Aggregate API Key Discovery

Explainer WICG Review

aksu@google.com,

26 June 2023

Google

Outline

- Context: Aggregate API
- Why: the need
- How: the proposal
- Show me: example
- Summary
- Tell Me More: questions

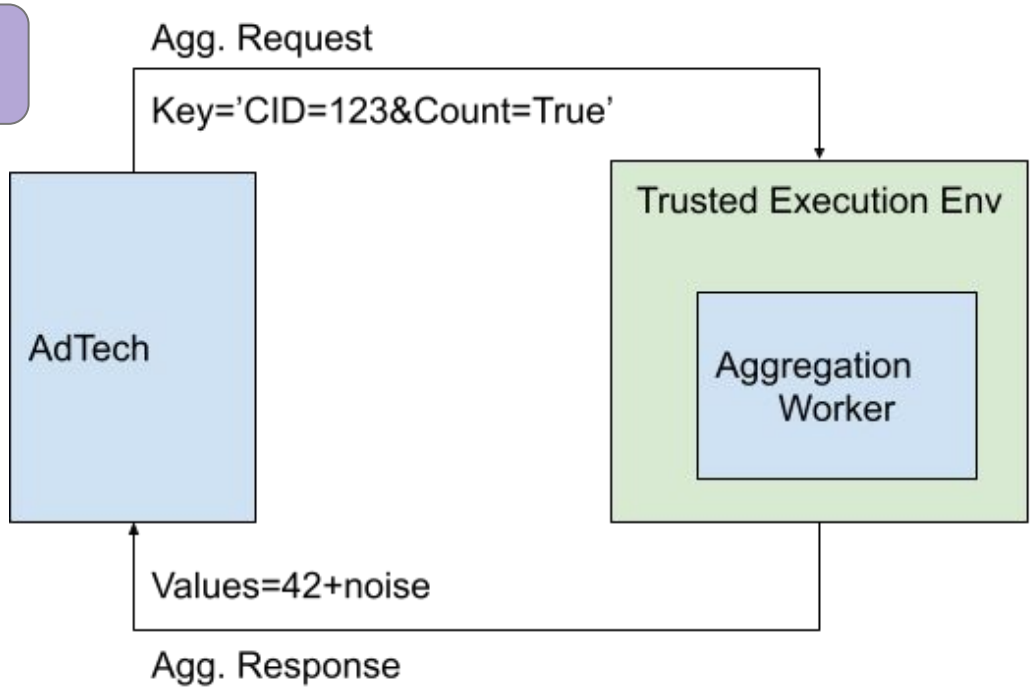
Context

ARA-APIs:

- Event API
  - Local Noise (DP via RR)
  - No Encryption, can be directly used
- Aggregate API
  - Central Noise (DP via DLaplace)
  - Encrypted, AdTech use via TEE

not about this

Focused on this API



Why: The Problem and the Pain

```
def aggregate(reports: AggregatableReports,
              domain_keys: Sequance(key)) ->
               Generator(Tuple(key, value)):
```

AdTech side pain:

- Aggregate API requires the list of keys with each query.
  - List of keys can be too large to track.
    - Just listing all 48 bits keys require 2^{48}x48 bits = 1,536 Terabytes
  - List of resulting buckets can be too large to post-process.
    - 2^{48} x (128+16) bits = 4,608 Terabytes
  - Assumes AdTech has a way for dense encoding.
- Existing request for [key discovery mechanism · Issue #583]()

Proposed extension:

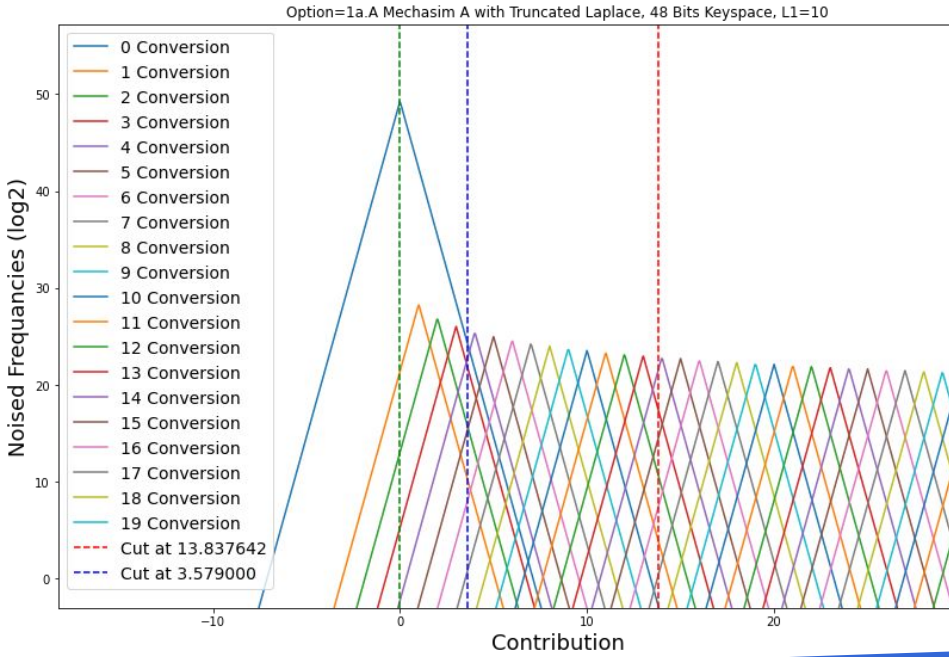
- a new way of querying that does not need an explicit keys list

