Why CTV Ad Fraud Keeps Working: A Verifiability Analysis of the Connected TV Supply Chain

Aleksander Sekowski

Independent researcher

Preprint, August 2026. Version 1.3.

This is the fourth paper in a series applying machine-checkability analysis to advertising protocols, after *VAST XML Validation at Bid-Time Scale* (doi:10.13140/RG.2.2.11404.27520), *How Machine-Checkable Is OpenRTB?* (doi:10.13140/RG.2.2.27937.57448), and *Measuring OpenRTB Dialects in Client-Side Header Bidding* (doi:10.13140/RG.2.2.26572.78720).

All codings, the fraud corpus, the mitigation matrix, the crawl code and its frozen snapshots, and every analysis script are at https://github.com/aleksUIX/ctv-verification-gap. Each number in this paper is produced by one of those scripts and can be recomputed. Ten primary disclosure PDFs are preserved privately (vendor copyright); the repository publishes a manifest of filenames, creation dates, and what each establishes.

Abstract

Connected TV advertising fraud is treated as a detection problem and priced as an unavoidable tax. We argue it is a predictable consequence of a verification architecture, and we measure that architecture field by field.

We define a Verifiability Index that classifies every field in an ad transaction that makes a claim about the world into one of three classes: attested, where a mechanism cryptographically binds the value to an accountable party independent of the party the value benefits; cross-referenceable, where a public machine-readable reference exists that can reject a false declaration; and self-declared, where the receiver has only the sender's word. The index is evaluated from the receiver's vantage at or after transaction time, which is what makes it channel-sensitive: server-side ad insertion removes the buyer's client-side observation point, and client observability is the assumption every web anti-fraud mechanism was built on.

Applying the index to OpenRTB 2.6 and VAST 4.3, we find that 91.5% of pricing-relevant claims a buyer receives about a CTV server-side impression are self-declared, against 70.0% for the same claims on web display (Table 1). No scored OpenRTB field on either path is attested. Device Attestation in OM SDK 1.6 is A-independent where a client still runs, and unavailable on the server-side path we score.

Web's advantage is almost entirely post-hoc: 22 of its 27 checkable pricing-relevant request-side fields (81%) depend on observing the render, which yields ground truth only after the auction has cleared. Neither channel can verify device IP, user agent, or device type from the payload and public files before the auction clears. On that strict (registry-only) tier, web moves from 70.0% to 94.4% self-declared and CTV stays at 91.5%, because every check CTV retains is already a registry lookup. Restricted to fields present on both paths the comparison does not invert (web 94.3%, CTV 94.3%). The durable result is the missing observation, not a bid-time ranking. On web, ground truth eventually arrives and closes a correction loop through discrepancy claims, make-goods and blocklisting. Behind server-side ad insertion it never arrives at all.

We then assemble a corpus of 17 named CTV fraud schemes publicly disclosed between 2018 and 2026 and map each to the fields it falsified. Sixteen of 17 falsified at least one self-declared field under all mapping

grades; restricting to fields the discoverer stated explicitly, the bound is 12 of 17. The session-defining core they relied on (IP address, device type, make, advertising identifier) is without exception self-declared. The single all-grades exception does not lie about the session at all: it strips the measurement tags from the delivered advertisement and fires them elsewhere, which is a threat our framework does not model and we say so.

Against a counterfactual in which ads.txt, app-ads.txt, sellers.json, schain and ads.cert 2.0 are all adopted and enforced at 100%, the result depends on a question the counterfactual has to state rather than assume. Scored against the app identity each scheme actually declared, the standards reject 12 of the 17. Scored against the same fabrication pointed at apps the operator owns and authorizes itself, all 17 survive, and that path is not hypothetical: three corpus schemes already used it, and the vendor that discovered most of this corpus measured the fake-CTV-app population at over 1,300 apps in 18 months. The authorization layer, fully adopted, would have forced most of this corpus to change which app it claimed to be and stopped none of it from fabricating sessions.

Inverting the analysis ranks the absent mechanisms by leverage: per-impression signed delivery would break 14 of the 17, device attestation 13, app-instance attestation 12, against 12 for the best deployed standard on the reading most favourable to it and 0 on the other. Two schemes are broken by nothing in the analysis, and they mark its boundaries: one whose every declared value was true while the audience was not, and one that attacked the measurement rather than the declaration.

A crawl of the Roku channel store (n=400, uniform random from a complete 31,647-channel frame) finds that the app-ads.txt authorization walk cannot begin for 89.0% of channels, because their store listings publish no developer URL for a crawler to follow, and completes for only 8.8%. On a purposive sample of 37 well-known channels the walk completes for 78.4%; restricted to the 31 that sell open programmatic inventory it completes for 93.5%. Where a developer URL exists, 79.5% of those walks find a published file, so the dominant failure is the store-listing bootstrap rather than publisher-file compliance.

Finally we audit AdCP 3.1.1, the emerging agentic advertising protocol, with the same instrument. It improves seller authorization and introduces a signed mechanism by which a brand can authoritatively deny owning a claimed CTV app. It does not attest delivery, and it rules per-impression attestation permanently out of scope in its own specification text. It does require receiver-verified, issuer-anchored attestation for whether a viewer is a unique human over 18. The industry can build the mechanism this analysis ranks first. It has pointed one at personhood and none at delivery.

All codings, the fraud corpus, the mitigation analysis, the crawl code and frozen snapshots, and every analysis script are published so each number can be recomputed.

Keywords: connected TV, ad fraud, OpenRTB, VAST, server-side ad insertion, protocol verification, supply chain transparency, attestation

1. Introduction

A buyer purchasing a connected TV impression is told a great deal: which app was open, what content was playing, which device was in the living room, where that living room is, and that the advertisement played to completion. Almost none of it can be checked.

This is not a claim about negligence or about weak enforcement. It is a claim about architecture, and the mechanism is simple enough to state in one paragraph. Web display advertising evolved a verification model that rests on the buyer, or a vendor acting for the buyer, executing code in the same context where the advertisement renders. That code sees the socket the request arrived on, the headers the browser sent,

the page around the ad, and the geometry of the ad slot. Declarations that contradict what the code observes can be rejected. Server-side ad insertion, the dominant delivery architecture in CTV, removes that code. The advertisement is spliced into the video stream before it reaches the television, and the request that reaches the exchange did not originate at the device. The buyer never observes the client. Every fact about the client becomes a claim relayed by an intermediary that is paid according to those claims.

One qualification belongs here rather than in a later caveat, because the rest of the paper depends on it. Server-side insertion is not a single deployment. Whether the tracking beacons also fire server-side is a configuration choice: AWS Elemental MediaTailor ships server, client and hybrid reporting modes, Google's dynamic ad insertion runs through a client-side SDK, Roku's advertising framework fires impression and quartile events from the client, and the IAB's own SSAI macro guidance defines a `[SERVERSIDE]` macro precisely so a receiver can tell which case it is in. The Media Rating Council, whose 2021 guidance on this subject is the normative document, observes that SSAI inventory on major platforms tends to be enabled for client-initiated measurement. So the loss of the client vantage is real but conditional, and where beaconing is client-initiated the receiver recovers some observation. Our analysis is scored for the case where it is not, we say so at each point where it matters, and Section 11 treats the distribution of deployment modes as the most important measurement this paper does not make.

The consequence is not that CTV is harder to police. It is that the mechanisms the industry deployed in response to a decade of web fraud address a different problem than the one CTV presents. ads.txt, app-ads.txt, sellers.json and the supply chain object all answer a question about authorization: may this seller offer this inventory. None of them answers the question CTV fraud turns on: did this impression happen at all. A scheme that fabricates sessions inside inventory it is authorized to sell satisfies every one of them.

The cheapest illustration is an operation that dispensed with the fraud entirely and just wrote the requests. In August 2021 Method Media Intelligence disclosed RapidFire: no botnet, no malware, no fraudulent apps, a script emitting counterfeit JSON bid requests straight into open CTV auctions across multiple supply-side platforms, reportedly earning its operators around ten million dollars a month. Every other scheme in our corpus builds infrastructure whose only purpose is to make fabricated traffic look observed, and one of them goes as far as pairing a spoofed user-agent string with a matching TLS cipher configuration so that the declared device and the observed encryption behaviour agree. RapidFire skipped all of it. On a delivery path where the creative goes to a stitching server and no render signal is ever required back, that infrastructure buys nothing, because there is nothing downstream that could contradict the claim. This paper is an attempt to say precisely which claims those are, and to count them.

1.1 Contributions

1. **The Verifiability Index** (Section 3), a field-level instrument that is recomputable from specification text, distinguishes three scoring tiers (a mechanism that is specified, one that is operated, and one that can actually be executed against the inventory in question), and separates attestation by an independent party from attestation by the interested party. We apply it to OpenRTB 2.6 and VAST 4.3.
2. **A measurement of the CTV verification deficit** (Section 5): 91.5% of pricing-relevant request-side claims are self-declared on the CTV server-side path against 70.0% on web (Table 1). No scored OpenRTB field on either path is attested; Device Attestation in OM SDK 1.6 is A-independent where a client still runs and class C behind server-side insertion. Response-side claims are more checkable on both channels (17.4% self-declared across all 23 coded assertions, 50% on the eight settled rows), which localizes the deficit to delivery context rather than creative.
3. **A corpus of 17 named CTV fraud schemes** (Section 6) mapped field by field, with each mapping flagged as documented by the discoverer, inferred from the described mechanism, or documented at family level only, and with a rejection reason published for every candidate we excluded.

4. **A counterfactual mitigation analysis** (Section 7) reported under two named readings, because a single number there is not well defined: the deployed standards reject the app identity 12 of the 17 schemes declared and reject none of the fabrication mechanisms behind them. An inverted ranking identifies per-impression signed delivery as the single highest-leverage absent mechanism, at 14 of 17.
5. **An empirical reachability measurement** (Section 7.3): the app-ads.txt walk cannot start for 89.0% of the Roku channel population and completes for 8.8%, with the failure localized to the store-listing bootstrap.
6. **An audit of the agentic layer** (Section 9): AdCP 3.1.1 does not attest delivery and declines to do so permanently, while requiring verified attestation of viewer personhood.

1.2 What this paper does not claim

We do not measure fraud prevalence. We produce no estimate of money lost, and we deliberately quote vendor figures with their claimant attached rather than generating our own, because a synthetic dollar total would be the most cited and least defensible number in the paper. Note also that essentially every dollar figure in this literature is vendor-modelled inventory value at an explicitly assumed $20 CPM rather than audited loss, so those figures are not comparable across vendors and we never sum them.

We do not claim the corpus is a census, and Section 11 quantifies how far it is from one using the discovering vendors' own aggregate claims. We do not claim that the schemes in it are all the CTV fraud there was, only that they are what has been publicly disclosed with mechanism detail.

We do not claim that attestation would end CTV fraud. Three corpus schemes are caught outright by none of the attestation mechanisms we consider, and two are caught by nothing in the analysis at all. Section 8.2 explains why, and the short version is that attestation answers whether a session happened, not whether it was worth what was paid for it.

---

2. Background and related work

2.1 The CTV transaction

A connected TV advertisement is bought through the same two protocols as a web video advertisement and delivered through a different architecture.

The bid request is an OpenRTB object. Where a web request carries a `site` object describing a page, a CTV request carries an `app` object describing an application, and this substitution is more consequential than it appears. A site is identified by a domain, which is a name in a global namespace with an owner, a registrar, and a file-serving capability that later mechanisms build on. An app is identified by a `bundle` value, which is platform-specific by construction: a package name on Android TV, a numeric store identifier on Roku, an ASIN on Fire TV. There is no namespace shared across platforms and, on some platforms, no relationship at all between the identifier and any domain its owner controls. The request also carries a `content` object describing what is playing, which has no analogue in web display buying and which prices a large share of CTV inventory.

The creative response is a VAST document. In video advertising the response is rarely the advertisement itself: it is commonly a `Wrapper` naming another VAST endpoint, which may name another, until an `InLine` element finally supplies media files and tracking URLs. Whoever resolves that chain observes every hop.

Delivery is where CTV diverges. Two architectures exist. In client-side insertion the application's player requests the advertisement, receives the VAST document, and renders the media itself, so a software development kit on the device is in the loop and constrained device-side measurement is possible. In server-side ad insertion, which dominates premium and live CTV inventory, an intermediary assembles content and advertising into a single continuous stream before it reaches the television. The stitcher receives a signal that an advertisement break is beginning, requests advertisements for the pod, resolves the VAST chain, fetches and transcodes the media files so they match the content stream's encoding profile, splices them into the manifest, and depending on how it is configured may fire the tracking beacons itself, hand them to the client, or split them. Section 4 treats that configuration as a variable, because the buyer's verification position depends on it more than on anything else in this paragraph.

The reasons this architecture won are good ones and they are not about fraud. A single stream is resistant to client-side ad blocking, avoids the buffering and player-state transitions that make client-side insertion visibly worse on a television, and works across a fragmented device population where a publisher cannot rely on any particular player capability. It also happens to remove every point at which a buyer could observe anything, and that side effect is what this paper measures.

The industry noticed the problem this created and standardized a response, and the shape of that response is the paper's thesis in miniature. Because a stitcher's request originates from a data centre, its traffic looks exactly like the datacenter traffic that fraud filters reject. VAST 4.1 (November 2018) therefore defines request headers, `X-Device-IP`, `X-Device-User-Agent`, `X-Forwarded-For` and `X-Device-Referer`, that the stitching service is expected to populate from the player's incoming request, specifically so that legitimate server-side traffic is not mistaken for fraud. Subsequent guidance built on the same premise: IAB Tech Lab published SSAI-specific VAST macros in November 2020 and guidelines for CTV and OTT device and app identification, and the Media Rating Council issued SSAI and OTT measurement guidance in August 2021.

Note carefully what that mechanism does. It does not restore the buyer's observation. It gives the stitcher a standardized place to state what the buyer can no longer see. The industry's answer to losing the observation point was to formalize the declaration, and the headers are trusted for the same reason the fields inside the bid request are trusted, which is that there is nothing else. ICEBUCKET, in 2020, falsified precisely these headers.

2.2 The declared-integrity layer

Between 2017 and 2022 the industry deployed a family of mechanisms in response to domain spoofing and unauthorized reselling. It is worth enumerating them with attention to the question each answers, because the uniformity of that question is the argument of Section 7. One correction to the framing before starting: the motive is not uniform even if the question is. ads.txt and app-ads.txt were designed against spoofing and say so, the app-ads.txt specification naming the case where "the domain of the webpage, or the ID of the mobile app has been falsified." sellers.json and the SupplyChain object were not. Their stated gap is that ads.txt authorizes sellers without revealing who they are or which intermediaries handled the request, which is a transparency problem rather than a spoofing one.

ads.txt (IAB Tech Lab, June 2017) has a publisher place a file at a well-known path on its domain listing the advertising systems authorized to sell its inventory, with the account identifier used and whether the relationship is direct or through a reseller. A buyer that sees inventory offered for a domain by an account absent from that domain's file can reject it. The question answered: **may this seller sell this domain's inventory?**

Its version history matters here because two later revisions are directly on this paper's subject and are routinely omitted. 1.0.1 (August 2017) added `SUBDOMAIN`; 1.0.2 arrived in March 2019; **1.0.3 (March**

2021) added `INVENTORYPARTNERDOMAIN`, introduced explicitly for CTV, and it is in our own list of affected fields; **1.1 (August 2022) added `OWNERDOMAIN` and `MANAGERDOMAIN`**, which answer who owns and who monetizes a property. The current version is 1.1. That last revision bears on Section 9.5, where we credit AdCP with introducing a way to ask who owns a claimed property: the unsigned ancestor of that question has existed since 2022, and AdCP's genuine advance is the signature and the property scoping rather than the question itself.

app-ads.txt is a separate specification, published 13 March 2019 and still at version 1.0, and the discovery procedure lives in it rather than in any ads.txt revision. It extends the same idea to applications, and must solve a bootstrap problem in doing so, because an app has no domain of its own to host a file at. The specification's answer is indirection through the store listing: a crawler reads the developer's website URL from the app's store page, then derives the `app-ads.txt` path from that domain. That indirection is the mechanism's load-bearing assumption, and Section 7.3 measures how often it holds. The question answered: **may this seller sell this app's inventory?**

sellers.json (2019) inverts the direction. Each advertising system publishes the sellers it represents, with identifiers, domains, and whether each is a publisher, an intermediary or both, or is declared confidential. Combined with the **SupplyChain object** (`schain`), which records the chain of intermediaries a bid request passed through, a buyer can walk the declared path and check each hop against the named system's own published list of sellers. This pair is the most complete transparency mechanism the industry has deployed, and Section 7.4 shows it working on the web. The question answered: **who are the parties in this supply path, and does each name the next?** SupplyChain version 1.1, announced 23 June 2026 with comment open to 21 August, adds technical-custody reporting so that SSAI platforms, ad servers, SDKs and wrappers become visible in the chain, which is the standards body acknowledging the gap described in Section 4.

