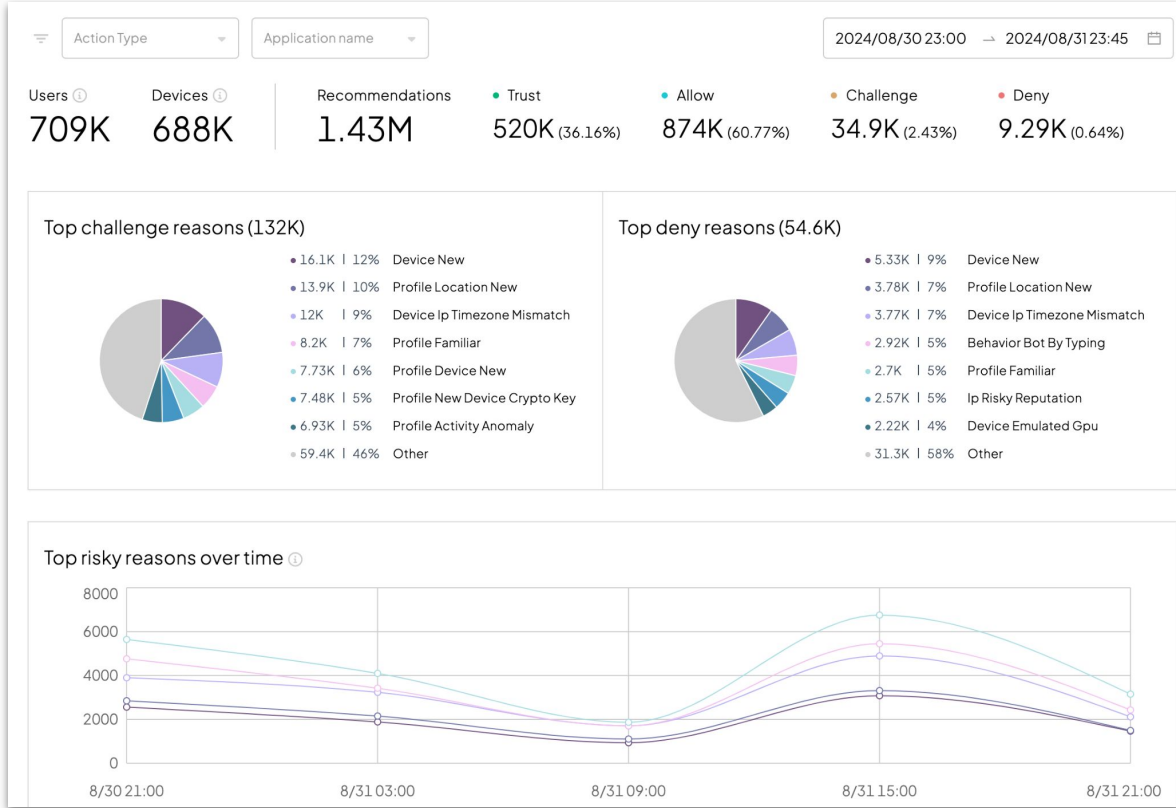
System
adaptability and
power

Inspiration for
leveraging Beam
in other projects



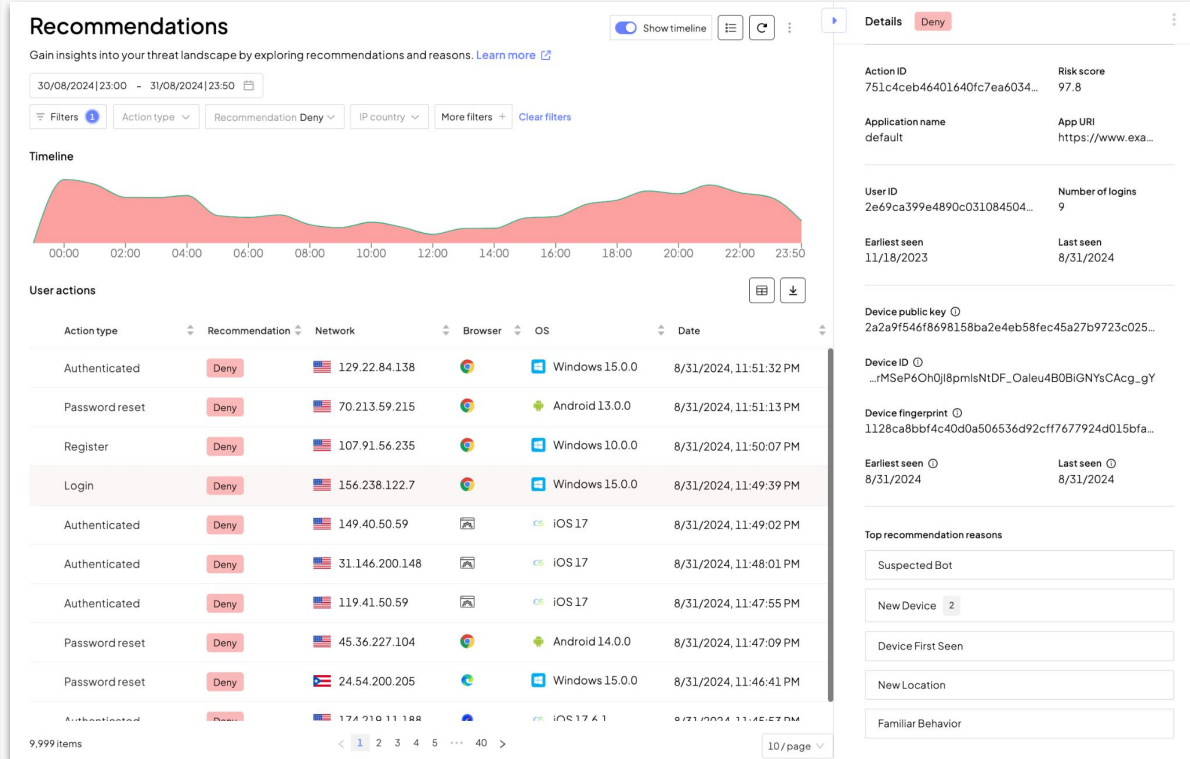
BEAM
SUMMIT

Transmit Security DRS product



BEAM
SUMMIT

Transmit Security DRS product



Thank you!

Questions?



Hai Saadon

<https://www.linkedin.com/in/hai-saadon-61a34a74/>



BEAM
SUMMIT