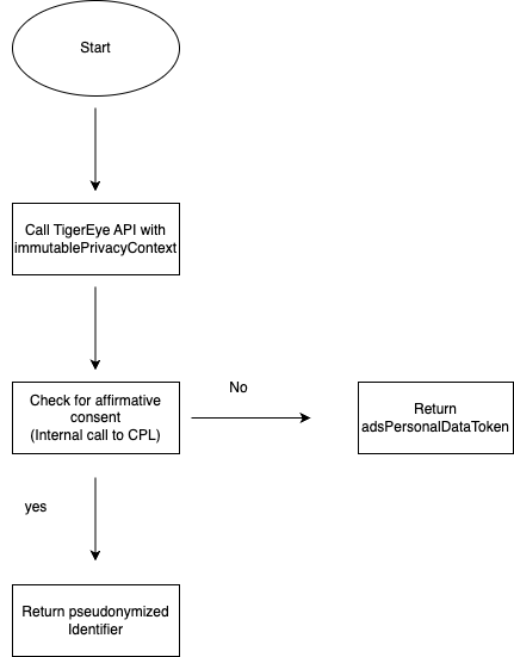
**DMA Tiger Eye API updates and code changes in 1P ACR services (IPS and SIGT)**

SIM Link:  
<https://sim.amazon.com/issues/P99587092>

**1. Background**

[DMA HLD: Consent Enforcement using TigerEye](https://quip-amazon.com/Gb3KApJZpdzk) proposes the solution to update the TigerEye API to only returns the pseudonymized Identifier only if we have affirmative consent to use the data for the identifier, else return adsPersonalDataToken which will be encrypted version of pseudonymized identifier. By hiding the pseudonymized identifier, TigerEye library limits the capability in which the customer data can be used by such systems. The adsPersonalDataToken is returned in case caller application’s downstream is exempted to use the data without consent check.   
  
To achieve this TigerEye API internally calls CPL to evaluate if calling Ad Product has required consent or not. For evaluation CPL calls the consent service(See [DMA HLD: consent gathering and vending infra](https://quip-amazon.com/v2u8A3mCv4Sk) for details) to retrieve the AdsUnifiedPrivacyString and execute policies defined by Customer Trust team. If CPL responds positively, a valid pseudonymized Id is returned else library returns the adsPersonalDataToken. This token remain opaque to all the ads system except the one who has implemented usage time consent check technique for DMA compliance as described in [Ads data bubble design](https://quip-amazon.com/gtSxAcLFlW5w/DMA-Consent-HLD-for-Ads-Data-Bubble#temp:C:DAT5ff5c157c4084f6b8a0d2f0c5).   
  
For consent evaluation CPL requires the marketplace/country-code and LOB that owns the personal data in-scope. There is no reliable way for the TigerEye API to learn such information on its own. Therefore the caller must encode this parameter into immutablePrivacyContext and pass this as input parameter when calling the TigerEye API. There will be an helper API in CPL to encode the parameter and create immutablePrivacyContext (See [Ad Privacy Data Backbone](https://quip-amazon.com/sapQARM3NV8m) for more details on immutablePrivacyContext).  
  
**High Level Execution Diagram:**  
  


**Acronyms**

* CRS - Consent Retrieval Service
* LOB: Line-of-business
* CuTE : Customer Trust Engineering
* CPL: CuTE owned library for the DMA policy evaluation
* TE - TigerEye
* AUP - Authorization Use Purpose

**2. API Changes :**

**Primitive Id** : Id with which CRS store the consent directly. IPS and SIGT Service uses retail customer id which is designated as primitive Id. Reference to be stated as primitive id given in [DMA HLD: Consent Enforcement using TigerEye](https://quip-amazon.com/Gb3KApJZpdzk) Appendix a  
\*Currently IPS and SIGT service does not have any **Non-Primitive Id or Event**passed in API calls.[(Link](https://code.amazon.com/packages/AdsVastImpressionTransformerLambda/blobs/57e2865bd415852578a7d11fb2eafae4a657abe3/--/src/com/amazon/ipsservice/impression/lambda/CustomerIdUtil.java#L106:~:text=PseudoIdOutput%20output%20%3D%20defaultPseud.getPseudoId(input)%3B))

**3. Current TigerEye API**

For IPS and SIGT Service till now the API calling was done through the following implementation ,explained in this document.[TigerEye Migration for IPS and SIGT](https://quip-amazon.com/4jMrAtkbNwIj)  
  
PseudoIdOutput getPseudoId(GetPseudoIdInput input) [Code Link](https://code.amazon.com/packages/AdsVastImpressionTransformerLambda/blobs/57e2865bd415852578a7d11fb2eafae4a657abe3/--/src/com/amazon/ipsservice/impression/lambda/CustomerIdUtil.java#:~:text=GetPseudoIdInput%20input%20%3D%20new%20GetPseudoIdInput(id%2C%20ConstantsUtil.IDENTIFIER_TYPE)%3B)  
Currently IPS and SIGT calls single API with retail customer id , which is part of the input list.   
When using a single API call primitive IDs (retail customer ID in IPS and SIGT), the TE library lacks context regarding the actual IDs to use for consent evaluation. Till now Tiger Eye relied on the client to provide the correct primitive IDs. In the case of batch API calls, IDs from multiple events can be combined to create a significant batch size. However, there is no guarantee that all items in the batch belong to a single event. To address this issue, Tiger Eye upgrade proposes to revoke access to this API for all callers. Instead, we will create a new API as shown in response.  
  
