Open Measurement Device Attestation Implementation Guidance

Version 1.0 | November 2025

About This Document

The IAB Tech Lab Open Measurement Device Attestation Integration Guidelines was developed by the Open Measurement Commit Group with inputs from the Participant Working Group.

Significant Contributions from: Vinod Panicker, Diptendu Bhowmick, Sudeep Kemka, Haritha Devarakonda, Suhas Tikoo, Karthick Mahadevan, Chris Troein, KC Reaney, Justin Adler-Swanberg, Garo Hussenjian, Michael Spaulding and Ron Pinelli, SVP Digital Research and Standards at the Media Rating Council

IAB Tech Lab Lead: Jill Wittkopp, VP Product, IAB Tech Lab

About IAB Tech Lab

The IAB Technology Laboratory is a nonprofit research and development consortium charged with producing and helping companies implement global industry technical standards and solutions. The goal of the Tech Lab is to reduce friction associated with the digital advertising and marketing supply chain while contributing to the safe growth of an industry. The IAB Tech Lab spearheads the development of technical standards, creates and maintains a code library to assist in rapid, cost-effective implementation of IAB standards, and establishes a test platform for companies to evaluate the compatibility of their technology solutions with IAB standards, which for 18 years have been the foundation for interoperability and profitable growth in the digital advertising supply chain. Further details about the IAB Technology Lab can be found at https://iabtechlab.com.

DISCLAIMER:

Table of contents

Glossary of Terms

Attester - The Attester is the device manufacturer that attests to the authenticity of their devices.

Client - The Client is an app/video player. In the device attestation use case, it is an app/video player that seeks to demonstrate the authenticity of the underlying device.

Device Spoofing - Device spoofing is a means of Invalid Traffic, or IVT in which the User Agent and/or the OpenRTB device object is misrepresented.

Issuer - The Issuer is responsible for signing Privacy Pass token requests (token issuance) from the Attester.

Open Measurement SDK - The Open Measurement Software Development Kit (OM SDK) is designed to facilitate independent measurement of ads served to web video, mobile app, and Connected TV environments. This includes ad impressions, viewability, and other events such as play/pause, TV off, etc.

Origin/Verifier - The Origin per the Privacy Pass protocol is referred to as the Verifier in this mechanism. This is a measurement/verification service that seeks to verify that Clients it communicates with are running on authentic devices.

Privacy Pass - Privacy Pass is a protocol published by IETF that this mechanism has adopted for a digital advertising verification use case. The Privacy Pass protocol enables web clients to assert a property about themselves without revealing private information.

Rate Limit - Attesters may limit the frequency of Client verification requests within this workflow. Similarly, Issuers may limit the number of tokens issued. Note that this is distinct from OM SDK's sampling of Verifiers.

Sampling - To help conserve device resources, OM SDK sets a predefined probability for Attestation Requests made from the Client to the Verifier. Additionally, the Verifier determines which Attestation Requests to challenge, further conserving Client device resources.

Seller - An entity offering digital advertising opportunities (inventory) for purchase. This can be a direct seller (such as a publisher) or an intermediary (such as a reseller or an SSP).

Executive Summary

This document outlines a mechanism to address the prevalent device spoofing problem faced in digital ads. The approach involves adapting the Privacy Pass protocol published by IETF to the digital ads verification use case. The Privacy Pass protocol enables web clients to assert a property about themselves without revealing private information. This document describes how Open Measurement SDK (OM SDK) facilitates apps and video players in asserting that the underlying device is authentic, so that measurement servers hosted by various industry participants can independently verify that ad impressions are being rendered on authentic devices.