Two CTV-specific identity guidance documents belong in this list because a practitioner will otherwise raise them as counterexamples. IAB Tech Lab's *OTT/CTV Store Assigned App Identification Guidelines* (August 2020) specify that the app identifier carried in the bid request should be the store-assigned identifier, which is why the correct term for the CTV case is store ID rather than bundle, and we use store ID where the CTV path is meant. And the *Guidelines for Identifier for Advertising on OTT Platforms* introduced an `ifa_type` declaration, which supports a genuine consistency check: a declared `rida` on a device that is not a Roku is rejectable on its face. We dismiss that check by name rather than by omission, for two reasons. The guidelines themselves were deprecated in 2023 in favour of an unversioned community extension whose active values are `dpid`, `ppid`, `sspid` and `sessionid1`, with `rida`, `aaid`, `idfa`, `afai` and `msai` all deprecated. And the check falls to the falsification criterion of Section 3.3 in any case: the adversary supplies both the `ifa_type` and the device fields it would be compared against, which is the same structure as the TLS cipher pairing that DoubleVerify documented inside an OctoBot command server.

ads.cert introduces cryptography, and its history is instructive. Version 1.0 (beta draft, November 2018) signed not the whole bid request but a publisher-declared subset of fields, using ECDSA P-256, and was never finalized; the specification now carries a deprecation stub. Version 2.0 (January 2022) has two published protocols, Call Signs and Authenticated Connections. Authenticated Connections is frequently described, including in an earlier draft of this paper, as signing the transport. It does not. It is application-layer message authentication: an HMAC-SHA256 over the request body and URL, keyed by a shared secret derived through X25519 key exchange, with counterparty keys published in DNS. It provides origin authentication and tamper resistance for the message, not merely session security.

One property of that construction matters for our purposes and is easy to miss. Because HMAC is symmetric, either party could have produced the signature, so the assurance is pairwise and non-transferable: only the intended recipient can verify it, and it cannot be forwarded downstream as evidence about an earlier hop. The specification is candid that this "recipient forgeability loophole" means a signature "doesn't have utility within an outside setting". So the question answered is: **is the party directly calling me who it**

claims to be, and has this message been altered in transit? Notably, its own stated motivating case is ours: schemes in which "parties have attempted to impersonate SSAI platforms", which are "challenging to identify, since traffic appears to originate from the same cloud platforms and hosting providers that service real SSAI businesses."

Deployment is close to absent on the open market. The reference implementation's last substantive commit was in 2023, it has no releases, and its README still describes itself as a proof of concept; a probe of the specification's mandated DNS location across fifteen major supply-chain domains, including IAB Tech Lab's own, returned no Call Sign records. The countervailing fact is that TAG's Certified Against Fraud Guidelines v10.1 (July 2025) require Authenticated Connections for SSAI billing notifications from certified companies, so there is mandated adoption inside the certified population even where open-market signal is absent.

The Open Measurement SDK, with its OMID interface, and the `AdVerifications` element in VAST are adjacent, and different in kind: they do not answer an authorization question but provide a hook for independent code to observe a render. They are the closest thing in the incumbent stack to what this paper argues is missing, and their dependence on a client execution context is exactly why they weaken in the architecture described in Section 2.1. Open Measurement SDK 1.5 does cover tvOS, Android TV, Tizen and webOS, so the constraint is the execution context rather than platform support, and OM SDK 1.6 is where device attestation landed in October 2025.

2.2.1 The accreditation layer, which has said most of this already

The mechanisms above are protocol artifacts. Running alongside them is a measurement-accreditation layer that uses a different vocabulary, and a paper that omits it will read to a practitioner as written from outside the field. It also, inconveniently for any claim of novelty, reached several of this paper's conclusions first.

The Media Rating Council classifies invalid traffic in two tiers. **General invalid traffic (GIVT)** is identifiable by routine filtration against known lists: declared bots, data-centre origins, activity that does not attempt to conceal itself. **Sophisticated invalid traffic (SIVT)** requires advanced analytics or human intervention, and its enumerated categories include, at category 5, "server-side ad insertion (SSAI) spoofing" and, at category 6, domain and app misrepresentation including app ID spoofing. An interim update in 2024 extended that explicitly to CTV: the CTV store ID falls under the existing app-misrepresentation requirements, and non-CTV apps misrepresented as CTV are to be treated as SIVT. In the MRC's terms, then, most of the corpus in Section 6 is SIVT category 5 or 6, and the taxonomy predates most of the corpus.

The MRC's *Server-Side Ad Insertion and OTT Guidance* (August 2021) is the normative document on precisely this paper's subject, and it already requires what Section 4 observes: accredited counting must be client-initiated, server-fired-only tracking must be segregated and disclaimed as non-compliant, and SSAI providers must be verifiable or their traffic treated as unknown. It further proposes certifying SSAI providers, on the stated rationale that invalid traffic perpetrators may disguise themselves as SSAI providers.

Two MRC positions are worth quoting against framings this paper might otherwise have adopted. On pre-bid: "MRC does not require pre-bid filtration, in fact, our Standards discourage or caution against it in several instances." That is a direct constraint on any recommendation phrased as a bid-time gate, and it converges with Section 5.5's finding that the deficit is not a bid-time one. And on register: the MRC does not use the term fraud, on the grounds that fraud is a legal term implying intent. We use it, because it is what the literature and the vendors use, but where we are describing what a mechanism can establish rather than what an operator meant, invalid traffic is the more accurate word.

TAG's Certified Against Fraud programme is the enforcement counterpart. Version 10.1 of the guidelines (July 2025) mandates 100% GIVT and SIVT filtration against an MRC-recognized standard, requires data-centre IP filtering while stating that TAG's own data-centre IP list is not sufficient on its own, requires

ads.txt and app-ads.txt support, requires ads.cert for SSAI billing notifications, and requires SSAI headers. TAG reports an 86% reduction in invalid traffic in CTV for certified versus non-certified supply.

That figure raises the question this paper has to answer about its own instrument, so we answer it here rather than leaving it implicit. Certification programmes and vendor pre-bid products are not class-B mechanisms under Section 3.3, and the reason is rule 4: a check whose reference is a membership-gated registry or a commercial model, rather than a public machine-readable artifact, is not executable by an arbitrary receiver. DoubleVerify's Video Filtering, which the corpus records as the detector for SneakyTerra and is MRC-accredited, is a model-based rejection running inside one vendor's serving path. Its CTV Certification programme reports an eleven-fold fraud-rate differential between certified and uncertified supply. Both are real and both are excluded from the index by rule, not by oversight. What they establish is that the problem is tractable with enough privileged data, which is a different claim from the one this paper makes about what the protocol lets any participant check.

Read as a set, the protocol mechanisms answer questions about permission and identity of parties. None answers a question about an event. A supply path can be fully authorized, every node mutually disclosed, every connection cryptographically authenticated, and every impression in it fabricated.

One clarification is owed to practitioners before going further, because without it the rest of this paper can be misread. Vendors do catch these schemes, routinely, and this paper does not claim CTV fraud is undetectable. It is caught by statistical and behavioural means: autonomous system concentration, device-graph inconsistency, session-length distributions, traffic-shape anomalies, honeypots, and platform telemetry. Every scheme in our corpus was found that way. The claim here is narrower and different: these fields are not **rejectable at transaction time from the protocol payload alone**. That distinction matters because detection after the fact is a vendor service with uneven coverage, priced separately and applied unevenly across the market, whereas rejectability is a property of the protocol available to every participant. A mechanism that makes a claim falsifiable moves fraud from a detection problem, where the defender must be cleverer than the attacker, to a verification problem, where the attacker must produce something they cannot produce. The rest of the paper measures how little of the CTV transaction sits in the second category.

2.3 Related work

Supply-chain transparency has an established measurement literature. Bashir et al. gave the canonical longitudinal analysis of ads.txt adoption and its limits at IMC 2019, establishing the template this paper extends to a channel where the analogous file is not merely unadopted but unreachable. Vekaria et al. measured inventory pooling abuse that persists despite ads.txt and sellers.json at IEEE S&P 2024, the closest methodological relative of this work and, like the rest of the literature, web-only. Papadogiannakis et al. traced fraudulent revenue flows through transparency gaps at WWW 2025 and used these files as measurement instruments at WWW 2023. We found no dedicated academic study of app-ads.txt, of the supply chain object, of sellers.json accuracy over time, or of ads.cert. Our search covered arXiv, dblp, the ACM and IEEE digital libraries, USENIX, PETS and NDSS proceedings, by metadata rather than full text, through July 2026, so this is a negative result from a bounded search rather than a proof of absence.

The academic claim stands, but industry measurement of CTV app-ads.txt exists and it would be poor practice to leave it uncited while claiming a first. Pixalate published adoption figures for 2020 and 2021: app-ads.txt present for 80% of the top 500 Roku apps and 62% of the top 500 Fire TV apps. That agrees with our own per-domain completion rate in Section 7.3, from a different sampling frame, which makes it corroboration rather than collision. The difference is what is being counted: Pixalate measured whether top apps have a file, and we measure whether the specification's discovery walk can be executed across the whole population, where the answer is very different and the failure sits at the store-listing bootstrap. Pixalate has also measured CTV bundle identifier fragmentation, reporting that 40% of Roku apps were targeted under more than one bundle ID, and SSAI prevalence at roughly 70% of ad-supported CTV apps

and 85% on Apple TV. Those are the closest published measurements to this paper's subject and we build on them rather than around them.

Ad fraud measurement itself begins with Stone-Gross et al. at IMC 2011 and continues through Dave et al. on click-spam at SIGCOMM 2012, Springborn and Barford on impression fraud at USENIX Security 2013, and Pearce et al. on ZeroAccess at CCS 2014. Dave et al. framed the problem this paper generalizes fourteen years early: the advertiser cannot independently verify what it bought. Mobile in-app fraud produced a detection literature (DECAF at NSDI 2014, MAdFraud at MobiSys 2014, FraudDroid at ESEC/FSE 2018) whose common premise is precisely what CTV removes, since each instruments the client to catch what the client is really doing.

Smart TV measurement exists but is framed around privacy rather than verification. Mohajeri Moghaddam et al. crawled over two thousand Roku and Fire TV channels at CCS 2019 and demonstrated that the CTV advertising stack is measurable at the network layer. Varmarken et al. measured smart TV advertising and tracking at PoPETs 2020 and fingerprinted CTV apps at PoPETs 2022. Tagliaro et al. analyzed the HbbTV protocol at NDSS 2023, the only prior protocol-level analysis of a television advertising system, scoped to privacy and to European broadcast. Anselmi et al. audited automatic content recognition at IMC 2024.

Attestation of ad delivery has a short and stalled history. Li et al. built AdAttester on ARM TrustZone at MobiSys 2015, producing unforgeable clicks and verifiable display proofs on mobile. That line of work was never extended to CTV, which is where the client vantage it replaces is absent and where it would therefore matter most.

To our knowledge this is the first academic treatment of CTV ad fraud as a protocol and verification problem. We distinguish two nearest neighbours. Shamieh et al. published CTV advertising infrastructure work at IEEE CCECE 2025, on audio-fingerprint creative identification, so we do not claim priority on academic CTV advertising research generally. And an October 2025 industry study by CleanTap, not peer reviewed, passed counterfeit CTV devices built from single-board computers into live auctions across major platforms. We cite it as corroborating industry evidence, an independent demonstration consistent with what our index predicts analytically, rather than as a controlled test of the instrument.

This paper is the fourth in a series applying machine-checkability analysis to advertising protocols. Prior work classified the normative content of OpenRTB (Sekowski 2026b, doi:10.13140/RG.2.2.27937.57448), analyzed VAST validation at bid-time scale (Sekowski 2026a, doi:10.13140/RG.2.2.11404.27520), and measured client-side OpenRTB dialects in live header-bidding traffic (Sekowski 2026c, doi:10.13140/RG.2.2.26572.78720). Those papers asked whether a machine can check conformance to a specification. This one asks a different question: whether anyone can check whether a field is true.

3. Method: the Verifiability Index

The index is designed so that a reader holding only the specifications can recompute it. That constraint drives every choice below, and it rules out weighted composite scores, which cannot be audited by a second party without access to the weights.

3.1 Unit of analysis and field kinds

The unit is the leaf field, identified by its path in the request or response (for example `BidRequest.app.bundle`), evaluated per protocol version and per channel. The field universe is generated mechanically from the OpenRTB 2.6 object catalog rather than hand-picked, which matters because a hand-picked set invites the objection that the fields were chosen to produce the result. The generated inventory

covers 371 fields across 34 objects, each carrying its specification section and source line range. Native, Audio and Digital Out of Home objects are excluded by rule as out of channel scope.

Not every field can be true or false, and counting the ones that cannot would understate the deficit by padding the denominator with fields no adversary would ever want to falsify. Each field is therefore assigned a kind:

- **assertion**: a claim about the world (device, app, content, user, supply path). The index is computed over these, and they number 201 of the 371.
- **instruction**: an auction parameter such as `at`, `tmax`, `cur`, `bcat` or `bidfloor`. An instruction cannot be false, only dishonoured. Auction integrity is a real problem with a different shape, discussed in a sidebar rather than mixed into these numbers.
- **identifier**: an opaque value with no external referent, such as `imp.id` or `source.tid`. Internal-consistency checks apply; verifiability does not.
- **mechanics**: structural and serialization fields, container objects, version markers and extension envelopes.

Kind assignment uses explicit published tables rather than heuristics, so the assignment is itself reviewable.

3.2 The vantage rule

This is the methodological move on which the paper's comparison rests. Verifiability is not a property of a field. It is a property of a field evaluated from a particular party's position at a particular moment. We evaluate from the receiving party's vantage at or after transaction time.

The same field can therefore hold different classes in different channels without the specification changing a character. `device.ip` on web display is checkable because the buyer's measurement code, when it runs, observes the socket the request actually came from, and a declaration that contradicts that observation is rejectable. Behind server-side ad insertion the buyer commonly has no code on the client, and where the deployment fires tracking server-side there is no observation against which to test the declared value. Section 4 treats the beaconing mode as a variable rather than a constant, because it is one. Nothing about the field changed. The observation point disappeared.

Each field is coded per channel context: `web`, `ctv_csai` for client-side insertion, and `ctv_ssai` for server-side insertion. We code `web` and `ctv_ssai`. Client-side insertion (`ctv_csai`) is not coded in this release; every CTV index number is the server-side case. Section 11 states that the mix of insertion modes is the measurement this paper does not make.

Response-side fields invert the direction. There the buyer is the declarer and the exchange, publisher or stitcher is the evaluator, so response-side fields are coded from the receiving party's vantage on that side. This produces an asymmetry worth stating plainly, because it cuts against the paper's own thesis in one direction: server-side insertion weakens buyer-side verification of session claims while strengthening seller-side inspection of the creative, since the stitcher must fetch, parse and transcode the asset in order to splice it. We report request-side and response-side results separately for this reason, and mixing them would dilute both findings.

3.3 Classes, and the two criteria that make them defensible

Three classes, mutually exclusive, assigned per field per channel per scoring tier.

Class A, attested. A mechanism cryptographically binds the field's value to an accountable identity and the receiver can verify that binding. The binding must cover the value, not merely the transport session.

Class B, cross-referenceable. A public, machine-readable reference exists against which the receiver can mechanically check the declared value for existence, authorization or consistency.

Class C, self-declared. Neither exists. The receiver has the sender's word.

Two criteria do the real work, and both were forced by cases that would otherwise have been coded misleadingly.