How: Proposed Mechanisms

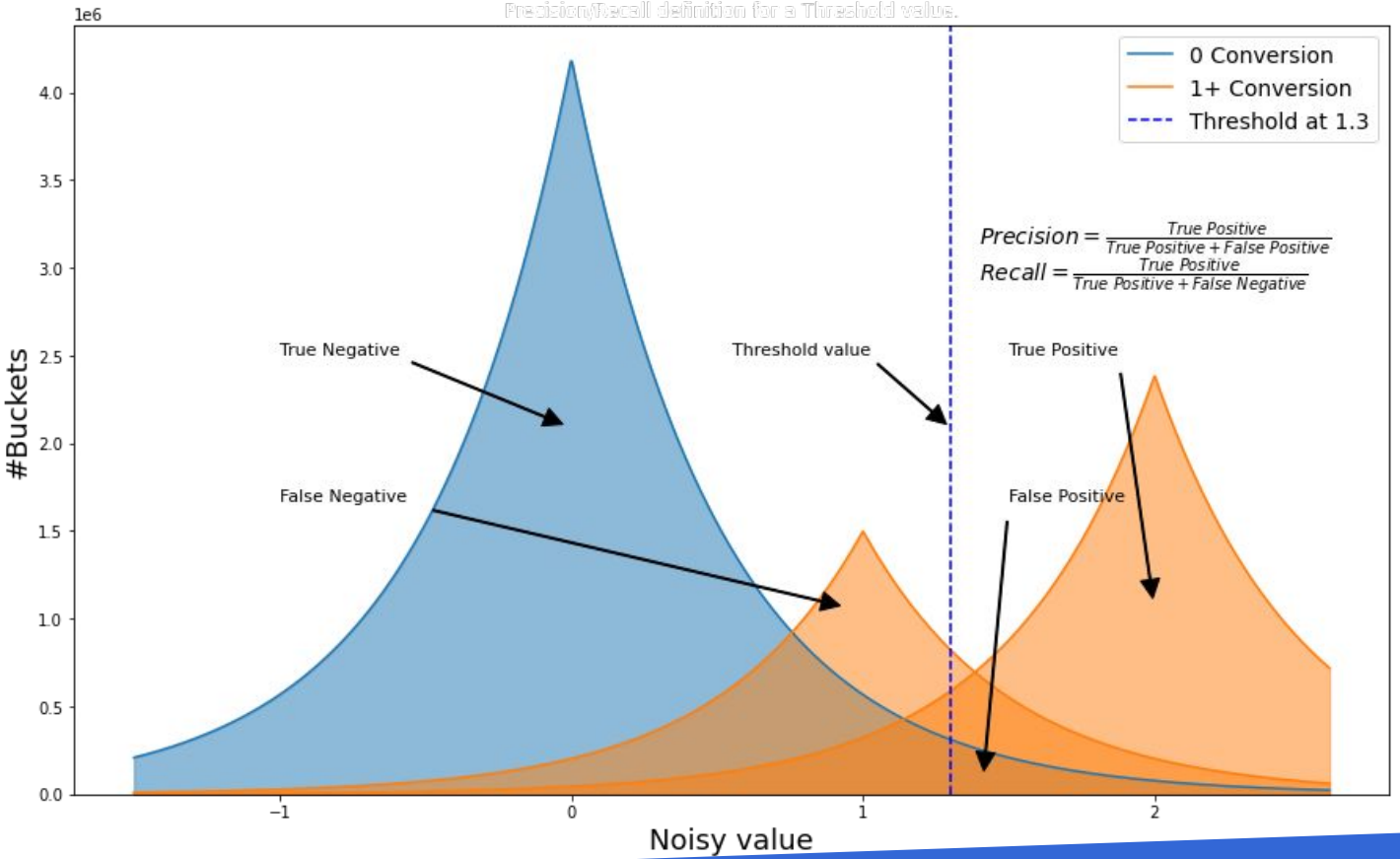
```
def aggregate_with_key_discovery(reports: AggregatableReports,
                domain_keys:Sequance(keys)=set(), threshold: Optional[float],
                keyMask: Optional[[bool]] ) -> Generator(Tuple(key, value)):
```