1. Introduction

1.1. Problem Statement

Bid requests and measurement beacons contain information about the device, which is used for various purposes, including ad selection and measurement. In OpenRTB bid requests, the "device" object carries this information, while the `User-Agent` HTTP header represents the same in case of measurement beacons, including impressions. Since this information is represented in a text string, it can be easily manipulated by intermediaries in the OpenRTB protocol and by actors fabricating HTTP requests. Such misrepresentation of device information is termed as device spoofing and can be performed for both benign and malicious purposes. For example, a web scraper trying to hide its identity may spoof a popular browser by using the `User-Agent` string appropriate to the browser and the underlying device. A malicious example would involve malware on a device spoofing various other devices that are considered more valuable from an ad monetization perspective. Large ad fraud operations detected in the past and prevalent bots leverage device spoofing (or `User-Agent` spoofing) as a key modus operandi to evade detection, typically layering on additional evasion techniques as required.

Device spoofing is a means of Invalid Traffic, or IVT. Per MRC Invalid Traffic Standards**,** “IVT is defined generally as traffic or associated media activity (metrics associated to ad and content measurement including audience, impressions and derivative metrics such as viewability, clicks, and engagement, as well as outcomes) that does not meet certain quality or completeness criteria, or otherwise does not represent legitimate traffic that should be included in measurement counts.”

Further language in the same guidelines outline that the objective of measurement organizations and their business partners shall be to ensure transparency with respect to where the ad is served from, the device type, and the `User-Agent` receiving the ad.

1.2. Solution

The solution involves implementation of a verification mechanism using the Privacy Pass protocol. Using Privacy Pass, legitimate devices can assert themselves as authentic with the help of the device manufacturer. The device manufacturer, acting as a trusted party, attests to the authenticity of their devices. This is termed Device Attestation, and it can be used to verify that ad impressions are being rendered on authentic devices and to uncover Sellers of spoofed inventory. The mechanism operates in the measurement context of the ad lifecycle and applies to all creative types. The collected, aggregate signals may be used to evaluate Sellers and may be applied pre-bid to prevent ad delivery on Sellers of spoofed inventory. While the mechanism depends on the Privacy Pass protocol, it could be extended to additional protocols in the future that offer similar guarantees and capabilities to Privacy Pass.

2. Considerations

2.1. Design Principles

The following principles have guided the design and development of this mechanism. They rationalize certain design decisions that may appear counter intuitive because the typical approach to IVT filtration involves using signals indicative of anomalous activity (negative signals) instead of such a signal that conveys trust (positive signal).

- As with the rest of the OM SDK, no user identity information should be required for the software to run.
- This mechanism leverages the Privacy Pass protocol, as it is a device attestation mechanism already supported by some device manufacturers. Existing device attestation mechanisms may not be automatically compatible.
- Practice data minimization from the onset. This solution design seeks to minimize data assets available even in attestation responses, such as keeping signals simple. Furthermore, there are legitimate reasons as to why a device attestation signal may not be available on demand, so systems should not expect a response from every attestation request.
- Device Attestation cannot be a gating criterion. The intention is to create a positive signal instead of a negative one, which means approaching the problem differently. Coverage of the device attestation signal cannot be assumed to be 100%. For example, legacy active devices may not have the capability. This means the focus will be on data aggregated at the Seller level instead of at the user or device level, and no persistent tracking of devices is enabled. For details on how to use the signal, see section 5. Signals and Reporting.
- To preserve user experience, device attestation should not be done pre-bid. This, too, feeds into the intention that signal data shall be aggregated at a Seller-level.

- In support of prior principles, sampling should be the default, instead of requesting attestation on every impression. Sampling will also allow for robustness of signal at the aggregated Seller-level.
- OM SDK should enable independent verification of the device attestation signal on ad impressions similar to impression/viewability measurement.

2.2. Device attestation mechanism qualifications

2.2.1. Defining device attestation

In this context, we define device attestation as an assertion made by a device manufacturer that their device is authentic. This is distinct from self-attestation or self-certification, in which the device asserts its authenticity on its own. The threat model assumes that the device can be compromised, or that the device OS can be run on an emulator while preserving self-attestation capabilities.