The falsification criterion. A check qualifies for class B only if it can reject at least one class of false declaration from a rational adversary. An observation mismatch qualifies (declared user agent against headers the buyer's code actually saw). An authorization or consistency mismatch qualifies (a seller absent from the app-ads.txt file for the declared bundle, a supply chain node absent from the named ad system's sellers.json, a store URL that does not resolve to the declared bundle). What does not qualify is a plausibility check that any adversary satisfies by choosing convenient values: that a declared IP falls in residential address space, or that a declared bundle identifier exists in some store. Without this criterion the index would score as verified every field an adversary trivially satisfies, and the channel comparison would collapse, because plausibility checks are exactly what remains available when observation is gone.

The independence clause. A signature proves authorship, not truth. If the attesting party is the party whose interest the value serves, the mechanism yields non-repudiation of a self-declaration: the lie becomes attributable rather than detectable. Class A therefore splits. **A-independent** requires that the attester be independent of the party the value benefits, or that the attestation derive from an observation the attester did not control; only this counts as verification in our headline numbers. **A-self** is cryptographically signed by the interested party, is reported separately, and is treated as an accountability mechanism valuable only after detection.

This clause is not hypothetical, and Section 9 shows why it is necessary. The agentic protocol AdCP signs seller-reported delivery webhooks with a publisher-anchored key, satisfying the literal binding requirement while leaving the delivered count exactly as trustworthy as the seller's word. The same protocol requires receiver-verified, issuer-anchored attestation for whether a viewer is a unique human over 18. Without the independence clause an index would score those two mechanisms identically, which would be absurd.

The clause is party-relative rather than vendor-relative. Platform device attestation is A-independent when a platform certifies traffic that third parties claim originated from its devices, which is how the Roku Advertising Watermark caught the CycloneBot scheme. The same platform attesting inventory it is itself selling is A-self. The class follows who benefits from the value in that transaction.

A third rule excludes bilateral-contract references from class B by explicit choice. A check whose reference is a private agreement between the two transacting parties can mechanically reject a mismatch, and so satisfies the falsification criterion, but no third party can run it and its content is negotiated rather than published. We record these in a `checkable_today` field and discuss them as their own category. We state the exclusion as a decision so it is not read as an oversight.

Finally: no partial credit and no weights. A field is A, B or C. Borderline cases are resolved by the rules above and flagged in the data, and the results report a variant with all borderline codings removed.

3.4 Three scoring tiers

A verification mechanism can fail at three separate points, and collapsing them hides where the failure actually is. Every field is scored three ways.

- **Index_spec**: the mechanism exists in a published standard, adoption ignored. ads.cert 2.0 counts here.

- **Index_deployed**: the mechanism is operated as live public infrastructure as of the coding date. sellers.json and app-ads.txt count; ads.cert 2.0 does not.
- **Index_reachable**: the mechanism can actually be executed against the inventory in question. A check that is specified and operated is still worthless if the data needed to start it is missing for the inventory being evaluated.

The third tier is unusual and it is where our crawl enters the index rather than sitting beside it. The app-ads.txt walk requires the store listing to publish a developer URL. Section 7.3 measures that 11.0% of the Roku channel population does. The fields whose only class-B justification is that walk are therefore class B under the first two tiers and class C under the third for the majority of the population. We report both, never silently, and we report the crawl separately for the channel population and for major services because they answer different questions. We do not claim to know which weighting matches the inventory a buyer actually receives; Section 11 records that as a limitation.

The spread across the three tiers distinguishes three situations the industry conversation routinely conflates: a standard that does not exist, a standard nobody adopted, and a standard that cannot be run on the inventory it was written for.

3.5 Pricing relevance, defined without reference to fraud

Every field on the CTV path is coded, which avoids cherry-picking. A field is additionally tagged pricing-relevant if it influences valuation, targeting, brand safety or measurement, and that tag is operationalized independently of the fraud corpus. A field qualifies if it is exposed as a targeting or reporting dimension in public demand-side platform documentation, or is an input to deal and private marketplace matching, or is an input to an industry measurement standard.

The exclusion here is deliberate and load-bearing. We never treat “a fraudster falsified it” as evidence that a field is pricing-relevant. Doing so would make the paper’s central finding circular: fraud clusters on unverifiable fields, and fields count as important because fraud touched them. Revealed preference is reported once, as a robustness check, and never as a definition.

3.6 Corpus and mitigation coding

A scheme enters the corpus if it is **named**, publicly disclosed by a named discoverer, with a mechanism description sufficient to identify at least one falsified field. The name requirement does real work: it excludes vendor aggregate categories, which would otherwise let a single report describing 1,300 fraudulent apps enter as one row and distort every count. Categories and unnamed disclosures are recorded with their own status values and excluded from the denominator, and every candidate we rejected carries a published reason, so the corpus boundary is contestable rather than implicit.

Field mappings are graded **documented** when the source states the field, **inferred** when it follows mechanically from the described mechanism, or **family** when the source describes it for the scheme’s family rather than for the scheme. No headline number rests on inferred mappings alone, and Section 6.1 reports the documented-only bound beside every count. Scale and financial figures are quoted verbatim with attribution and never restated as our own estimates. Dates for identification and disclosure are recorded separately, because in this corpus they differ by up to 29 months and secondary coverage routinely reports one as the other.

The mitigation matrix asks, per scheme and per standard, whether the scheme would be caught, partially constrained, or unaffected assuming 100% correct adoption and enforcement. Every cell carries a one-sentence justification, and every row carries a confidence grade reflecting how much mechanism detail the discoverer published, so the headline can be reported with and without the weakly sourced rows.

For the authorization standards, each cell carries **two** verdicts rather than one. Whether an authorization file rejects a scheme depends on whether we score the app identity the scheme declared or the fabrication mechanism, which can be pointed at a different app, and a matrix that leaves that choice unstated will apply it inconsistently across rows. Section 7 defines the two readings and reports both. Where a source publishes too little to score a cell at all, the verdict is `unresolved` and the affected count is reported as a bound.

3.7 Threats to validity

Single-coder judgment. The codings are one author's reading of specification text. To quantify that risk we drew a stratified sample of 47 fields across both coding passes and all classes, and had an independent coder classify them from the method document alone, blind to the original codings and without access to the coded data. Agreement statistics are reported in Section 5.4.

Coding at scale. Of 206 coded assertion fields, 43 were coded individually and 163 by explicit published pattern rules over field paths, with every rule and its rationale committed. Results are reported with and without the rule-coded fields so a reader can see whether the batch pass moves any conclusion. It does not, and the hand-coded subset shows a larger channel gap than the full set.

Mechanism-existence judgments. Deciding whether a mechanism qualifies as public infrastructure involves judgment at the margins, notably for membership-gated registries such as the TAG payment identifier registry and Ad-ID. These are flagged borderline and the affected fields are listed.

Counterfactuals are reasoned, not experimental. Section 7 asks what a standard would have caught. We cannot run that experiment. We mitigate by publishing per-cell justifications so disagreement can be localized to specific cells rather than aimed at the conclusion.

---

4. The CTV identity chain

Figure 1 sets the two architectures side by side. The nodes are nearly the same. What differs is where an observation can be made.

The same fields, evaluated from two different vantages

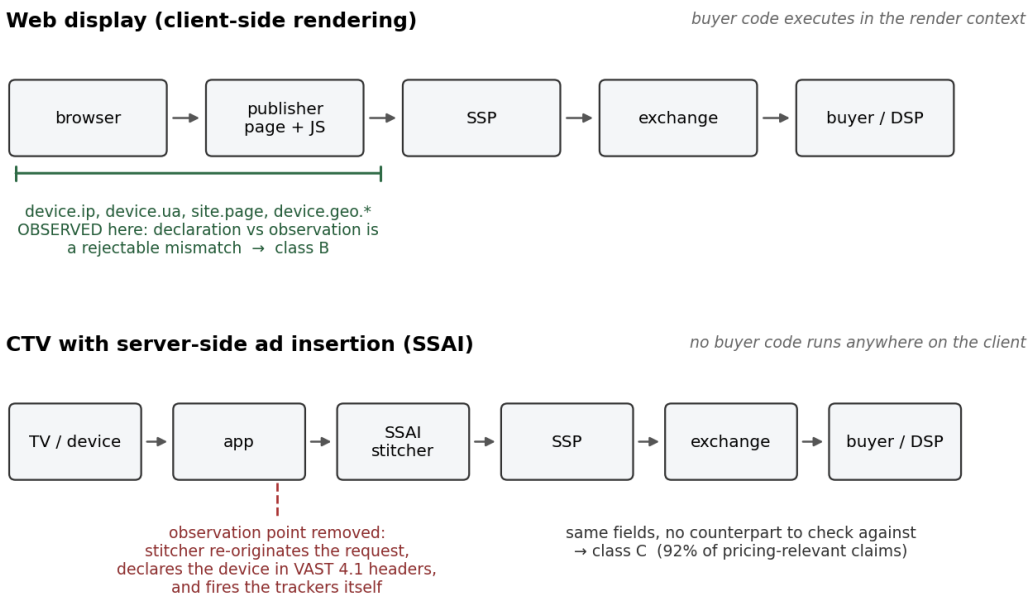

Figure 1. Where an observation can be made. The two delivery paths carry almost the same nodes. On web display the creative renders in a context where the buyer's measurement code executes, so a subset of declarations can be contradicted by something observed. On the CTV server-side path the request that reaches the exchange did not originate at the device, and where the deployment also fires tracking server-side there is no point at which the buyer observes the client at all.

On the web display path, the browser fetches the page, the publisher's code runs an auction, and the winning creative renders in a context where the buyer's own measurement code executes. That code is the verification substrate. It reports the URL it actually rendered on, the viewport geometry it actually occupied, and, by the simple act of making an HTTP request, the socket the client actually used. The chain still carries declarations, and those declarations can still be wrong, but a subset of them can be contradicted by something observed.

On the CTV server-side path an additional party sits between the device and everything else. The stitching server assembles a continuous video stream containing both content and advertising, so the television receives one stream and makes no separate ad request.

The chain is also longer than the two-hop version usually drawn, and the extra hops strengthen the argument while breaking the simpler telling of it. The stitcher does not typically call an exchange directly. It calls the publisher's ad server, which runs its own decisioning and calls supply-side platforms, which originate the OpenRTB request that a buyer eventually sees. So the distance between the declared client and the party observing anything is two or three hops, not one, and the nodes that appear in a SupplyChain object are ad servers and SSPs rather than the stitcher whose account of the device everything downstream inherits. Anyone reading the schain to find out who asserted `device.ip` will not find them there. IAB Tech Lab's SupplyChain version 1.1, announced in June 2026 with comment open to August, adds technical-custody reporting precisely so that SSAI platforms, ad servers, SDKs and wrappers become visible in the chain, which is an acknowledgement of this gap by the body that defined the object.

The consequences compound:

1. **The request to the exchange did not originate at the device.** Its socket belongs to a server. `device.ip` becomes an upstream party's account of which client is watching, conveyed as a field value rather than observed as a connection property. The convention of relaying it through headers such as `X-`

`Device-IP` and `X-Device-User-Agent` is precisely what the ICEBUCKET operation falsified in 2020, and note what that means: the scheme did not have to defeat a check on `device.ip`. It supplied the value that the downstream chain then copied into `device.ip` in good faith.

2. **The user agent is re-originated.** An intermediary constructs it. There is no header the buyer can compare it against.
3. **Where the deployment fires tracking server-side, the impression event carries no evidence of a render.** This is the conditional claim and the one most often overstated, including in an earlier draft of this paper. Server- side insertion is a family of deployments, not one: AWS Elemental MediaTailor ships server, client and hybrid reporting modes, Google's dynamic ad insertion runs through a client-side SDK with a documented Open Measurement integration, Roku's advertising framework fires impression and quartile events from the client, and IAB Tech Lab's SSAI VAST macro guidance defines a `[SERVERSIDE]` macro so that a receiver can distinguish the cases. The Media Rating Council, in the normative document on this subject, states that SSAI inventory on major platforms tends to be enabled for client-initiated measurement. Where beaconing is client-initiated the impression event does carry device-side evidence and part of the buyer's vantage is restored. Pixalate's transparent versus non-transparent SSAI distinction is the useful frame, and we score the non-transparent case throughout. What fraction of inventory is in each mode is, as far as we can establish, unmeasured in public, and Section 11 records that as the most consequential gap in this paper's evidence.
4. **Independent measurement often has nowhere to execute.** VAST carries an `AdVerifications` element for exactly this purpose, and on the web the verification resource it names runs in the render context. Behind server-side insertion there is frequently no such context. On the platform we crawled the situation is stronger than "unexecuted": Roku's advertising framework supports VAST 2.0, VAST 3.0, VMAP and FreeWheel SmartXML, with no Open Measurement SDK, no OMID and no `AdVerifications` element in the grammar at all, and third-party measurement handled through Nielsen and Comscore integrations instead. IAB Tech Lab published a draft VAST CTV Addendum in July 2024 that backports Universal Ad ID and Open Measurement support into VAST 2 and 3 for exactly this reason. Our VAST codings are therefore platform-conditional as well as channel-conditional, and Section 5.2 says which.

The declarer of record therefore changes at the SSAI hop, and every downstream party inherits assertions rather than observations. This is why we evaluate the index per channel context rather than per protocol version: the specification is identical on both paths, and the verifiability is not.

It is worth recording that the Media Rating Council reached the operative conclusion in August 2021, four and a half years before this paper. Its guidance on server-side ad insertion and over-the-top delivery requires that accredited counting be client-initiated, requires server-fired-only tracking to be segregated and disclaimed as non-compliant, and requires SSAI providers to be verifiable or their traffic to be treated as unknown. It also proposes an SSAI provider certification scheme whose stated rationale is that invalid traffic perpetrators may disguise themselves as SSAI providers. That is ICEBUCKET described in the abstract, by a standards body, in the year after ICEBUCKET was disclosed. What this paper adds is not the observation. It is a field-level measurement of how much of the transaction the observation covers, and a count of what the intervening standards did about it.

5. Index results

Table 1 gives the class distribution for pricing-relevant request-side assertion fields, which is the buyer's verification position stated as a number. Figure 2 shows the same result graphically.

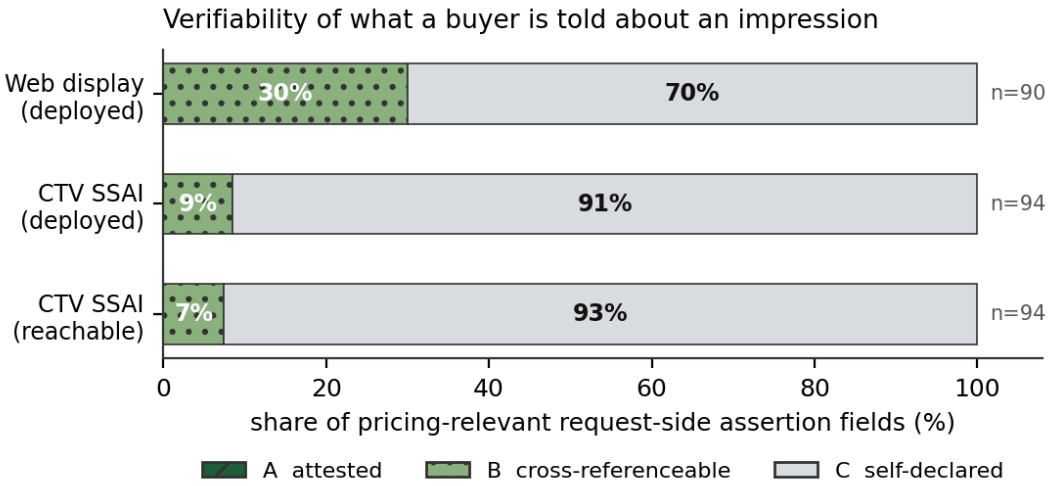

Figure 2. The Verifiability Index, pricing-relevant request-side assertion fields. Class A is attested, class B cross-referenceable, class C self-declared. The three bars are web deployed, CTV SSAI deployed, and CTV SSAI reachable. No scored field in either channel is class A. Table 1 and Section 5.5 report the strict (registry-only) tier, which is not drawn here.

Table 1. Pricing-relevant request-side assertion fields by class.

context and tier	n	A	B	C	self-declared
Web display, deployed	90	0	27	63	70.0%
CTV SSAI, deployed	94	0	8	86	91.5%
CTV SSAI, reachable	94	0	7	87	92.5%
Web display, strict (registry checks only)	90	0	5	85	94.4%
CTV SSAI, strict	94	0	8	86	91.5%

The denominators differ (90 against 94) because some fields exist on only one channel's path. Section 5.5 reports the variant restricted to fields applicable on both, which is the like-for-like comparison.