  
**Steps To Upgrade (**[**wiki**](https://w.amazon.com/bin/view/AD_Privacy_Security/Product_and_Programs/Saber/DMA/DMAConsentOnboarding/#HTigerEyeOn-boarding:~:text=If%20you%E2%80%99re%20an,for%20on%2Dboarding.)**)**  
As IPS and SIGT is already existing TigerEye client, Reference Document : [DMA Enforcement - TigerEye API Changes](https://quip-amazon.com/aGpOAA4JMSa3), [TigerEye Migration for IPS and SIGT](https://quip-amazon.com/4jMrAtkbNwIj)

* Request AAA access. Refer here [ [TigerEye CloudAuth Section](https://w.amazon.com/bin/view/AD_Privacy_Security/Advertising_Privacy_Engineering/TigerEye/UserDocumentation#HCloudAuth)].
* Build connectivity with Consent Retrieval Service (CRS) - Refer here: [ [Consent Retrieval Service](https://w.amazon.com/bin/view/AD_Privacy_Security/Advertising_Privacy_Engineering/TigerEye/UserDocumentation/#HConsentRetrievalService) ]
* Migrate the existing API to the new API:

**current TigerEye APIs to new DMA APIs according to wiki:**

|  |  |
| --- | --- |
| **Current TigerEye API** | **DMA TigerEye API** |
| [getPseudoId](https://w.amazon.com/bin/view/AD_Privacy_Security/Product_and_Programs/Saber/DMA/DMAConsentOnboarding/#HTigerEyeOn-boarding:~:text=Q1%3A%20Can%20you%20provide%20the%20mapping%20of%20current%20TigerEye%20APIs%20to%20new%20DMA%20APIs%3F) | getConsentedPseudonymizedIdentifierBag |

\*As IPS and SIGT is not a Vault System, we need not follow the CPL(Central Policy Library) wiki for on-boarding.

**4. Response from TigerEye API:**

New TigerEye API returns PseudonymizationResponse. It contains: 

1. PseudonymizedIdentifier list maps one to one with the input allIdentifiers.  PseudonymizedIdentifiercontains three fields pseudoId,wasPseudonymizationSuccessful and exception.
   1. pseudoId will be non-null when operation is successful and client has all the consent required to use the data. pseudoId is of type Identifier which is a POJO class of pseudoId and objectType. pseudoId.getIdentifier() is same as pseudoId received in response today.
   2. wasPseudonymizationSuccessful is a boolean value indicating if the pseudonymization for this Id is successful or not. If this value is false, then clients are supposed to either skip the identifier or park and retry later (library has already attempted retry). Details of the failure is available in exception object.
   3. exception field will be non-null when operation was not successful. If client decide to retry the pseudonymization for the failed Id in event, they should pass the adsUnifiedPrivacyStringapplied in the retry request.
2. adsPersonalDataToken will be non-null when operation is successful and client do not have the required consent to use the data. Client should not use this data and the adsPersonalDataToken needs to be sent to the downstream team to tunnel the data to the vault system as mentioned in [Ads Data Tunnelling HLD (“default opaque personal data”) [Privileged and Confidential]](https://quip-amazon.com/pa2NAS7ExQI9)
3. adsUnifiedPrivacyStringapplied which is the AUP used to evaluate the policy. Clients are required to pass this AUP to downstream team.
4. isConsentedForAllAdsPurpose this will be true when event is consented for general purpose bubble else false.

The API also throws the exception PseudonymizationException which is a generic exception and wraps actual exception.   
  
To address this concern, a new API will be created as :   
  
PseudonymizationResponse getConsentedPseudonymizedIdentifierBag(PseudonymizationRequest input)   
throws PseudonymizationException  
  
Where,   
class PseudonymizationRequest {  
/\*\*  
*list of all PII (Personally Identifiable Identifier)*  
\*/  
@NonNull   
List<IdentifierToPseudonymize> allIdentifiers;  
  
  
//*immutablePrivacyContext of event*  
@NonNull  
ImmutablePrivacyContext immutablePrivacyContext;  
  
/\*\* Optional field to pass if the privacy string is received from upstream calls or via other mechanism as defined in [DMA HLD: consent gathering and vending infra](https://quip-amazon.com/v2u8A3mCv4Sk) (like s3 lookup). adsUnifiedPrivacyString is passed, we will use this privacy string for policy evaluation.   
\*\*/  
String adsUnifiedPrivacyString;  
  
// *Timestamp when the corresponding event occur in UTC. Same as eventTimeStamp passed in today’s API* *call*.   
@Nullable  
Instant eventTimestampInUTC;  
}  
  
And class IdentifierToPseudonymize {  
@NonNull  
Identitifier identifier;   
  
@Nullable  
Instant semiStableTimestamp;  
}  
  
And class Identitifier {  
@NonNull  
ObjectType objectType; //Enum objectType  
  
@NonNull  
String idValue;  
  
}  
  
  
And class PseudonymizationResponse {  
  
  
@NonNull  
List<PseudonymizedIdentifier> pseudonymizedIdentifier; //pseudonymized id in the same order as input.  
  