Device attestation necessarily involves the device manufacturer in the process so that there is a device authenticity check (verification) every time an attestation is requested. Even if a bad actor compromises an authentic device and harvests tokens from it to replay from bots or other sources, they will need to scale the operation to meaningfully benefit from it. This is because tokens are cryptographically bound to challenges issued by Verifiers, so they can only be replayed to the same Verifier, which significantly limits the utility of the harvested tokens. The tokens would also need to be replayed within a specific time window to be considered valid, since the challenge is temporal in nature. Also, due to potential rate limits applied by the underlying Privacy Pass implementations and by Attesters to safeguard against such token-harvesting attacks, the number of tokens that can be generated using a single device for the purposes of harvesting them is not unbounded. These safeguards have adjustable thresholds that can theoretically be calibrated based on contextual data, so Verifiers, Clients, Attesters, and Issuers can take differential action based on anomalous counterparty activity.

2.2.2. Root of trust

For robust security, there should be a hardware-backed root of trust (RoT) used by the device for authenticating itself with the Attester. This is typically implemented through a Trusted Platform Module (TPM) that hosts a device-specific private key. The Client-Attester protocol must involve the device using its unique private key for signing arbitrary data/challenges that the Attester can verify. The Client-Attester protocol should not result in an attestation of an emulator running the device OS.

3. Roles and Responsibilities

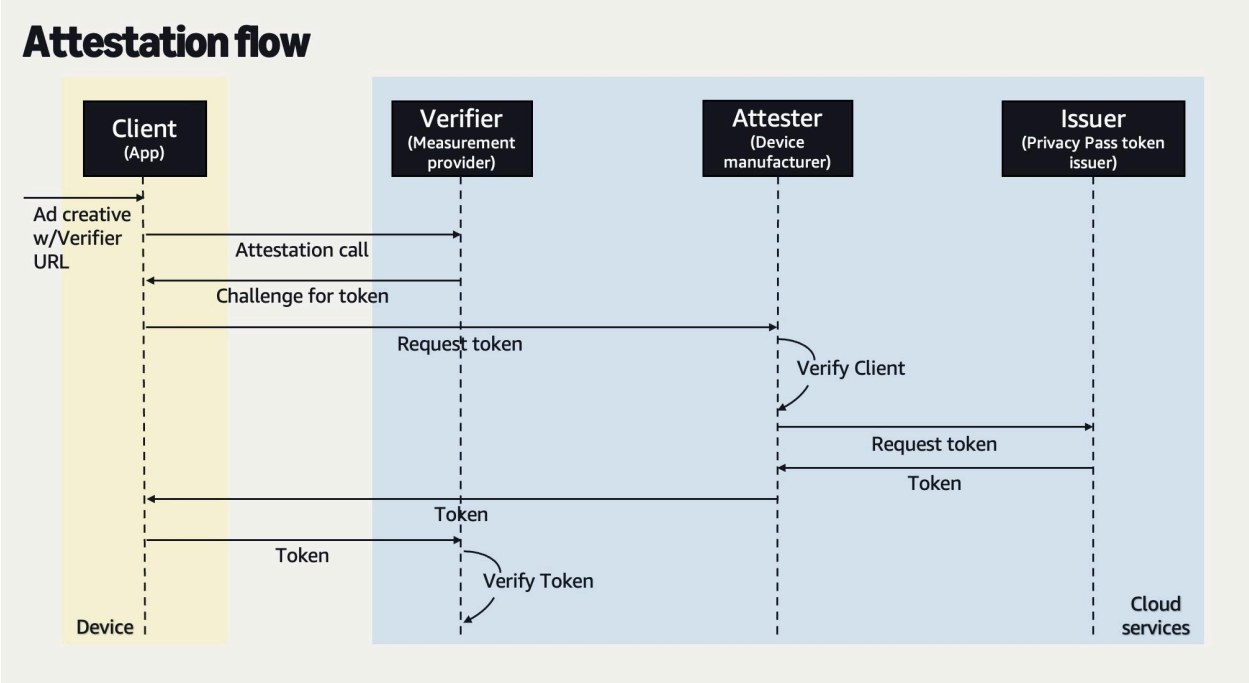
The device attestation mechanism operates purely in the measurement context and is intended to be used in a sampled manner to uncover Sellers of spoofed device inventory. The mechanism intentionally adopts a restrictive approach in terms of prescribing the Privacy Pass