Class A is empty on the server-side path, and it stopped being empty elsewhere in October 2025. This requires care, because the honest version is both more interesting and more favourable to the industry than the flat claim we would otherwise make.

In October 2025 IAB Tech Lab released Device Attestation support in version 1.6 of the Open Measurement SDK, adapting the IETF Privacy Pass protocol so that a buyer can independently verify that inventory came from a genuine device, using privacy-preserving attestations issued by the device manufacturer. Early adopters include DoubleVerify, HUMAN, Google, IAS, Amazon Ads and AdsWizz. Under this paper's own independence clause that is textbook A-independent attestation: the attester is the manufacturer, which does not benefit from the value of the claim, and the receiver verifies it. It is precisely the mechanism our Section 8 ranking places near the top, and it exists.

Three properties determine where it lands in the index, and all three follow from this paper's method rather than qualifying it.

First, **it rides on the Open Measurement SDK**, so it requires a client execution context. That is exactly what server-side ad insertion removes. The vantage rule therefore places it at class A on paths where OM SDK runs and class C on the server-side path, for the same architectural reason that puts `AdVerifications` at B on web and C behind a stitcher. The industry's answer to device spoofing was built on the substrate that CTV's dominant delivery architecture eliminates.

Second, **coverage is partial**: Apple devices and Fire TV at release, with Roku, Samsung, LG and Vizio absent. Roku is the largest US platform and the one we measure in Section 7.3, and its advertising framework does not parse VAST 4.x at all, so the surrounding verification machinery is not available there either. Under Index_deployed this is a mechanism operating on part of the ecosystem; under Index_reachable it is unavailable for the inventory this paper is about.

Third, **it is post-bid and sampled by design, not per-impression at transaction time**, and the guidance says so in terms that could have been written for this paper: "To preserve user experience, device attestation should not be done pre-bid", and "Device Attestation cannot be a gating criterion." Verification operates at the seller level, joined to supply-side dimensions through macros or out-of-band mechanisms rather than carried in the bid request. It establishes whether a seller's traffic is genuine over time; it is not a check a buyer runs on the impression in front of it.

That last property is worth stating as a general fact about the protocol rather than a property of one mechanism. The strings `attest` and `attestation` do not appear anywhere in OpenRTB 2.6, OpenRTB 3.0 or AdCOM 1.0. There is no standardized field in which an attestation could travel with a bid request even if every device produced one. The object on which the auction is decided still carries a plaintext `device` object, and IAB Tech Lab's own problem statement for the attestation work concedes the consequence: because device information "is represented in a text string, it can be easily manipulated by intermediaries in the OpenRTB protocol." The gap this paper measures is, in that sentence, the standards body's own finding.

So the corrected finding is sharper than the one we set out to report. The mechanism this analysis identifies as highest-leverage was standardized nine months before this paper, by the industry's own standards body, with the major verification vendors as adopters, and it was built on the one foundation that server-side ad insertion takes away. On the CTV server-side path, class A remains empty. That is not because nobody built the tool. It is because the architecture the inventory is delivered through cannot hold it.

Two exclusions remain definitional and we state them: ads.cert 2.0 is not class A here because Section 7.5 treats it separately, and OMID measurement without attestation is class B rather than A because it observes rather than attests.

The deployed gap is large. A buyer on the CTV path has a check available for eight of 94 pricing-relevant claims. On web the same buyer has one for 27 of 90.

Reachability widens it slightly, to 92.5%, once the app-ads.txt result of Section 7.3 is applied to the two fields whose only class-B justification is that walk.

5.5 What the gap actually is: two kinds of check

The last two rows of Table 1 carry the paper's most important result, and it is not the one we expected to report.

Class B admits two structurally different checks. **Registry checks** run on the payload plus a public file before the auction clears: the app-ads.txt walk, a sellers.json lookup, store-listing resolution. **Observation checks** require the receiver to observe the render: buyer JavaScript, OMID, the socket the request arrived on, HTTP headers, client hints. Both can reject a false declaration, so both satisfy our falsification criterion, but they differ in when the answer arrives and therefore in what the receiver can do with it.

Web's checkability is overwhelmingly the second kind. Of its 27 checkable pricing-relevant request-side fields, 22 (81%) are observation-dependent. Scoring only registry checks moves web from 70.0% to **94.4%** self-declared. CTV does not move at all, from 91.5% to **91.5%**, because every check CTV retains is already a registry check. Across all request-side assertions the same split is 46 of 55 (84%).

So the honest statement of the finding is not that CTV cannot verify claims at bid time. **Neither channel can verify device IP, user agent, or device type from the payload and public files before the auction**

clears. On the unpaired strict tier the two channels are within three percentage points of each other, and CTV is nominally the better of the two (91.5% against 94.4%), because the checks it has are runnable pre-bid while web's are not. That ranking is a composition artifact. Restricted to fields present on both paths (Table 2), strict-tier CTV is 94.3% self-declared against web 94.3%: the comparison does not invert. The extra CTV registry checks that produce the unpaired inversion are app-identity fields (`app.bundle`, `app.storeurl`, `app.name`) that web does not carry. They do not make device claims checkable.

What differs is whether ground truth ever arrives. On web it arrives at render, and that closes a correction loop: discrepancy claims, make-goods, seller and domain blocklisting, and pre-bid filtering of subsequent traffic from sellers whose declarations were contradicted. The individual impression is not rejected, but the lie has a cost and the cost compounds. Behind server-side ad insertion that loop never closes, because the observation never happens, not before the auction and not after it. The remaining signals are statistical, which is a different and weaker instrument, and one available only to parties who buy it.

This reframing strengthens rather than weakens the paper's argument, and it explains something the simpler version could not: why CTV fraud persists across eight years and seventeen documented operations while the equivalent web schemes were driven to elaborate cloaking. It also relocates the recommendation. The fix is not a better pre-bid check, since web does not have one either. It is a mechanism that makes ground truth arrive at all, which is exactly what Section 8 ranks first.

Table 2 reports the robustness variants, now including the cuts that move the result most.

Table 2. Robustness variants. Self-declared share, deployed tier unless noted.

variant	web n	web	CTV n	CTV	gap
all coded assertions	188	60.6%	188	82.5%	21.8 pp
pricing-relevant, both sides	96	66.7%	100	87.0%	20.3 pp
pricing, request-side (headline)	90	70.0%	94	91.5%	**21.5 pp**
pricing, request-side, applicable both channels	88	71.6%	88	94.3%	22.7 pp
pricing, request-side, borderline removed	79	79.8%	82	93.9%	**14.1 pp**
pricing, request-side, hand-coded only	24	33.3%	25	72.0%	38.7 pp
pricing, request-side, strict tier	90	94.4%	94	91.5%	**−2.9 pp**
pricing, request-side, applicable both, strict	88	94.3%	88	94.3%	0.0 pp
by distinct mechanism rather than field	80	78.8%	94	91.5%	12.7 pp

Three of these deserve comment because they cut against us.

Removing borderline codings cuts the gap by a third, from 21.5 to 14.1 percentage points, and the reason is specific: 11 of the 12 borderline codings in the headline subset are web class-B fields, mostly geographic consistency checks. None is web class-C. The borderline flag therefore marks precisely the codings that flatter the web baseline, so this variant is the most adversarial cut available and we report it in Table 2 rather than burying it. Roughly seven percentage points of the deployed gap rests on judgments we ourselves marked uncertain.

Counting mechanisms rather than fields also cuts it, to 12.7 points. Eight `device.geo.*` fields rest on one check (comparison against the observed IP's geolocation) and five `imp.video.*` fields rest on one OMID viewport measurement, so field counting lets a single web mechanism buy eight class-B fields. The direction of the finding is unchanged, but the magnitude is partly an artifact of how finely OpenRTB subdivides one observable, and a reader should treat the field-count gap as an upper bound and the mechanism-count gap as a lower one.

The applicable-both variant is a weaker control on the deployed tier than it appears. Restricting to fields present on both paths removes 3 of CTV's 8 checkable fields, because `app.bundle`, `app.storeurl` and `app.name` have no web counterpart by construction, and only 2 of web's 27. It therefore bounds the composition confound in one direction only, by discarding CTV's entire app-authorization surface, and its wider deployed gap of 22.7 points should not be read as a stricter test of the 21.5-point headline. On the strict tier the same cut is the right control for the inversion claim, because that claim is about device and session fields both channels carry. It is the row that does not invert (a tie at 94.3%).

5.1 Where the surviving CTV checks live

The checkable fields on the CTV path are not distributed across the transaction. They cluster, and where they cluster is itself a finding. Counting all request-side assertion fields, the 14 with a class-B check on the CTV path break down by mechanism as: sellers.json 7 (the four `schain.nodes[]` fields plus `app.publisher.id`, `app.publisher.name` and `app.publisher.domain`), store-listing resolution 4 (`app.storeurl`, `app.name`, `app.domain`, `app.paid`), app-ads.txt 2 (`app.bundle`, `app.inventorypartnerdomain`) and identity-provider token validation 1 (`user.eids[].uids[].id`).

Read that list against what it omits. Every one of those checks concerns who is selling or which app is named. **None concerns the device, the viewer, the content, or whether a render occurred.** Not one field describing the thing being bought, a person watching a screen, has a check on the CTV path.

The web breakdown for the same fields is the mirror image: 46 observation-dependent checks, 7 sellers.json, 1 identity-provider token, 1 unclassified (`site.inventorypartnerdomain`, an ads.txt lookup the tagger does not name). The authorization plumbing is nearly identical across the two channels, which stands to reason since it is the same plumbing. The entire difference is the 46 observation checks, and Section 5.5 is about what those are worth and when.

This produces the asymmetry we flagged in Section 3.2 and it is worth stating against our own interest: server-side insertion strengthens creative-side verification. The stitcher must fetch, parse and transcode the media file in order to splice it, which puts it in the strongest inspection position of any party in the chain. Table 3 shows the consequence.

Table 3. Request-side versus response-side, self-declared share.

side	n (web / CTV)	web	CTV
request-side, all assertions	165 / 165	66.7%	91.5%
response-side, all assertions	23 / 23	17.4%	17.4%

Response-side claims are identically verifiable in both channels on this cut. The figure is sensitive to confidence flags: dropping the 15 borderline rows leaves eight settled assertions at 50% self-declared on both channels (`response_settled` in `compute_index.py`). The deficit is not "CTV is unverifiable." It is precise: claims about **where and to whom** an advertisement was shown are unverifiable, and claims about **what** was shown are largely verifiable. Every scheme in Section 6 exploits the first category and none of them needs to touch the second.

5.2 The VAST layer

Coding the creative layer separately (12 assertion surfaces, from VAST 4.3) reproduces the same split one level down.

Asset-side assertions are class B on both channels because the receiver holds or must resolve the artifact: `MediaFile` attributes can be checked by fetching and probing the asset, `Duration` against the asset's real duration, the `Wrapper` redirect chain by resolving it, and `Advertiser` against the creative's actual destination.

Every event-side assertion is class C on both channels. `Impression` and each quartile `Tracking` event is an unauthenticated HTTP request. It carries no proof of render, no proof of device, no proof of session. Anyone who can reach the URL can assert the impression, and the only available checks are rate, ordering and duplicate heuristics that a rational adversary satisfies by firing the beacons in order on a timer. This is the single sentence of the paper that most concisely explains CTV fraud: **the event that money is paid on is an unauthenticated HTTP GET.**

One qualification, since a practitioner will supply it if we do not. Billing does not rest on that GET alone in a mature buying relationship: a demand-side platform reconciles its logged impressions against a count from an MRC-accredited measurement vendor, and discrepancies are negotiated. That is real, and it is also the correction loop of Section 5.5 rather than a check on the event: it operates in aggregate, after settlement, between parties who have a contract, and it is unavailable to a receiver evaluating a single impression from the payload.

Between these sits `AdVerifications`, which is the mechanism whose loss defines the channel gap. It is class B on web, where the named verification resource runs in the render context and can contradict a claimed render, and class C behind server-side insertion, where there is no context for it to run in. The specification carries the hook. The architecture removes the socket it plugs into.

That coding is platform-conditional as well as channel-conditional, and on the platform we crawled the situation is more absolute than the coding suggests. Roku's advertising framework parses VAST 2.0, VAST 3.0, VMAP and FreeWheel SmartXML. It has no Open Measurement SDK, no OMID and no `AdVerifications` element, and routes third-party measurement through Nielsen and Comscore integrations instead. So on the largest US CTV platform by installed base, the hook whose loss we are describing is not present in the grammar to begin with. IAB Tech Lab's draft VAST CTV Addendum (July 2024) backports Universal Ad ID and Open Measurement into VAST 2 and 3 for exactly this reason, which is an acknowledgement that the CTV population is not running the version the specification lineage assumes. We record `AdVerifications` as class C on the CTV server-side path for the reason above, and note that on Roku specifically the reason is different and stronger.

And the framework has a blind spot here that the corpus exposes. Every coding in this subsection assumes that if a verification hook executes, the check it performs is sound. ViperBot violates that assumption directly: its mechanism is to strip the measurement tags out of the delivered advertisement and reinsert them into cheap slots on real devices, so the discrepancy signal that should expose the undelivered impression never arrives, and DoubleVerify reports that all measurement vendors' tags were affected. The declared values are all true. The verification layer is the thing under attack. Nothing in the Verifiability

Index represents that threat, because the index scores whether a value can be contradicted and says nothing about whether the contradicting mechanism can be removed. We flag it as outside the instrument rather than as something the instrument scores badly, and it is the one row in Section 7 that no mechanism catches outright.

5.3 Specification, deployment, reachability

Within OpenRTB the specified and deployed tiers differ for exactly one field, `source.pchain`, whose payment-identifier registry is membership-gated rather than open infrastructure. This is a genuine but narrow result. For the 163 batch-coded rows, specified and deployed classes are identical by construction (`batch_code.py`); the one-field finding is confined to the hand-coded subset. It is worth stating why the gap is narrow: the deployed standards mostly do not address these fields at all, rather than addressing them aspirationally. There is little standards theater in OpenRTB itself because there are few standards pointed at these fields to begin with.

The interesting gap is at the third tier, and it required going out and measuring the ecosystem rather than reading specifications. Section 7.3 reports it.

5.4 Inter-coder reliability

A stratified sample of 47 fields, drawn across both coding passes and all classes, was recoded from the method document alone, with no access to the original codings. The second coder was a separate instance of a large language model given METHOD.md and no other context. That is a test of whether the method is self-sufficient as written, not expert corroboration, and it is a weaker instrument than an independent domain expert.

Table 4. Inter-coder agreement, n=47. Second coder is an LLM instance working from METHOD.md only.

dimension	raw agreement	Cohen's kappa
CTV SSAI class	95.7%	0.895
CTV SSAI class, both-applicable only	95.5%	0.860
Web class	68.1%	0.534
Web class, both-applicable only	82.8%	0.659
Pricing-relevance tag	70.2%	0.408

The CTV class assignment, which is what the headline rests on, replicates strongly. The two coders independently produced an identical self-declared share of 79.5% on the sample, and only **two** of 47 fields drew a substantive class disagreement on the CTV path: `source.schain.complete` (we code C, the second coder B) and `user.eids[].uids[].id` (we code B, the second coder C). The second of those was flagged borderline in our data before the recode, which is the behaviour one wants from a confidence flag.

The two weaker rows are informative rather than reassuring, and we report them without smoothing.

Web class disagreement is mostly notational. Ten of the disagreements are one coder marking a field not applicable to web where the other assigned it a class. These are `app.*` paths whose objects (`Content`, `Publisher`) also appear under `Site` in the specification: we coded the object as present on both paths, the second coder read the literal `app.` prefix as app-only. Restricting to fields both coders judged applicable lifts raw agreement to 82.8% and kappa to 0.659. This is a path-notation ambiguity in our inventory rather than a disagreement about verifiability, and it affects the web baseline only, never the CTV numbers.

Pricing relevance is the softest judgment in the method, at kappa 0.408 with a 95% bootstrap interval of [0.16, 0.65] (`reliability.py`, seed 20260725), and it is also the dimension that decides which fields enter the headline denominator. That combination is the weakest point in our instrument and we state it without softening.