Proposed Mechanism:

- Idea:
  - Use truncated laplace
  - Threshold
    - balance precision/recall.
  - Key Mask
    - reduce key space.
  - Pre-declared buckets (domain_keys)
    - increase recall.



Precision/Recall Tradeoff



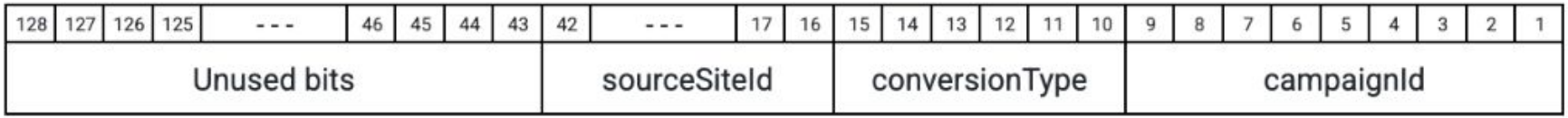
Show Me: Example

Assume AdTech wants to track

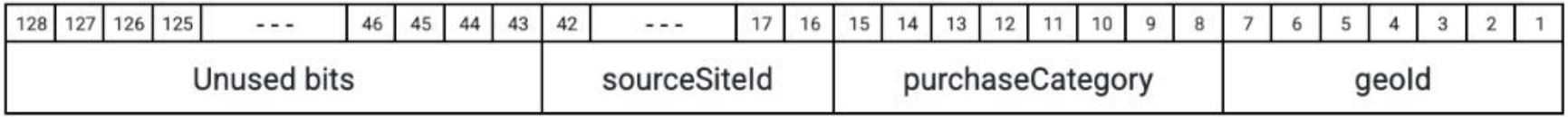
- Conversion count on sourceSiteId x Conversion type x campaignId breakdown and
- Conversion values on sourceSiteId x Purchase category x geoId breakdown.

Thus, have following aggregate key layout.

Key 1: “campaignCounts” = sourceSiteId x Conversion type x campaignId

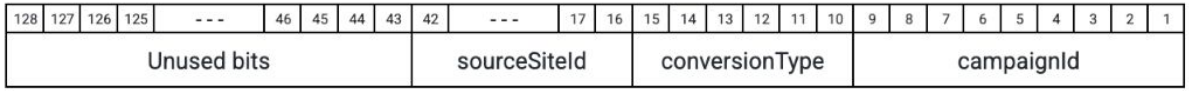


Key 2: “geoValue”: sourceSiteId x Purchase category x geoId

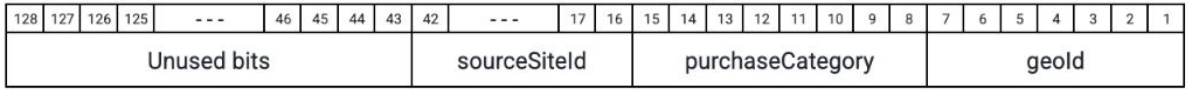


Show Me: Example

Key 1: “campaignCounts” = sourceSiteId x Conversion type x campaignId



Key 2: “geoValue”: sourceSiteId x Purchase category x geoId




```
# lower 42 bits are used, rest is unused.
keyMask = [0]*(128-42) + [1]*42
aggregated_results = aggregate_with_key_discovery(
                        reports=reports, threshold=0.715,
                        keyMask=keyMask)
for bucket in aggregated_results:
  sourceSiteId = (bucket.key >> 15) & 0x7ffffff # extract
sourceSiteID bits
  sourceSite = decode_siteid(sourceSiteId)
  # process rest based on sourceSite
  ...
```

Per offline estimation
@threshold=0.715
Precision is 99.981%
Recall is 99.981%

Summary

- Key discovery helps to work with unbounded domain file
- It is backward compatible
- Feedback welcome, use issues/864

Tell Me More: questions?

- Feedback ?
- Comments ?
-