protocol as the underlying protocol to be used for device attestation in addition to specific deployment models, roles, and privacy + security properties, the specifics of which are detailed in the roles section below. This is because industry trust in the mechanism hinges on the mechanism operating robustly across as many devices as possible at a certain minimum bar when it comes to performance, privacy, and security. A weak implementation that has an exploitable vulnerability jeopardizes trust in the overall mechanism and can be disastrous to broad adoption. Over time, once real-world data becomes available, these policies can be updated to support additional device attestation protocols and modalities as appropriate.

There should be minimal dependency on the good intentions of supply chain participants. Effectively, parties with an incentive to subvert this mechanism should not be able to exercise control over it.

3.1. Roles

The Privacy Pass protocol defines 4 roles – Client, Origin, Attester, and Issuer. Since the Origin role is primarily intended to be played by an entity responsible for verification, we rename it to Verifier to facilitate understanding and recollection. Effectively, that makes the roles Client, Verifier, Attester, and Issuer respectively. Role descriptions and examples in the OM SDK-enabled device attestation context follow.



The Attestation flow begins with an ad creative with a verification script being delivered to the Client on which the impression will be rendered. The verification script will contain the URL to the Verifier, which will be passed by the verification script to OM SDK via an Attest API call. OM SDK will be responsible for subsequent calls from the Client.

3.1.1. Client

The Client is an app/video player that seeks to demonstrate that the underlying device it's running on is authentic. The Client will need to integrate OM SDK either directly or indirectly (via an Ad SDK) to get the Privacy Pass capability. Clients (and the underlying devices) trust the Attester to protect the user's private data through the device attestation process.

3.1.2. Verifier

The Verifier is a measurement/verification service that seeks to verify that Clients it communicates with are running on authentic devices, using the Privacy Pass protocol. Typically, measurement/verification providers will play the role of the Verifier, but any entity that has a need to verify measurement data can play the role of a Verifier. That includes ad servers, DSPs, and SSPs. Verifiers are expected to challenge Clients, verify tokens, and aggregate the attestation signals across relevant dimensions.

Verifiers need to be aware that various Attesters may have different approaches to rate limiting in order to manage the number of device attestation requests. Verifiers should not assume that this methodology is the same across all Attesters.

3.1.3. Attester

The Attester is the device manufacturer that attests to the authenticity of their devices. The Client-Attester protocol for device verification may be proprietary, but the Attester also communicates with the Client and the Issuer over the Privacy Pass protocol.

3.1.4. Issuer

The Issuer is responsible for signing Privacy Pass token requests (token issuance) from the Attester. Verifiers can work with any Privacy Pass token Issuer onboarded on to the Attesters for relevant devices the Verifiers seek to verify. Verifiers choose Issuers they can trust but a Verifier may also choose to operate in the added role of an Issuer if they are so inclined (refer to section 3.2.2. Joint Verifier-Issuer model).

Issuers should only issue tokens for Attesters if and only if they have validated the attestation mechanism via both review and testing. The Issuer evaluates how the attestation mechanism reflects the Attester's definition of an authentic device, and how the attestation mechanism guarantees that a device is authentic by that standard.

As part of examining the attestation mechanism, the Issuer should identify the signals used by the attestation mechanism and determine how those signals might be faked or modified. The Issuer should use this information to create tests from both real and fake devices, to check that the Attester's device attestation implementation meets stated requirements in 4.2.1. On-boarding to Attesters.