The disagreement is directional: the second coder was consistently more generous, tagging keyword arrays, `device.model`, `device.osv`, `device.connectiontype`, `bid.dur` and `app.id` as pricing-relevant where we did not. Applied to the sample, their broader reading enlarges the subset from 22 to 29 applicable fields and moves the CTV self-declared share from 81.8% to 75.9%.

That direction is against us on the level, and an earlier version of this paper described it backwards, so we are explicit: **our narrower tagging yields a self-declared share about six percentage points higher than the broader reading would.** The narrow tag is anti-conservative on the level. What it does not do is disturb the channel comparison, because the identical tag is applied to both channels, and the comparison is what the paper's argument rests on. A reader who prefers the broader reading should reduce our level estimates by roughly six points and leave the gap alone.

We draw two conclusions. The class instrument is reliable; the pricing-relevance boundary should be read as an interval, not a point. Section 5 therefore reports the index over all coded assertions alongside the pricing-relevant subset.

Three further caveats on this analysis, since a reliability section that flatters itself is worse than none. The kappa intervals are wide at n=47: CTV class is 0.895 with a 95% bootstrap interval of [0.71, 1.00], web class 0.534 [0.35, 0.71]. The sample is 40% hand-coded against 20.9% in the population, so the unweighted values are not population estimates. And the identical 79.5% share the two coders produced on the sample is arithmetic cancellation rather than convergence: two disagreements offset, one in each direction.

6. The fraud corpus

We assembled the CTV fraud schemes that meet one inclusion rule: publicly disclosed by a named discoverer, with enough published mechanism detail to identify at least one falsified bid-request or VAST field. "Named" is load-bearing, and it is what keeps this a corpus rather than a summary of vendor marketing. The result is 17 schemes spanning 2018 to 2026, and Figure 4 places them against the dates the relevant standards became available. Discoverers are DoubleVerify's Fraud Lab (10 rows), HUMAN's Satori team, formerly White Ops (4), Pixalate (2) and Method Media Intelligence (1), with Oracle Moat co-discovering one.

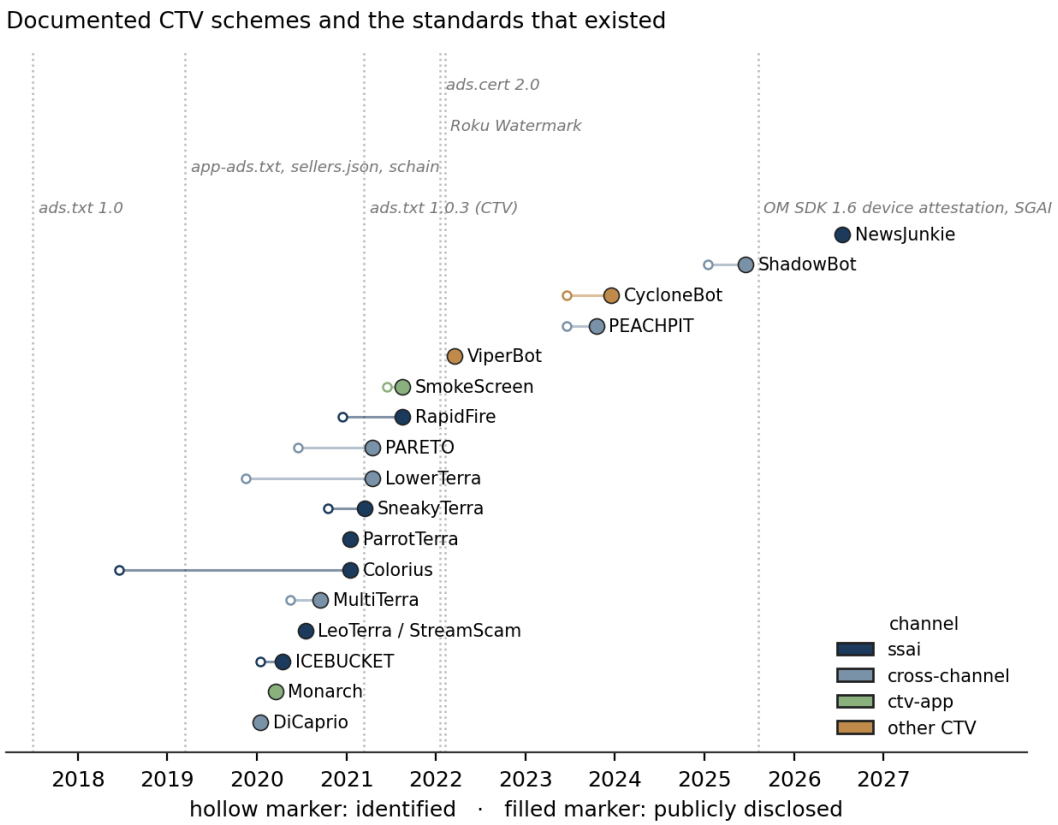

Figure 4. The corpus against the standards that existed. Hollow markers are identification, filled markers public disclosure, and the connecting line is the lag between them, which runs to 29 months. Every standard in Section 2.2 was specified and available before the most recent schemes in the corpus.

Around those 17 sit seven cases that are recorded and deliberately not counted: one phase variant, one umbrella operation, three labelled contrast cases, one adjacent hardware operation, one unnamed disclosure and one vendor-described category. A further ten candidates were considered and rejected. The full set, with a rejection reason for each, is in `data/fraud-corpus.json`, published so that a reader who thinks we drew a line wrongly can see where we drew it. Four decisions there are worth stating in the text because a reader who disagrees with them should be able to aim at the decision rather than the count.

BeatSting and VASTFLUX are not CTV schemes, and an earlier version of this corpus counted both. DoubleVerify's own disclosure headline calls BeatSting the first large-scale ad impression fraud scheme in audio, and its 60-plus participating apps are mobile audio and radio apps. No primary source for VASTFLUX mentions CTV, Roku, Fire TV or set-top boxes: the entry vector was a mobile in-app banner and the population was primarily iOS. Its only CTV touchpoint is that command-server configurations instructed VAST URLs to declare a traffic source of "OTT Video" to fetch higher CPMs, which is mobile inventory mislabelled as CTV rather than CTV fraud. Both are retained as contrast cases, where they are genuinely informative, and neither is in any count. VASTFLUX's exclusion matters arithmetically as well as definitionally: its claimed 12 billion daily requests exceed every genuine CTV scheme in this corpus combined, so it distorted any aggregate it entered.

These are not independent operations, and we no longer describe them that way. DoubleVerify's OctoBot report states that seven CTV schemes it had tracked since November 2019 were executed by a single group, established through shared applications, SDKs, command servers, platform accounts and open-source intelligence work. Three of our rows are documented OctoBot members. Separately, DoubleVerify's CycloneBot report notes that the apps CycloneBot spoofed included apps associated with SmokeScreen,

which is a second lineage link. The defensible phrase is “named schemes with published mechanism detail,” and we use it.

LeoTerra and StreamScam are one operation on DoubleVerify’s word alone. DoubleVerify states that LeoTerra “later resurged in December 2020, when other companies identified this same scheme using the name StreamScam.” Oracle Moat, which disclosed StreamScam independently, never confirmed the shared identity, and Oracle Moat has since been shut down, so the assertion will remain unrebutted rather than confirmed. We treat them as one row and credit both discoverers, so that collapsing the names does not erase Oracle Moat from the record.

Two dates in circulation are wrong, and PDF creation dates settle both. MultiTerra is September 2020, not February 2022: the flash report’s creation date is 15 September 2020, its copyright is 2020, DoubleVerify’s own OctoBot report puts identification at May 2020, and the February 2022 item that produced the error is a separate aggregate economic-impact release. CycloneBot is December 2023, not February 2024: creation date 14 December 2023, copyright 2023, and the mitigation timeline inside the report runs from 4 to 25 June. Both corrections move a scheme across a year boundary in Figure 4, and one of them moves MultiTerra from after the standards were mature to during their rollout.

Throughout, we separate the date a scheme was identified from the date it was disclosed, because in this corpus the gap between them runs from zero to 29 months. Colorius is the extreme case: identified in 2018, named for the first time in January 2021, retrospectively, inside a sales one-sheet.

6.1 The crosscheck

Mapping each scheme’s falsified fields onto the index gives the paper’s central empirical result. All counts in this subsection are produced by `compute_corpus.py` from the corpus and the codings, and none is hand-counted.

Sixteen of the 17 schemes falsified at least one field we code class C on the CTV path. Fourteen distinct class-C fields carried fabrication across the corpus: `device.ip`, `device.devicetype`, `device.make`, `device.ifa`, `device.ua`, `device.model`, `device.os`, `device.osv`, `device.geo.country`, `device.w`, `device.h`, `imp.video.w`, `imp.video.h` and `app.content.title`.

Three precisions are owed, because the tempting version of that sentence overstates it.

First, the set is concentrated rather than uniform. `device.ip` and `device.devicetype` appear in 10 rows each, `device.make` in 8, `device.ifa` in 7, `device.ua` in 5, and the remainder in one to four. The recurring core is four fields, not fourteen.

Second, the count depends on how generously we read the sources. Restricting to fields the primary source states explicitly, and discarding everything we marked as mechanically implied, the figure falls from 16 of 17 to **12 of 17**, with three rows then contributing no indexed field at all. That is the conservative bound and it is the one to argue with. Across the 82 field codings in the corpus, 72% are graded documented, 23% inferred and 5% documented at family level rather than for the specific scheme.

Third, the single exception is more interesting than the rule. **ViperBot falsifies no class-C field**, because it does not lie about the session at all. It removes the measurement tags from the delivered ad and reinserts them into cheap slots on real devices, so the discrepancy signal that would expose the undelivered impression never arrives. Our framework assumes that if a verification hook executes, the check it performs is sound. ViperBot attacks the hook. Section 5.2 records this as a threat model the Verifiability Index does not represent, and it is the only row in the mitigation matrix that nothing catches outright.

The most-falsified field in the entire corpus is not in the class-C list at all. **`app.bundle` is falsified in every one of the 17 rows, and we code it class B.** Section 6.2 is about why that is not the contradiction it looks like, and we put it here rather than later, because a reader who notices it before we raise it will reasonably

stop trusting the analysis. Four other class-B fields were also falsified: `app.name` in 5 rows, and `app.storeurl`, `app.publisher.id` and `bid.adm` in one each.

The mechanisms are monotonous once the index is in hand, and that monotony is the finding. ICEBUCKET falsified the device values that server-side insertion relays through HTTP headers, appropriating more than 300 legitimate app IDs from real publishers, at a claimed peak of 1.9 billion requests per day. LeoTerra and StreamScam forged household IP addresses, app identifiers, device identifiers, models and OS versions across roughly 3,600 apps, and scaled device diversity by bulk-downloading public device-information lists. DiCaprio fabricated Roku requests from inside a mobile app, rotating 98 bundle identifiers, 114 store URLs, 134 app names and 11 device models with the video dimensions hardcoded to 1920 by 1080. MultiTerra hijacked thousands of residential IP addresses and had a single IP impersonate 16 different handsets across nine apps inside 20 minutes. PARETO ran an SDK on roughly a million infected Android phones that impersonated CTV operating systems on command from a control server, spoofing more than 6,000 CTV apps. NewsJunkie, disclosed in July 2026 with the full modern standard set deployed, generated server-side requests with forged device, app and IP details behind residential proxies, and its discoverer's stated explanation for why it worked is the absence of client-side detection signals in CTV.

Two rows deserve individual attention because they mark the boundaries of the argument.

RapidFire is this paper's thesis with the botnet removed. In August 2021 Method Media Intelligence disclosed an operation with no bots, no malware and no fraudulent apps: a script emitting counterfeit JSON bid requests directly into open CTV auctions across multiple supply-side platforms, reportedly earning its operators around ten million dollars a month. Every other scheme in this corpus builds infrastructure whose only purpose is to make fabricated traffic look observed. SDKs on real phones, hijacked residential IP addresses, downloaded device lists, TLS cipher configurations tuned per declared user agent. RapidFire skipped all of it, because on a delivery path where the creative goes to a stitching server and no render signal is ever required back, that infrastructure buys nothing. It is also one of only four CTV fraud operations whose perpetrators have been publicly identified as people or companies rather than codenames.

LowerTerra shows an adversary defeating a consistency check by controlling both sides of it. DoubleVerify publishes decrypted instructions from an OctoBot command server showing a user-agent value set per request, paired with a matching TLS cipher configuration, so that the declared device and the observed encryption behaviour agree. This is directly relevant to Section 3.3's falsification criterion: a check that compares two values an adversary supplies is not a check, and here we have the adversary's own configuration file proving it.

6.2 The claim we are not making

A naive version of this paper would stop at "fraud only touches unverifiable fields." That claim is false and we want to be the ones to say so. `app.bundle` is falsified in all 17 rows and we code it class B. Four other class-B fields were falsified too. Those fields have a check available in principle, and the schemes worked anyway. There are three distinct reasons, and separating them is what turns the observation into an argument.

First, **authorization is not identity binding.** The app-ads.txt check answers whether a seller may sell inventory for a bundle. It does not answer whether the traffic in front of you came from that bundle. A scheme that fabricates sessions for a bundle whose sell path it is authorized to use passes the check by construction. Monarch, monetizing its own registered Roku channels through its own ad system, and PEACHPIT, spoofing device identity from inside 39 of its own real store-listed apps, both look like this.

Second, **the operator can choose which app to be.** This is the reason Section 7 reports its counterfactual as a range rather than a number. Rejecting a spoofed app identity does not remove a revenue mechanism that is indifferent to which app identity it declares. The relevant question is what the alternative costs, and

DoubleVerify measured it: over 1,300 fraudulent CTV apps identified in 18 months as of August 2020, more than half of them flagged in 2020 alone. DoubleVerify also names the reason those apps are cheap to make, in terms this paper would not improve on: build them from public-domain content, submit them to a store on a premium platform to inherit that platform's credibility, and then, "because of the way most CTV ad inventory is delivered (via SSAI), information about the placement is often self-declared."

Third, **reachability**, which Section 7.3 measures: for most of the app population the walk still cannot be run at all, so the check whose limits we are describing is in most cases not even attempted.

The precise claim is therefore sharper than the naive one. The fields that make fabrication possible have no check. The fields that do have a check have one that binds a seller to inventory rather than traffic to identity, and the cheapest way past it is to become the seller. Both halves are necessary to explain the corpus, and neither is fixed by more adoption of what exists.

---

7. What the standards would and would not have stopped

For each scheme and each deployed standard we asked a counterfactual: at 100% correct adoption and enforcement, would the documented mechanism be caught, partially constrained, or unaffected? Every cell in the published matrix carries a one-sentence justification so disagreement can be aimed at specific cells.

Asking that question carefully turned out to require asking it twice, and an earlier version of this section did not, which made its headline number an artifact of an unstated inconsistency. The problem is specific to the authorization standards. An authorization file answers whether a seller may sell a named app's inventory. Whether that rejects a scheme depends on which of two things we are scoring:

- the **declared reading**: the app identity the scheme is documented to have declared. No adaptation is credited. If the operator spoofed a third party's app, the walk rejects it, full stop.
- the **adapted reading**: the same fabrication mechanism, pointed at apps the operator owns and authorizes itself.

Neither reading is the right one on its own. The declared reading credits a standard with everything it catches against the scheme as documented, which is what a counterfactual should do. The adapted reading asks whether the revenue survives, which is what a defender cares about. The matrix now carries both verdicts in every authorization cell, and this section reports both.

7.1 What the authorization layer rejects, and what survives it

On the declared reading, the deployed standards reject the app identity that 12 of the 17 schemes actually declared. Four survive all five standards: Monarch, SneakyTerra, ViperBot and PEACHPIT. A fifth, RapidFire, cannot be scored, because its discoverer published no field itemisation and so whether the counterfeit requests named third-party apps or the operators' own is not on the public record. We score that cell `unresolved` rather than guess it, which puts the survivor count at 4 or 5 of 17. Dropping the two low-confidence rows moves it to 3 or 4 of 15, so the finding does not rest on the weakly sourced cases.

On the adapted reading, all 17 survive. Every rejection in the paragraph above is a rejection of a declared identity, not of a mechanism. Two rows, DiCaprio and PARETO, are scored `degraded` rather than `unaffected`, because their yield came from the brand value of the publishers they impersonated and self-owned apps do not carry it. For the other fifteen, the fabrication is indifferent to which app it names.