@NonNull  
String adsUnifiedPrivacyStringApplied; //consent string applied for policy evaluation  
  
*/\*\*  
\* Return true when identifierBag is consented for general purpose bubble else false.  
\* If operation fails, value will be null.  
\*/*@Nullable  
Boolean isConsentedForAllAdsPurposes;  
  
  
*/\*\*  
\* adsPersonalDataToken to be passed to the vault systems when required consent is not present to use the identifier  
\* and associated data in Ads Data bubble.*  
*\* Null if isConsentedForAllAdsPurposes is NULL or true.  
\* byte[] containing encrypted version of all pseudonym which requires encryption for tunneling them to vault.  
\*/*byte[] adsPersonalDataToken;  
}  
  
And class PseudonymizedIdentifier {  
Identifier pseudoId; //pseudoId if operation succeeded and application is allowed to retrieve else null.   
  
boolean wasPseudonymizationSuccessful //true when operation was successful  
  
Exception exception; //Internal exception if operation failed. It will be null if operation succeed   
}  
  
  
**4.1 Update AdsPseudonymizationLibrary**  
To start using the new Tiger Eye response , we first need to consume the latest commit of [AdsPseudonymizationLibrary](https://code.amazon.com/packages/AdsPseudonymizationLibrary/trees/mainline)and start their DMA development against the stubs([Link](https://w.amazon.com/bin/view/AD_Privacy_Security/Product_and_Programs/Saber/DMA/DMAConsentOnboarding/#HTigerEyeOn-boarding:~:text=Teams%20can%20consume%20the%20latest%20commit%20and%20start%20their%20DMA%20development%20against%20the%20stubs.))  
  
**4.2 Proposed Change for IPS and SIGT Service:**

**try{**  
  
**Identifier identifier = Identifier.builder()**   
 **.identity(id)**  
 **.objectType(ObjectType.AMAZON\_CUSTOMER\_ID)**  
 **.build();**  
  
**final ImmutablePrivacyContext ipc = ImmutablePrivacyContext.newBuilder()**  
  
 **.**setCountryCode**(CountryCode.**US**)//Currently used country code for US with DMA onboarding**  
 **.build();**  
  
IdentifierToPseudonymize identifierToPseudonymize = new IdentifierToPseudonymize(identifier,null);  
//ipc is passed as 2nd parameter if the privacy string is not handled by upstream calls   
final PseudonymizationRequest pseudoRequest = new PseudonymizationRequest(  
 identifierToPseudonymize,  
 ipc,  
 null,  
 Instant.now()  
 );  
PseudonymizationResponse response = pseudonymizer.getConsentedPseudonymizedIdentifierBag(  
pseudoRequest);  
PseudonymizedIdentifier pseudonymizedIdentifier = response.getPseudoIdentifier().get(0);  
if (!pseudonymizedIdentifier.isWasPseudonymizationSuccessful()) {  
throw pseudonymizedIdentifier.getException();  
}  
final String pseudoCustomerId = pseudonymizedIdentifier.getPseudoId();  
  
return Optional.of(pseudoCustomerId);  
}  
catch (TimeoutException e) {  
final String errorMessage = "Timeout while calling pseudonymized customer Id";  
log.error(errorMessage, e);  
} catch (PseudonymizationException e) {  
final String errorMessage =  
"PseudonymizationException while calling Service to pseudonymize retail customer Id";  
log.error(errorMessage, e);  
} catch (Exception e) {  
final String errorMessage = "Unexpected exception while calling Service to pseudonymize retail customer Id";  
log.error(errorMessage, e);  
}  
return Optional.empty();

**4.3 Cache Properties:**[Link](https://w.amazon.com/bin/view/AD_Privacy_Security/Advertising_Privacy_Engineering/TigerEye/UserDocumentation/)  
The exact scaling required for L2 cache depends on the expected size, which is currently TBD (To Be Determined) according to FAQ in reference Documnet. TE library anticipates in receiving 12 million new consents daily, with the assumption that 10% of these consents will need to be stored, growing at a rate of 2GB per day.  
The new upgrade allow clients to store the AUP in the cache as well.  
As IPS and SIGT service uses [custom L2 cache](https://quip-amazon.com/aGpOAA4JMSa3/DMA-Enforcement-TigerEye-API-Changes#temp:C:IOY2510b2cdb3414b25965e5ef40), the service don't need to make changes in implementation. The library provides an interface and model that will handle this for the service. The upgrade still needs to be tested in beta   
  
**Code Review**:  
<https://code.amazon.com/reviews/CR-103090151>  
<https://code.amazon.com/reviews/CR-103090924>