The Issuer is not responsible for determining the validity of any given issuance request, since the Issuer will not have access to any Client-specific information. The Issuer instead is responsible for determining the validity of the attestation mechanism.

3.2. Deployment models

The Privacy Pass protocol supports various deployment models, but this mechanism prescribes two for maintaining privacy and security:

3.2.1. Split Verifier-Attester-Issuer model

The entities playing the Verifier, Attester, and Issuer roles are all distinct. An example deployment would be a measurement provider as a Verifier, a device manufacturer as an Attester, and an independent Privacy Pass token Issuer. In examples existing upon publication of this document, CDNs (content delivery networks) have acted as independent Privacy Pass token Issuers.

3.2.2. Joint Verifier-Issuer model

The same entity plays the roles of Verifier and Issuer. An example of this is an Ad Verification company playing the role of both the Verifier and the Issuer. The device manufacturer plays the role of the Attester. Ultimately the Attester selects the Issuers that they onboard, but this model offers more control of the chain of trust to the Verifier. Per section 4.3 of RFC 9576 (Privacy Pass Architecture), Issuers that produce tokens for only one Verifier are not suitable in this model, since an Attester can potentially infer the Verifier from a token request.

4. Deployment Guidance

4.1. Verifier Guidance

4.1.1. Challenging devices

Verifiers can choose to challenge Clients randomly against Attestation Requests or apply intelligence based on various approaches, including information it may have about the specific Seller or Client in question. The attestation signal confidence at a Seller level can be built over time and a sliding window can be maintained around Seller-level attestation data, issuing challenges to maintain signal confidence over time. Since device-specific rate limits on responses to challenges is not public information, it is possible that responses to challenges will not be received when rate limited, which should become apparent over time.

Refer to section 3.2. Deployment models for details. Verifiers may operate in the single or joint model. If the Verifier is operating in the single model, the expectation is that the Verifier tests the Issuer on an annual basis.

In order to limit how long a challenge will be accepted by the Verifier, it is recommended to include the `max-age` token challenge parameter with a value of 2 minutes as defined in section 2.1.2 of RFC 9577 (Privacy Pass HTTP Authentication Scheme).

4.1.2. Verifying tokens

Since this mechanism operates in the measurement context, there is no urgency to verify tokens in real-time. Tokens can be collected and verified offline. When token challenges are issued with the `max-age` parameter as defined in section 4.1.1. Challenging devices, tokens should only be considered valid if they have been received before the time limit defined in `max-age`. The Verifier should consider successful token challenges received before the defined `max-age` value to be valid. A token received after the `max-age` will be considered a Missing Token.

4.1.3. Privacy Considerations

Device-level information such as Device ID will not be available through OM SDK. To maintain user privacy, any macros used in the Attestation Request should also not communicate device-level or user-level information to the Verifier.

4.2. Issuer Guidance

4.2.1. On-boarding to Attesters

Issuers should assure themselves of Attesters being able to robustly verify their devices for authenticity before issuing tokens to Attesters requesting them on behalf of Clients. Issuers should also periodically verify the robustness of the Attester-Client protocol through independent verification approaches.

Issuers should re-review attestation mechanisms at least once a year. Issuers should not issue tokens for any attestation mechanisms that require user interaction for the attestation flow to be completed. To be clear, the user should not have to actively participate in the ad experience for the attestation mechanism to complete.

While Issuers hold Attesters accountable, Verifiers hold Issuers accountable. More guidance for Verifiers can be found in section 4.1. Verifier Guidance .

4.2.2. Issuing tokens

Issuers should be able to support both privately-verifiable and publicly-verifiable tokens per the Privacy Pass specification. Issuers should also implement rate limits to prevent abuse. Issuer rate limit thresholds are out of scope for this document. Issuers should only issue tokens to Attesters using trusted methods that align with the Privacy Pass specification.