The gap between those two numbers, 12 rejected and 0 rejected, is the whole argument of this section, and it would be cheap if the adapted reading were speculative. It is not. Three schemes in the corpus already operated that way at the time of their disclosure: Monarch through its own registered Roku channels,

SmokeScreen through its own store-listed screensaver apps, PEACHPIT through 39 of its own apps with 15 million-plus real installs. And the category has been measured by the same vendor that discovered most of this corpus: over 1,300 fraudulent CTV apps in 18 months as of August 2020. Adaptation is not a hypothetical an adversary might attempt. It is the observed majority behaviour of the fake-app population, and it costs a store submission.

So the honest form of the finding is not a single number. It is a conditional: **the authorization layer, fully adopted and enforced, would have forced most of this corpus to change which app it claimed to be, and would have stopped none of it from fabricating sessions.**

Two rows in the matrix are worth naming because nothing in it catches them outright, under either reading and including the attestation mechanisms of Section 8. Monarch's every declared value is true: real apps, real installs, real renders, real registered sellers. What it misrepresented was the content and the audience, and no mechanism here attests either. ViperBot's every declared value is true as well, because it attacks the measurement rather than the declaration. They mark the two failure modes that this paper's framework does not reach, and Section 8.2 is about the first while Section 5.2 is about the second.

7.2 Why authorization files cannot catch fabrication

Every deployed standard in the list answers a question of the form "is this party permitted to sell this inventory." None answers "did this impression occur." That is not an implementation shortfall; it is what the standards were designed to do. It is worth splitting the design motive, though, because a single sentence about it is half wrong. ads.txt and app-ads.txt were designed in response to spoofing, and say so: the app-ads.txt specification names the case where "the domain of the webpage, or the ID of the mobile app has been falsified." sellers.json and the SupplyChain object were not. Their stated gap is different: ads.txt authorizes sellers without revealing who they are or which intermediaries handled the request. That is a transparency problem, not a spoofing problem, and it is why those two standards score `partial` on four rows and `caught` on none.

The web contrast case makes the limit visible. Merry-Go-Round did not evade ads.txt. It operated its own domains, which published their own ads.txt files authorizing its own selling accounts. The authorization layer returned a clean answer because the answer it gives is about permission, and the operator had granted itself permission. An authorization file cannot reject self-authorized inventory, on any channel. Read alongside the adapted reading in Section 7.1, Merry-Go-Round is not merely a contrast case. It is the same move the CTV schemes can make, already executed and measured on a channel where the walk is cheap and universally reachable.

ads.cert 2.0 deserves separate mention because it is the newest and most cryptographic of the set, and because our previous treatment of it was wrong in a way worth correcting explicitly. It is not transport signing. It is an application-layer message signature: HMAC-SHA256 over the request body and URL, keyed by a shared secret derived via X25519, with keys published in DNS TXT records. Because HMAC is symmetric, the assurance is pairwise and non-transferable, which the specification itself records as a recipient forgeability loophole, so a signature cannot be relayed downstream as evidence to a third party.

That makes its scoring split rather than uniform. It catches **one** corpus scheme outright, and the row is instructive: Colorius ran over 400 counterfeit SSAI servers, and servers impersonating providers whose keys they do not hold cannot produce a valid signature. Authenticated Connections genuinely breaks that. What it does not break is an operator that registers as its own SSAI provider and signs its own fabricated requests correctly, which is why it scores `partial` on 13 rows and `unaffected` on 3. Our previous claim that it catches zero schemes understated it on our own terms, and the corrected version is more useful: it defeats impersonation of a named party and does nothing about self-declared content, which is the same boundary every other standard in this section runs into.

For deployment, though, the caveat is severe. The reference implementation's last substantive commit was in 2023, it has no releases, and it still describes itself as a proof of concept. A DNS probe of the location the specification mandates, across fifteen major adtech domains including IAB Tech Lab's own, returned zero Call Sign records. Against that, TAG's Certified Against Fraud programme requires ads.cert for SSAI billing notifications from certified companies, so it is not entirely inert. We score it at spec level and say so.

7.3 The reachability measurement

The counterfactual above grants the standards 100% adoption. Reality is not merely short of that; for CTV app authorization it is structurally short, in a way we can measure.

The IAB app-ads.txt discovery procedure requires a crawler to read the developer's website URL from the app's store listing and then fetch `app-ads.txt` from that domain. Roku publishes the necessary values as HTML meta tags (`appstore:developer_url`, `appstore:store_id`, `appstore:bundle_id`), so absence of the developer URL is not a scraping artifact: it means the specification's discovery step has nothing to read.

We built a sampling frame from the complete public Roku channel-store sitemap, 31,647 channels, and drew two samples: a uniform random sample of 400, and a purposive sample of 37 flagship channels of widely recognized ad-supported services. Figure 5 shows where the walk stops.

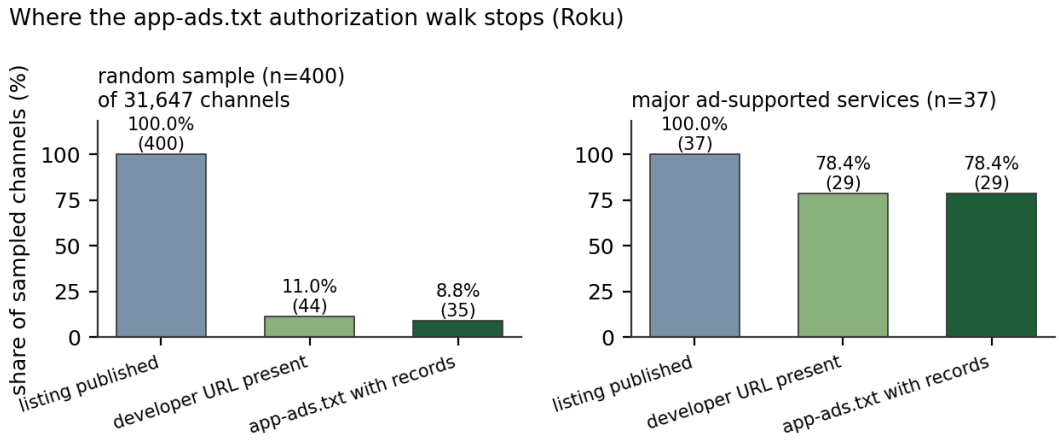

Figure 5. Where the app-ads.txt walk stops on Roku. The failure is concentrated at the first hop. For 89.0% of the random sample the store listing publishes no developer URL, so the specification's discovery procedure cannot begin, independently of whether the publisher would have complied.

Table 5. The app-ads.txt walk on Roku. Wilson 95% intervals in brackets.

stage	random, n=400	major services, n=37
listing publishes a bundle identifier	400 (100%)	37 (100%)
listing publishes a developer URL	44 (11.0%) [8.3, 14.4]	29 (78.4%) [62.8, 88.6]
`app-ads.txt` served with valid records	35 (**8.8%**) [6.4, 11.9]	29 (**78.4%**) [62.8, 88.6]
completion once the walk can start, per channel	79.5% [65.5, 88.8]	100%
completion once the walk can start, per unique developer domain	73.5% (25/34)	100%

The authorization check that would let a buyer reject an unauthorized seller for a declared bundle is executable for 8.8% of the Roku channel population. Every channel offers an identifier a bid request can carry; fewer than one in nine offers the means to check who may sell it.

The attribution matters more than the headline. Where a developer URL exists, the developer usually does publish the file, at 79.5% of channels and 73.5% of unique developer domains once the ten channels sharing a developer are deduplicated. So roughly one startable walk in four still fails, which we should not call “nearly always” as an earlier draft did. The dominant failure is nonetheless the store-listing bootstrap rather than publisher compliance, and that points at a specific and cheap remedy: platforms could require a developer URL at channel submission. The recommendation is available only because the measurement separates the two failure points.

The purposive stratum needs a correction that cuts against how we first described it. We labelled it “flagship ad-supported services”, but eight of the 37 publish no developer URL and six of those eight are subscription or walled-garden services with little or no open programmatic inventory (Netflix, Prime Video, YouTube, Starz, BritBox, Crunchyroll), for which app-ads.txt was never the relevant mechanism. Restricted to services that actually sell open programmatic CTV inventory, the rate is 29 of 31, **93.5%**, and the two genuine failures are **Freevee** and **Xumo Play**, both significant ad-supported services. We report both framings: 78.4% for the stratum as constructed, 93.5% for the subset the label actually described.

That correction sharpens rather than blunts the comparison, because it widens the distance between the population and the services buyers recognize: 8.8% against 93.5%.

One data-quality observation is worth recording. The Roku Channel’s `developer_url` field contains `https://therokuchannel.roku.com/app-ads.txt`, that is, the file URL pasted into the field meant to hold the developer’s site, which a specification literal crawler would resolve to `.../app-ads.txt/app-ads.txt`. Our crawler happens to recover, because it takes the host component of every developer URL unconditionally and discards any path. That is a blanket normalization rather than targeted repair, and it makes our result conservative in the sense that it can only find more files than a stricter reading would, never fewer.

7.4 For comparison, the layer that does work

To establish that we are not simply describing an industry where nothing functions, we ran the analogous check on the web side. Taking every `source.schain` node observed in a residential-vantage capture of web display traffic (19 unique ad system domains, 53 unique node references), we fetched each named domain’s `sellers.json` and tested whether the declared seller identifier actually appears in it.

16 of 19 domains (84.2%) publish a parseable file, and 48 of 50 checkable node references (96.0%) resolve to a disclosed seller. Two references under one ad system are dangling: seller identifiers absent from a file that contains 2,621 other sellers, which a receiver enforcing the check would reject. One of the three failures is instructive: the file is served with HTTP 200 but is invalid JSON, containing an unquoted hexadecimal `seller_id`, so a receiver cannot parse it and the check silently fails open. That is the same class of data-quality failure that the prior paper in this series found in the bid stream itself.

The comparison is the point, and it should be stated with both rates visible: the cross-reference layer is executable for 84.2% of the web ad systems we observed and for 8.8% of the CTV channel population. We are applying the same instrument to both and getting an order-of-magnitude difference in whether the check can be run at all. Note also that 84.2% is not a clean bill of health: a 15.8% failure rate on a deployed standard would be a finding in its own right in a paper about web. This is why the paper treats web as a baseline rather than as a second failure case.

Limitation, stated here rather than only in Section 11: these are web-vantage node references. Measuring CTV supply chains at the wire requires a CTV bid-stream vantage we do not have, and the reason we do not have one is the same architectural fact the paper is about.

8. Inverted ranking: where the leverage is

A matrix of what does not work implies a ranking of what would. Inverting it (Figure 3) ranks mechanisms by how many corpus schemes each breaks. app-ads.txt appears twice in that figure, once per reading, and the pair is the point rather than an untidiness.

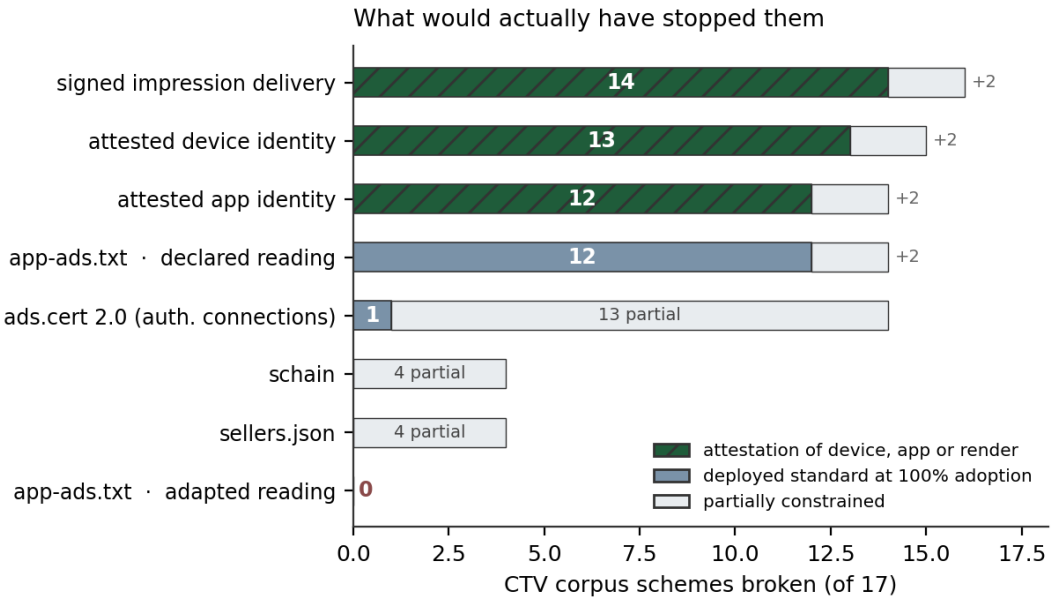

Figure 3. Mechanisms ranked by CTV corpus schemes broken, of 17. Solid segments are schemes the mechanism rejects outright, pale segments those it partially constrains. app-ads.txt appears twice because the two readings of Section 7.1 give 12 and 0. No attestation row has that property, and the asymmetry is the finding rather than an artifact of scoring.

Table 6. Mechanisms ranked by CTV corpus schemes broken, of 17.

mechanism	status	caught	partially constrained
per-impression signed delivery	not standardized	**14**	2
attested device identity	standardized Oct 2025, client-vantage dependent	13	2
attested app-instance identity	not standardized	12	2
app-ads.txt, declared reading	deployed	12	2
app-ads.txt, adapted reading	deployed	0	0
ads.cert 2.0	deployed	1	13
schain	deployed	0	4
sellers.json	deployed	0	4
ads.txt	deployed	0	0

Two things in that table need saying rather than leaving to the reader.

app-ads.txt appears twice, and the pair of rows is the finding rather than an untidiness. Under the reading that scores the app identity each scheme declared, it is the equal of app-instance attestation. Under the reading that scores the fabrication mechanism, it catches nothing. No attestation row has that property, and the asymmetry is not an artifact of how we chose to score: an operator can move fabricated traffic into an app it owns for the price of a store submission, and cannot move fabricated traffic into a device it owns

without buying the device. That converts a marginal-cost-zero operation into a capital-intensive one. The ranking is measuring that asymmetry, not the cleverness of any individual check.

And ads.cert 2.0 is not a zero. It catches Colorius, whose 400 counterfeit SSAI servers could not have signed as the providers they impersonated. Our previous draft scored it at zero, which understated it on our own terms.

Excluding the two low-confidence rows the ordering is unchanged, so it does not rest on the weakly sourced cases.

The single most useful sentence we can offer a standards body: **the mechanism that would break the most documented CTV fraud is the one nobody has standardized**, and it is not exotic. It is a signed statement, from the party that actually rendered the advertisement, that the advertisement was rendered.

8.1 This is not hypothetical, and it is now partly standardized

Two deployments establish that device attestation works in practice.

CycloneBot, disclosed by DoubleVerify with Roku in December 2023, was confirmed using the Roku Advertising Watermark. The sequence matters and we had it slightly wrong in an earlier draft. DoubleVerify's behavioural detection found the scheme, in the impressions-per-device distribution and the ISP distribution; Roku's watermark signals then confirmed that Roku was not impacted and that no fraudulent impressions had been sold or bought from Roku, which DoubleVerify states "helped validate DV's findings and confirm the spoofing of CTV devices." The watermark did not do the catching. It supplied the ground truth that settled a statistical suspicion, which is precisely the function Section 5.5 argues CTV lacks, in precisely the position Section 5.5 argues it works: after the fact, not as a gate.

Two clarifications about the mechanism, since both are commonly got wrong. The Roku Advertising Watermark is not a content watermark: it is a signed JWT obtained from an API available only on authentic Roku devices, appended to measurement beacon headers, and version 2.0 added app-level attestation, target-domain binding and bid-stream passability. It is also not RIDA, which is a resettable and forgeable UUID.

More consequentially, IAB Tech Lab standardized the mechanism in October 2025 as Device Attestation in OM SDK 1.6, on IETF Privacy Pass, with the major verification vendors as adopters. Section 5 sets out why this does not close the gap this paper measures: it needs an OM SDK client context that server-side insertion removes, it covers Apple devices and Fire TV rather than Roku, Samsung, LG or Vizio, and it is sampled at seller level rather than checked per impression. The ranking in Table 6 therefore models what the mechanism can do, not what is available on SSAI inventory in 2026.