4.2.3. Attester communication

Issuers should communicate with Attesters over a mutually authenticated TLS connection. If possible, certificate pinning should be employed for better security.

4.3. Attester Guidance

4.3.1. On-boarding Issuers

Attesters should ensure that Issuers they on-board issue tokens for multiple Verifiers and not just for one Verifier. While this prevents the Attester from inferring which Verifier a token is being requested for, it also protects user privacy by preventing a proliferation of Issuers which can be individually used to track users.

5. Signals and Reporting

5.1. Signals available for Verifier

Each of these signals is intended to be associated with an ad impression. Since downstream measurement events such as clicks, views, conversions, etc. are also associated with impressions, these signals can be cascaded to those events as appropriate.

- **Attestation Request:** The Verifier received a request from a Client attempting the device attestation mechanism. The Attestation Request carries an impression identifier that must be used for deduplication purposes. Against one impression ID, only one attestation request may be counted.
- **Challenge Issued:** The Verifier challenged a Client to prove the authenticity of the underlying device.
- **Successful Token Verification:** The Verifier was able to successfully verify the token per section 2.2.3 of RFC 9577 (Privacy Pass HTTP Authentication Scheme), indicating an authentic device.
- **Failed Token Verification:** The token verification failed because the expected token did not match the received token per section 2.2.3 of RFC 9577 (Privacy Pass HTTP Authentication Scheme).
- **Missing Token:** The Client failed to present a Token to the Verifier in the expected timeframe. This could be due to device spoofing, rate limiting, token expiration, or other failures.
- **Other Errors:** Errors could be from crypto operation failure, transient system issues, or incorrect implementation of the Privacy Pass RFC by the Verifier. Errors encountered by non-Verifier entities are out of scope, as are Failed Token Verifications.

For a given impression, there may be multiple Verifiers challenging the Client simultaneously. Due to the sampling-based approach (as discussed in section 5.5 Sampling), a Verifier may also not get the opportunity to challenge or may not receive a token in response to a challenge. Effectively, this means that attestation-based metrics should not be expected to reconcile across distinct Verifiers. Similarly, metrics derived from the attestation signal should also not be expected to reconcile across distinct Verifiers. For more details on providing transparency through metrics, see section 5.6. Metrics.

5.2. Token availability expectations

Token-bearing responses to challenges cannot be always expected due to various legitimate reasons. Devices could choose to ignore challenges for unspecified reasons, Attesters or Issuers could be unreachable or rate-limiting requests, etc. Therefore, transient unavailability of tokens should not be considered anomalous. However, when attestation signals are aggregated at Seller-level dimensions to generate statistically-significant datasets, anomalies pertaining to token unavailability can be considered notable.

5.3. Aggregating the attestation signal

The device attestation signals do not carry any device-level identifier that can be used to block individual devices or to track them over time. The signals are intended to be used in an aggregated manner to evaluate Sellers for signs of spoofed device inventory. The signals should be aggregated at different dimensions that represent the flow of ad spend, such as the seller_id field in Supply Chain Object. Depending on the supply chain there may be other identifiers such as Publisher ID / Exchange ID / etc. that could also serve as dimensions to aggregate the signals across.

5.4. Layering data from additional contexts

The measurement context typically communicates more information from the Client to the Verifier (typically the measurement endpoint) that can be layered with the device attestation signal for cascading the signal across multiple ad impressions, determining when to challenge the Client, associating with supply chain data, and more.

While the impression ID is communicated as part of the Attestation Request from the Client to the Verifier, approved macros, or out-of-band mechanisms to join the impression ID with OpenRTB's Supply Chain Object, can be used to communicate additional supply-side and demand-side dimensions. From the HTTP request, the Verifier may also extract the Client IP address and User-Agent information and use it for sampling challenges.

5.5. Sampling

5.5.1. Client-Verifier Attestation Request sampling

Since device attestation involves multiple entities, optimizing for resources is crucial to safeguard user experience. The mechanism's primary goal is to build and continuously maintain confidence in device authenticity of inventory from distinct Sellers. To achieve this efficiently, it relies on a sampling-based approach from the outset. This ensures that valuable resources on user devices are not wasted and the user experience is preserved. The sampling itself is fair and resistant to manipulation: for each Verifier on a device attestation-supported impression, OM SDK makes an Attestation Request to the Verifier based on a predefined probability. This allows each Verifier to collect signals to maintain a Seller-level rolling window of confidence. This predefined probability will be adjusted based on real-world learnings as appropriate.

5.5.2. Verifier-Client Challenge sampling

Verifiers need to build confidence in device authenticity of inventory from distinct Sellers in an ongoing manner. They can use a simple probability-based sampling approach similar to how OM SDK samples Attestation Requests to the Verifier. They can also use in-band and out-of-band contextual data to determine whether to respond with a Challenge on an incoming Attestation Request. Some examples of such contextual data would be device type, signal confidence by Seller, Client User-Agent, etc. The sampling rate can be dynamically adjusted by the Verifier to maintain a sliding window of the device attestation metrics at high confidence. The sampling rate may also need to be adjusted based on out-of-band feedback from Attesters or in case of rate limiting by Attesters (if detectable and if deemed necessary).

5.6. Metrics

These metrics are measured at the Verifier. For the rate metrics, Verifiers should establish minimum thresholds of statistical significance to qualify metrics as reportable against specific dimensions to prevent misinterpretation.

5.6.1. Attestation Eligible Impressions

To ensure the accuracy of attestation-based metrics, they must be calculated against a qualified baseline of `Attestation Eligible Impressions`. This practice prevents results from being skewed by inventory that does not support device attestation. OM SDK signals the availability of device attestation support to the verification script at the start of each ad session. Refer to section 7. Appendix for examples.

Attestation Eligible Impressions = Impressions from device attestation-supported ad sessions

At minimum, Verifiers should provide this metric at the `Gross` level (i.e. `Gross Attestation Eligible Impressions`), but they may choose to provide at various filtered levels, such as `Net`

(General Invalid Traffic filtration applied), or `Total Net` (General and Sophisticated Invalid Traffic filtration applied).

At a publisher or app level, this metric can be used to track adoption of this mechanism and identify coverage gaps so that they can be improved over time. When analyzed at the Seller level for a specific app, a significant and unexpected lack of `Attestation Eligible Impressions` compared to other Sellers for that same app can be an indicator of anomalous inventory that may not be originating from the claimed source.

5.6.2. Attestation Attempted Rate

The Attestation Attempted Rate is the ratio of Attestation Requests to `Attestation Eligible Impressions`.

Attestation Attempted Rate = Attestation Requests / Attestation Eligible Impressions

The primary application of this metric is to identify Seller-specific anomalies that may indicate signal suppression (for example, the verification JavaScript being suppressed). The key principle is that the sampling rate for Attestation Requests is determined by OM SDK and not by individual publishers or Sellers.

5.6.3. Attested Impression Rate

The `Attested Impression Rate` measures the success rate of device attestation challenges, providing a direct signal of device authenticity.

Attested Impression Rate = Successful Token Verifications / Challenges Issued

The denominator, Challenges Issued, is intentionally used to ensure this metric evaluates the outcome only for impressions where an Attestation Request by the Client was responded to with a Challenge by the Verifier. This isolates the outcome of the attestation process from the attestation attempt frequency, which is important because Verifiers can choose to challenge Clients in a sampled manner.

This metric's primary purpose is to provide buyers with transparency into the integrity of inventory from a given Seller. When observed across Seller-level dimensions, a consistently high Attested Impression Rate indicates a Seller with authentic devices that support device attestation. Conversely, a low rate can signal spoofed devices or other issues preventing successful attestation, warranting further investigation.