It is instructive about the shape of the solution as well as its feasibility. The watermark works because Roku certifies traffic that third parties claimed came from Roku devices, so the attester is independent of the party the claim benefits. By the independence clause of Section 3.3 this is A-independent attestation. The same platform attesting inventory it sells itself would be A-self, and would prove nothing about that inventory. The current state is therefore a set of proprietary, per-platform islands, which is the ecosystem-level version of the problem a standard exists to solve.

8.2 What attestation cannot fix

Three corpus schemes are caught outright by none of the attestation mechanisms we consider, and naming them is necessary to keep the recommendation honest. Each is scored as partially constrained on at least one mechanism rather than wholly untouched, but what survives in each case is the part that matters.

Monarch monetized its own registered Roku channels through its own registered ad system. The apps were real, the devices were largely real, and the advertisements genuinely rendered. The pitch was brand-safe

classic television; the delivery was screensaver and pet-video apps, registered under a “special interest” category to get past Roku’s restrictions on screensaver advertising. Device attestation constrains only the device-farm component that Pixalate’s impression concentration points at, and the organic installs pass.

A clarification is needed here because the ground has shifted since 2020. Screensaver inventory is no longer illegitimate by nature: IAB Tech Lab standardized a CTV screensaver ad format in December 2025. What Monarch did wrong was not selling screensaver inventory. It was selling screensaver inventory as something else, and evading a platform restriction to do it. The example survives the standardization, and arguably sharpens with it: now that the format is legitimate, the only thing separating a compliant screensaver buy from Monarch’s is an accurate content object, and the content object is class C on every path we coded.

SmokeScreen ran real user-installed apps on real external streaming devices, rendering advertisements continuously while the television screen was off. Catching that requires attesting display state, whether the panel was actually on, which no proposed standard covers and which the delivery platform is not positioned to observe.

ViperBot is the third and it is a different kind of failure. It did not lie about a session at all: it stripped the measurement tags out of the delivered advertisement and fired them from cheap slots on real devices, so the discrepancy signal that should have exposed the undelivered impression never arrived. A signature bound to a specific creative in a specific placement would catch the redirection. A bare attestation that some render happened on some genuine device would not, and would in fact certify the substituted slot. Read together with Section 5.2, ViperBot marks the assumption the Verifiability Index makes and does not examine: that if a verification hook executes, the check it performs is sound.

That leaves two schemes, Monarch and ViperBot, that nothing in the whole analysis catches outright, and the distinction between them bounds the paper’s recommendation in two directions.

Attestation addresses **fabrication**: claims about sessions that never happened. It does not address **misrepresentation**: real sessions truthfully reported and sold as something they are not. Content-object claims (`content.title`, `content.series`, `content.genre`, `content.livestream`, `content.network`, `content.channel`) are class C in every channel and every tier we measured, no attestation mechanism touches them, and CTV pricing depends on them heavily. And it does not address **measurement suppression**, where the declared values are all true and the verification hook is the thing under attack. Section 10 returns to the first as the most likely direction of migration, and we flag the second as outside the instrument entirely rather than as something the instrument scores badly.

8.3 The incentive asymmetry

We keep this brief because it is analysis rather than measurement. The party best positioned to attest delivery is the platform or device that renders the advertisement. The party that bears the loss when delivery is fabricated is the buyer. The party that would carry the integration cost is largely the seller. Attestation is therefore a mechanism whose cost and benefit sit with different parties, which is the standard precondition for a coordination failure and the standard argument for a shared standard rather than per-platform solutions. The AAO Verified programme’s retreat from attesting live delivery to attesting sandbox conformance, documented in Section 9, is a concrete instance of that cost pressure operating.

9. The agentic layer: AdCP

Everything above is retrospective. The reason to publish it now is that a new protocol layer for agent-to-agent media buying is being specified at the time of writing, and it is possible to ask whether it inherits the gap.

We audited AdCP version 3.1.1, repository state at commit `c68d54542` (2026-07-03), on 2026-07-25, applying the same instrument. Trusted Match, measurement, campaign governance, sponsored intelligence and verified identity attestation carry an explicit experimental status and are outside the specification's stability guarantee, so this is a snapshot of a moving target and is offered as input to a working group rather than as a verdict.

9.1 What AdCP is, and therefore what to ask of it

AdCP is not an OpenRTB replacement and does not claim to be. Its own documentation is explicit that it handles the campaign workflow while OpenRTB handles the impression-level auction (`docs/faq.mdx:126`), and that planning constructs do not propagate into bid requests. Consistent with that, it carries no device identity at all: there is no advertising identifier, IP address, user agent, device make or model, household or session field anywhere in its core or media-buy schemas. Device data is categorical (a `ctv` device type, a `roku_os` or `tizen` platform) and the familiar OpenRTB device fields appear only as macro names an ad server fills in.

So the question is not whether AdCP fixes the CTV device fields; it never sees them. The question is whether the new layer adds verification of **delivery**, since Section 8 ranks per-impression signed delivery first.

9.2 It does not attest delivery, and declines to do so permanently

Four independent confirmations, all from the normative surface rather than the explanatory documentation:

1. Read operations, including the delivery report, are listed as out of signing scope in both version 3.0 and the planned 4.0, on the stated reasoning that signing exists to protect state-changing operations.
2. No response-signing profile applies. The specification states that transport response signing is not defined for any task in the 3.x line, and the list of tasks permitted to sign a response is closed at two brand-verification calls.
3. The delivery schemas contain no signature or attestation member.
4. The specification says so in as many words, describing the underlying delivery platform as the system of record for counting and noting that the protocol itself is not accredited for measurement: "AdCP is the wire and the contract, not the ledger" (`docs/reference/known-limitations.mdx:71`). Invalid-traffic filtration and viewability are explicitly located outside the protocol.

By default the seller is the authority: where billing measurement terms are absent, the specification states the default is seller-attested. Even the third-party-measured path is a relay, since the seller pulls the vendor's numbers on its own schedule and publishes them, and the specification notes that the vendor itself does not call the protocol.

Most decisively, the exclusion is deliberate and permanent. Under a heading reading "Out of scope (ever)" (`docs/building/by-layer/L1/security.mdx:585`), the specification states that threading a signed attestation through per-impression bid requests is "operationally infeasible … and unnecessary," on the reasoning that spend authorization happens at media-buy time rather than per impression.

That reasoning is coherent for the problem AdCP set out to solve, which is authorizing and orchestrating spend. It is precisely orthogonal to the problem this paper measures. A protocol built from scratch for autonomous agents, with five application-layer signing surfaces, a JSON Web Signature profile, publisher key pinning and a key-transparency roadmap, breaks none of our seventeen schemes by the delivery route, and documents why it chose not to try.

9.3 The transport asymmetry

One incidental finding is clean enough to state on its own. The same delivery numbers carry different integrity properties depending on direction of travel. Pulled through the delivery report they are protected by transport security only. Pushed as a webhook they are signed under RFC 9421 with a required content digest, verifiable at rest after the session closes.

A buyer that receives delivery by webhook therefore holds a non-repudiable artifact of the seller's own numbers. By the independence clause this is A-self attestation: it converts undetectable overcounting into attributable overcounting. The specification agrees about what a signature proves, stating that a valid signature proves only that the request came from the agent whose key signed it.

9.4 The mechanism exists, pointed elsewhere

The sharpest observation available in this audit is not an absence but a misallocation.

AdCP requires genuine A-independent attestation for one class of claim. Forwarded proof-of-personhood and age attestation (World ID first, designed to be issuer-agnostic) must be verified by the receiver, and the specification instructs that an absent or unverifiable attestation MUST be treated as no attestation rather than as an asserted-true claim. It is bound against replay with a nullifier and a signal binding, carries a hard expiry, is scoped to relying parties published in the brand's own file, and the attestation bytes are inside the signed envelope. The experimental status page describes the intent in almost exactly this paper's vocabulary: buyers verify rather than trust an assertion.

That is what a class-A mechanism looks like, built by this industry, in this protocol, in this specification cycle. It attests **whether the viewer is a unique human over 18**. Nothing in the protocol attests **whether the impression happened**.

9.5 What AdCP genuinely improves

The audit is not a dismissal, and three improvements are real.

Its authorization file is a strictly richer app-ads.txt: publisher-hosted, scoped by property, placement, delegation type, country and validity window, with optional key pinning, and it comes with a task that automates the delivery-side authorization check and reports unauthorized impressions. Our Section 7.3 finding suggests the binding constraint on such files is reachability rather than expressiveness, but the expressiveness is a genuine advance.

Second, and specifically relevant to CTV app spoofing, a brand can be asked whether a claimed property is theirs, and the answer is signed. Property types include CTV applications explicitly, with store enumerations covering Roku, Fire TV, Samsung, LG and Vizio. The trust model is asymmetric in the useful direction: a denial is authoritative and unilateral, while an assertion of ownership is informative but not trust-extending. A buyer can therefore obtain a signed, authoritative "that CTV app is not mine" from the party in the best position to know. This is a real falsifying mechanism that OpenRTB plus app-ads.txt does not provide, and we code it A-independent with a borderline flag, on the reasoning that for an ownership claim the owner is the authority, so authorship and truth coincide for the negative answer.

Third, the protocol makes "whose count is this" a machine-readable qualifier, distinguishing seller-attested from vendor-attested completion, and the specification names the CTV case directly, noting that the two paths can differ materially in server-side insertion environments because the player's view of completion may differ from a vendor's. We exclude this from class B under our bilateral-contract rule, since the reference it is checked against is a negotiated commitment rather than public infrastructure, but as a conceptual advance it is the most interesting thing on the delivery axis: it does not verify the number, and it does force disclosure of whose word the number rests on.

Two further mechanisms deserve mention because they illustrate the vantage rule operating on cryptography. Content provenance can reference a C2PA manifest, a real cryptographic binding, but the schema names its own limitation: file-level bindings break when ad servers transcode or re-encode assets. Server-side insertion transcodes by construction, so a class-A mechanism on the web is class C on the CTV path for architectural reasons rather than adoption reasons. And the trusted-match envelope does sign property and placement identifiers, but its signature is static and cacheable for a 24-hour epoch across all requests for the same triple, verification is recommended at a 5% sample rate, and its own stated claim is that the request came from an authorized router. It cannot distinguish one impression opportunity from a million, so we code it B rather than A.

9.6 New surfaces where lying is cheap

The agentic layer also introduces assertion surfaces that have no OpenRTB equivalent, are pricing-relevant, and are class C. They include the natural-language campaign brief that drives which inventory a seller offers and at what price; a publisher-generated content summary delivered at serve time, which the specification itself instructs buyers to treat as untrusted publisher-generated content and ships anyway because agent buyers want it; a seller-asserted brand-safety pass or fail consumed as a buyer filter; a buyer-declared normalized performance score with a self-asserted provenance label, which steers seller optimization and is the cheapest lie in the protocol with a direct delivery consequence; and paired unsigned declarations about whether paid influence was disclosed inside an AI assistant's answer, where the harm is not measurable from the wire at all.

9.7 Verification language outrunning verification mechanism

Finally, an observation about specification hygiene that is a finding in its own right. Three places in AdCP's documentation describe verification that its normative profile does not provide. Buyer-facing material describes product-discovery responses as signed, including a worked verification example, where the security profile states that sellers must not apply response signing to synchronous responses and lists that task as out of scope. A schema and a governance page instruct buyers to verify signed agent responses against pinned keys, where no such response signing exists outside the two brand tasks. And the ecosystem verification programme still advertises canonical test campaigns running against a seller's real ad server, while the current specification replaced that with attesting correct handling of sandbox-flagged traffic, on the stated grounds that the sandbox version is universally achievable with no new operational infrastructure.

That last change is the most consequential for this paper. The one ecosystem mechanism that would have observed real delivery was withdrawn on cost grounds and replaced with wire-format conformance, and the specification is candid that a protocol wrapper around a stub ad server can honestly earn the weaker qualifier. Credit where it is due: AdCP documents its own limits, including that its trust anchors amount to trust-on-first-use with continuity, more honestly than the incumbent standards document theirs. The gap we report is not concealed. It is written down.

10. Where fraud goes next

This section is a forecast, dated July 2026, and offered because the index supports it rather than because prediction is the paper's purpose. The reasoning is mechanical: look for fields that are class C in every tier, are pricing-relevant, have no mitigation in flight, and carry increasing price leverage.

Content-object claims are the strongest candidate. `content.title`, `content.series`, `content.genre`, `content.livestream`, `content.network` and `content.channel` are self-declared on every path we coded,

no public registry binds them to a stream, no proposed standard addresses them, and CTV's move toward content-level and channel-level buying is increasing their price impact. They are also the fields against which attestation offers no help at all, per Section 8.2. Monarch already misrepresented content in 2020, before content targeting mattered as much as it does now.

Co-viewing multipliers are structurally attractive. The impression quantity multiplier scales billed impressions directly from an unsigned vendor claim relayed by the seller. A mechanism that multiplies billing and has no falsifier is an unusual thing to ship.

The agentic surfaces from Section 9.6 are new, unmeasured, and priced. Performance feedback is the one we would watch first, because a buyer can steer delivery by misreporting it and a seller can inflate its own attractiveness in discovery, so both sides have a use for it.

A systematic search for named schemes in each of these categories returned nothing, and the absence is worth reporting as evidence rather than as a gap. There is no named scheme anywhere in the public record for co-viewing multiplier inflation, content-object or metadata spoofing, ad pod stuffing on CTV, physical CTV device farms, or CTV made-for-advertising and free ad-supported television arbitrage. Peer39 quantifies the last category at roughly ten thousand channels without naming an operation, and Monarch's concentration signature, 58% of users producing 76% of impressions, was read as a device farm by its discoverer but never labelled as one. So this forecast is not a claim that these surfaces are unexploited. It is a claim that the most machine-checkable signals in CTV, and the one that multiplies billing directly, have produced no named case at all while eight years of device-fabrication schemes were being named and counted. Either they are not being exploited, in which case the forecast is a real prediction, or they are and nobody is looking in a way that produces a name.

10.1 The counter-development: server-guided ad insertion

A paper arguing that CTV's verification deficit follows from its delivery architecture has to address the architecture being adopted to undo it, and we would be describing a stale problem if we omitted it.

Server-guided ad insertion (SGAI) keeps the seamless-playback property that made server-side stitching win while returning ad resolution and measurement to the player. Rather than splicing transcoded creative into a single manifest, the server signals an interstitial break, using HLS Interstitials or DASH event mechanisms, and the client resolves the advertisement, renders it, fires the beacons and runs Open Measurement SDK instrumentation in its own execution context. The Streaming Video Technology Alliance standardized it in July 2025 and IAB Tech Lab took it up later that year.

On this paper's terms, SGAI is the correct fix, and it is worth being precise about why. It does not add a check. It restores the vantage. Every class-C coding on the CTV server-side path in Section 5 rests on the receiver having no client-side observation point; SGAI gives the observation point back, which is what converts the transaction from one where declarations are unfalsifiable to one where they can be contradicted. The mechanism our ranking places first, a signed statement from the party that rendered the advertisement, becomes possible to site once there is a rendering party in the loop again.

Three qualifications keep this from being a conclusion. It is new: standardized in 2025, with deployment not yet measurable at the time of writing. It is opt-in per publisher and per platform, so the long tail where Section 7.3 locates the reachability failure is the last place it will arrive and the place it matters most. And restoring a client context is necessary rather than sufficient: ViperBot ran on real client renders and defeated the measurement anyway. SGAI moves the CTV transaction toward the web position, and Section 5.5 is precisely an argument that the web position is a post-hoc correction loop rather than a solved problem.

We state the forecast as dated and falsifiable. If a documented scheme in 2028 turns on content-object spoofing or multiplier inflation, this section was right for the right reason. If CTV fraud instead migrates

somewhere our index scores as verifiable, the instrument needs revision and we would want to know. And if SGAI adoption becomes measurable, the right follow-up study is the index recomputed against SGAI inventory, which we would expect to sit much closer to the web figure and which would make this paper's central number a statement about a specific historical architecture rather than about a channel.

11. Limitations

Single-coder judgment, partially quantified. The codings are one author's reading. Section 5.4 quantifies the risk on a 47-field stratified sample: strong replication for the CTV class assignment (kappa 0.895), a notational ambiguity affecting the web baseline, and a genuinely soft pricing-relevance boundary (kappa 0.408) whose disagreement direction we show to be conservative for our conclusion. The pricing-relevance tag should be read as an interval, not a point.