5.6.4. Error Rate

The error rate helps to provide clarity on the number of attestations failing due to reasons other than failed token verifications. Possible errors are defined in section 5.1.

Error Rate = Errors / Challenges Issued

5.7. Auditing Metrics

5.7.1. Auditing Attestation Eligible Impressions

To provide an auditable measure, Verifiers can reconcile `Attestation Eligible Impressions` against their own measure of impressions that should be device attestation-capable. This requires capturing and checking two conditions for each eligible ad session:

- **OM SDK Version Requirement:** The impression originates from an app running OM SDK known to support device attestation, which would be versions 1.6 or higher. For the native SDK version, use data.context.app.libraryVersion on sessionStart events. For the JS SDK version, use data.context.omidJsInfo.serviceVersion on sessionStart events. The version is also the key used for baselining the `Attestation Attempted Rate` (section 6.2 Establishing Baselines for Normalization).
- **Supported Environment:** The Client device's operating system and hardware are known to provide the necessary support for the underlying Privacy Pass-based device attestation protocol. The device information can be obtained from the deviceInfo node within the sessionStart event. See examples in 7. Appendix for more details.

This process provides an additional layer of assurance in the integrity of the reported metrics.

5.8. Feedback loop and error handling

There is no built-in feedback loop mechanism within the Privacy Pass protocol to communicate false positives and false negatives. If a Verifier or Issuer, through an out-of-band mechanism, determines that tokens were issued to fake devices, they should communicate directly with the Attester and reconcile log-level data to debug the issue. The mechanism is resilient to some degree of errors due to the metric aggregation-based approach to determine anomalies.

6. Using the Signals

Identifying potentially anomalous Sellers involves a three-stage analytical process: assessing the volume of measurable inventory, establishing baselines for key metrics, and interpreting Seller-level metrics against those baselines.

6.1. Assessing the Volume of Measurable Inventory

The `Attestation Eligible Impressions` metric provides the key signal for analysis. As an indicator of inventory where device attestation is technically possible, it is useful for tracking adoption of this mechanism across publishers and apps over time.

While not a direct indicator of potentially fraudulent activity, significant discrepancies in this metric across different Sellers for the same app can highlight anomalies that may warrant further review.

6.2. Establishing Baselines for Normalization

Since the sampling rate for Attestation Requests is determined by OM SDK and can potentially change over time for new OM SDK versions, Verifiers must first establish baselines for the `Attestation Attempted Rate` and the `Attested Impression Rate` metrics for each environment-specific OM SDK version (for example, OM SDK for iOS). This normalization is essential to correctly interpret Seller-level metrics and is also useful to account for any variables that may cause metric deviations for legitimate reasons.

6.3. Interpreting Seller-level Metrics

By comparing Seller-level metrics against the baselines, Verifiers can potentially identify distinct patterns that uncover Seller-level anomalies that may warrant deeper investigation. When a Seller's `Attestation Attempted Rate` for an environment-specific OM SDK version is significantly lower than the baseline, it potentially indicates signal suppression by the Seller. When a Seller's `Attested Impression Rate` for an environment-specific OM SDK is significantly lower than the baseline, it potentially indicates device spoofing by the Seller.

7. Appendix

Example of sessionStart event for FireTV

```
{
  "sessionStart":{...,
    "data":{...,
    "supportedAttestationMechanisms":[
      {
        "mechanism":"FireTVFOSDAT",
        "version":"1.0"
      }
    ],
  }
}
```

Example of Attestation Mechanism in sessionStart event for Safari

```
{
  "sessionStart":{...,
    "data":{...,
    "supportedAttestationMechanisms":[
      {
        "mechanism":"ApplePAT",
        "version":"default"
        "executionEnvironment":"native"
      }
    ],
  }
}
```