The corpus depends on vendor disclosure, and we can put a number on how badly. Every scheme was discovered and described by a commercial verification vendor with an interest in the description. Ten of the seventeen rows come from one vendor. Mechanism detail varies widely, two rows are graded low-confidence, and all headline numbers are reported with and without them. Field mappings flagged inferred never carry a headline claim alone, and the conservative documented-only bound is reported alongside every count in Section 6.1.

The more useful version of this limitation is a lower bound on what the corpus cannot see, taken entirely from the discovering vendors' own aggregate claims in the same documents that name the schemes. DoubleVerify reported 12 SSAI schemes caught in the year to March 2021, and nearly 20 schemes of all types in the year to January 2021 with roughly half manipulating SSAI, against six corpus rows for that period. It reported roughly 40 schemes and variants over 2022, most impacting CTV, against two rows. It reported six new CTV-falsifying SSAI variants in 2022 to August, and three new LeoTerra variants in the first half of that year alone. Its OctoBot report states that seven interconnected CTV schemes existed from November 2019, of which three are publicly named, so four documented CTV schemes exist that no corpus assembled from public disclosure can enumerate. And it identified over 1,300 fraudulent CTV apps in the eighteen months to August 2020. Every one of those figures is larger than the number of named schemes available for the same period, in most cases by an order of magnitude.

The disclosure record is thinning while the claimed problem grows, and that is a finding rather than an apology for our short tail. DoubleVerify named exactly one CTV-touching scheme across 2025 and 2026, ShadowBot, which is mobile-first, while claiming more than 50 distinct CTV bot attacks and variants in 2025 and 140% more CTV schemes in the first quarter of 2026 than in the same quarter of 2025. Naming stopped while claimed volume grew by roughly fiftyfold. HUMAN named fifteen operations between January 2024 and July 2026, two of them CTV, with none named between October 2025 and February 2026, the longest quiet stretch on record. HUMAN's CTV disclosures are also bimodal in rigor: ICEBUCKET and PARETO published Roku channel identifiers, developer names, user-agent counts and reverse-engineered TLS cipher emulation, while NewsJunkie names no app identifiers, bundle identifiers, device strings or sellers at all. A research method that depends on vendor disclosure is getting a worse input each year, and any future work in this line should assume that trend continues.

Two structural absences belong here as well. No criminal prosecution has ever been brought specifically for CTV ad fraud: every ad fraud conviction on record traces to desktop-era infrastructure. And Integral Ad Science, one of the three major verification vendors, has never named a single CTV scheme, despite having owned a CTV ad server since 2021. Of the seventeen schemes here, four have had a perpetrator publicly identified as a person or company. The rest are codenames.

No prevalence measurement. This paper says nothing about how much CTV fraud there is. Our scale figures are quoted from discoverers with attribution and are not independently verified. Note in particular that essentially every dollar figure in this literature is vendor-modelled inventory value at an explicitly assumed $20 CPM rather than audited loss, that the figures are not comparable across vendors, and that we never sum them. The first-and-largest claims are similarly relative to each vendor's own visibility and are mutually incompatible, so they cannot be assembled into a ranking.

The deployment-mode distribution is the measurement this paper most needs and does not have. Section 1 notes that server-side ad insertion can fire tracking from the server, from the client, or in a hybrid arrangement, and that our scoring assumes the server-side case. What fraction of CTV inventory runs in each mode is, as far as we can find, unmeasured in public. Pixalate's transparent versus non-transparent SSAI distinction is the closest available frame. Until that fraction is known, the index reported here should be read as the value for non-transparent SSAI rather than for CTV as a whole, and the honest summary of our headline number is that it describes an architecture whose prevalence within the channel we have not established.

Reasoned counterfactuals. Section 7 cannot be run as an experiment. Its value rests on per-cell justifications being individually contestable, which is why every cell carries one and why the ambiguity that produced the previous draft's single wrong number is now two labelled columns.

The index and the counterfactual are not independent instruments. `app.bundle` being class B and app-ads.txt rejecting spoofed app identities are the same judgment expressed twice. Where the paper cites both it states the shared dependency, and neither should be read as corroborating the other.

One platform, one channel-store measurement. The app-ads.txt result is Roku only. Roku is the largest US CTV platform by device installed base and publishes the frame that made the measurement possible, but one platform is not the ecosystem, and platforms differ in how they expose developer metadata. Extending this to Fire TV, Samsung and LG is the obvious next measurement, and store anti-automation on some of those platforms is the reason it is not in this paper.

No CTV bid-stream vantage. Our supply-chain cross-reference measurement is web-side. Obtaining a CTV wire vantage requires either platform cooperation or a demand-side seat, and the difficulty is itself an instance of the paper's thesis.

Membership-gated registries are a judgment call. Whether the TAG payment identifier registry or Ad-ID counts as public infrastructure is arguable. The affected fields are flagged borderline and listed.

Geographic consistency may fail our own falsification criterion. Eight `device.geo.*` fields are class B on web because a buyer can compare the declared city against the geolocation of the observed IP. A rational adversary who controls the IP can look that IP up in the same database and declare a matching city, which is the shape of check Section 3.3 disqualifies for residential-range plausibility. We have not recoded those fields to class C. Doing so would move the deployed headline gap from 21.5 to about 12.6 percentage points, coinciding with the mechanism-count variant already in Table 2. We leave them borderline, report both cuts, and treat the 21.5-point field-count gap as an upper bound.

Client-side CTV insertion is not scored. Section 3.2 names `ctv_csai` as a context. The released codings cover `web` and `ctv_ssai` only. Every CTV index number is the server-side case. CSAI and hybrid beaconing are the industry's obvious reply, and the index does not measure them.

AdCP is a moving target. Section 9 is a snapshot of version 3.1.1 at a named commit, with the surfaces most relevant to this paper marked experimental by the specification itself.

12. Conclusion

The buyer of a connected TV impression is in a structurally worse verification position than the buyer of a web display impression, and the difference is not adoption or diligence. It is that server-side ad insertion removes the observation point that web anti-fraud was built around. We measured the consequence: 91.5% of pricing-relevant request-side claims a buyer receives about a CTV server-side impression are self-declared against 70.0% on web (Table 1), 92.5% once reachability is accounted for, and no scored OpenRTB field on either path is attested.

The standards deployed in response to a decade of web fraud do not address this, because they answer a question about authorization rather than about occurrence. Fully adopted and enforced, they would reject the app identity that 12 of our 17 documented CTV schemes declared, and would stop none of them from fabricating sessions, because the same fabrication runs through apps an operator registers itself. That is not a hypothetical adaptation: three of the seventeen already worked that way, and the vendor that discovered most of them measured the fake-CTV-app population at over 1,300 apps in eighteen months. Meanwhile, on the largest US CTV platform, the app-level authorization check cannot even be executed for 89% of the channel population, because the store listing publishes nothing for a crawler to follow.

The mechanism with the most leverage is a signed statement from the party that rendered the advertisement that the advertisement was rendered. It would break fourteen of the seventeen.

The industry has begun building the adjacent piece. IAB Tech Lab standardized device attestation in October 2025 and the major verification vendors adopted it, which is the strongest counter-evidence to any reading of this paper as fatalistic. But it was built on the Open Measurement SDK, so it needs the client execution context that server-side ad insertion removes, and it ships for Apple devices and Fire TV rather than for the platform holding the largest share of US CTV households. Per-impression signed delivery remains unstandardized, and the protocol now being written for autonomous agents to buy media rules it permanently out of scope while requiring verified attestation of whether the viewer is a human over 18.

That is the paper's summary. The industry knows how to build verifiable claims, and in the last year it started. What it has not done is make one reach the architecture through which most CTV advertising is actually delivered, or point one at the claim its entire settlement layer depends on: that an advertisement played.

References

Academic works. DOIs were checked against the publisher record where it was reachable.

[1] Bashir, M. A., Arshad, S., Kirda, E., Robertson, W., and Wilson, C. *A Longitudinal Analysis of the ads.txt Standard*. In Proc. ACM Internet Measurement Conference (IMC), 2019. doi:10.1145/3355369.3355603.

[2] Bashir, M. A., and Wilson, C. *Diffusion of User Tracking Data in the Online Advertising Ecosystem*. Proceedings on Privacy Enhancing Technologies (PoPETs), 2018(4).

[3] Anselmi, G., Vekaria, Y., D'Souza, A., Callejo, P., Mandalari, A. M., and Shafiq, Z. *Watching TV with the Second-Party: A First Look at Automatic Content Recognition Tracking in Smart TVs*. In Proc. ACM IMC, 2024. doi:10.1145/3646547.3689013. arXiv:2409.06203.

[4] Crussell, J., Stevens, R., and Chen, H. *MAdFraud: Investigating Ad Fraud in Android Applications*. In Proc. ACM MobiSys, 2014.

[5] Dave, V., Guha, S., and Zhang, Y. *Measuring and Fingerprinting Click-Spam in Ad Networks*. In Proc. ACM SIGCOMM, 2012. doi:10.1145/2342356.2342394.

[6] Dong, F., Wang, H., Li, L., Guo, Y., Bissyandé, T. F., Liu, T., Xu, G., and Klein, J. *FraudDroid: Automated Ad Fraud Detection for Android Apps*. In Proc. ACM ESEC/FSE, 2018. doi:10.1145/3236024.3236045.

[7] Li, W., Li, H., Chen, H., and Xia, Y. *AdAttester: Secure Online Mobile Advertisement Attestation Using TrustZone*. In Proc. ACM MobiSys, 2015.

[8] Liu, B., Nath, S., Govindan, R., and Liu, J. *DECAF: Detecting and Characterizing Ad Fraud in Mobile Apps*. In Proc. USENIX NSDI, 2014.

[9] Mohajeri Moghaddam, H., Acar, G., Burgess, B., Mathur, A., Huang, D. Y., Feamster, N., Felten, E. W., Mittal, P., and Narayanan, A. *Watching You Watch: The Tracking Ecosystem of Over-the-Top TV Streaming Devices*. In Proc. ACM CCS, 2019. doi:10.1145/3319535.3354198.

[10] Papadogiannakis, E., Kourtellis, N., Papadopoulos, P., and Markatos, E. P. *Who Funds Misinformation? A Systematic Analysis of the Ad-related Profit Routines of Fake News Sites*. In Proc. The Web Conference (WWW), 2023. doi:10.1145/3543507.3583443.

[11] Papadogiannakis, E., Kourtellis, N., Papadopoulos, P., and Markatos, E. P. *Welcome to the Dark Side: Analyzing the Revenue Flows of Fraud in the Online Ad Ecosystem*. In Proc. WWW, 2025. doi:10.1145/3696410.3714899. Earlier version arXiv:2306.08418.

[12] Pearce, P., Dave, V., Grier, C., Levchenko, K., Guha, S., McCoy, D., Paxson, V., Savage, S., and Voelker, G. M. *Characterizing Large-Scale Click Fraud in ZeroAccess*. In Proc. ACM CCS, 2014. doi:10.1145/2660267.2660369.

[13] Sadeghpour, S., and Vlajic, N. *Ads and Fraud: A Comprehensive Survey of Fraud in Online Advertising*. Journal of Cybersecurity and Privacy, 1(4):804-832, 2021.

[14] Sekowski, A. *VAST XML Validation at Bid-Time Scale: Latency Analysis and Integration Patterns for Programmatic Video Pipelines*. Preprint, 2026. doi:10.13140/RG.2.2.11404.27520.

[15] Sekowski, A. *How Machine-Checkable Is OpenRTB? Classifying the Normative Content of the Protocol That Clears Real-Time Advertising*. Preprint, 2026. doi:10.13140/RG.2.2.27937.57448.

[16] Sekowski, A. *Measuring OpenRTB Dialects in Client-Side Header Bidding*. Preprint, 2026. doi:10.13140/RG.2.2.26572.78720.

[17] Shamieh, F., Abu Sharkh, M., and Hawilo, H. *Enhancing Ad Identification in the Fragmented CTV Advertising Ecosystem*. In Proc. IEEE CCECE, 2025.

[18] Springborn, K., and Barford, P. *Impression Fraud in On-line Advertising via Pay-Per-View Networks*. In Proc. USENIX Security, 2013.

[19] Stone-Gross, B., Stevens, R., Zarras, A., Kemmerer, R., Kruegel, C., and Vigna, G. *Understanding Fraudulent Activities in Online Ad Exchanges*. In Proc. ACM IMC, 2011. doi:10.1145/2068816.2068843.

[20] Tagliaro, C., Hahn, F., Sepe, R., Aceti, A., and Lindorfer, M. *I Still Know What You Watched Last Sunday: Privacy of the HbbTV Protocol in the European Smart TV Landscape*. In Proc. NDSS, 2023.

[21] Varmarken, J., Le, H., Shuba, A., Shafiq, Z., and Markopoulou, A. *The TV is Smart and Full of Trackers: Measuring Smart TV Advertising and Tracking*. PoPETs, 2020(2). arXiv:1911.03447.

[22] Varmarken, J., Al Aaraj, J., Trimananda, R., and Markopoulou, A. *FingerprinTV: Fingerprinting Smart TV Apps*. PoPETs, 2022(3).

[23] Vekaria, Y., Nithyanand, R., and Shafiq, Z. *The Inventory is Dark and Full of Misinformation: Understanding the Abuse of Ad Inventory Pooling in the Ad-Tech Supply Chain*. In Proc. IEEE Symposium on Security and Privacy, 2024. arXiv:2210.06654.

Specifications and industry standards.

[24] IAB Tech Lab. *Authorized Digital Sellers (ads.txt)*. v1.0 June 2017; v1.0.1 August 2017 (`SUBDOMAIN`); v1.0.2 March 2019; v1.0.3 March 2021 (`INVENTORYPARTNERDOMAIN`, introduced for CTV); v1.1 August 2022 (`OWNERDOMAIN`, `MANAGERDOMAIN`). Current version 1.1.

[25] IAB Tech Lab. *Authorized Sellers for Apps (app-ads.txt)*, v1.0, 13 March 2019. A separate specification from ads.txt, and the document that defines the store-listing developer-URL discovery procedure this paper measures in Section 7.3.

[26] IAB Tech Lab. *sellers.json*, v1.0, 2019, and the OpenRTB SupplyChain object. SupplyChain v1.1, announced 23 June 2026, adds technical-custody reporting covering SSAI platforms, ad servers, SDKs and wrappers; public comment open to 21 August 2026.

[27] IAB Tech Lab. *OpenRTB API Specification*, version 2.6. Object catalog release 202606 used throughout.

[28] IAB Tech Lab. *Digital Video Ad Serving Template (VAST)*. Version 4.0 (January 2016) introduced `AdVerifications` and `UniversalAdId`; version 4.1 (November 2018) defines the server-side ad insertion request headers `X-Device-IP`, `X-Device-User-Agent`, `X-Forwarded-For` and `X-Device-Referer`; 4.2 June 2019; 4.3 December 2022; 4.4 draft only. Note that IAB's own version index misdates 4.1 to August 2017.

[29] IAB Tech Lab. *Server-Side Ad Insertion (SSAI) VAST Macros*, v1.0, November 2020. Defines the `[SERVERSIDE]` macro, by which a receiver can distinguish server-fired from client-fired tracking.

[30] IAB Tech Lab. *VAST CTV Addendum*, draft, July 2024. Backports Universal Ad ID and Open Measurement support into VAST 2 and 3, which is the version range CTV platforms actually parse.

[31] IAB Tech Lab. *OTT/CTV Store Assigned App Identification Guidelines*, August 2020. The basis for using "store ID" rather than "bundle" on the CTV path.

[32] IAB Tech Lab. *Guidelines for Identifier for Advertising (IFA) on OTT Platforms*, with the `ifa_type` extension. Deprecated as of 2023; the successor is an unversioned community extension whose active values are `dpid`, `ppid`, `sspid` and `sessionid1`.

[33] IAB Tech Lab. *ads.cert 2.0*: Call Signs and Authenticated Connections, January 2022, and the deprecated ads.cert 1.0 signed bid requests (beta draft, November 2018, ECDSA P-256, never finalized).

[34] IAB Tech Lab. *Open Measurement SDK*. Version 1.5 covers tvOS, Android TV, Tizen and webOS; version 1.6 (October 2025) adds Device Attestation built on IETF Privacy Pass, with DoubleVerify, HUMAN, Google, IAS, Amazon Ads and AdsWizz among the announced adopters.

[35] Media Rating Council. *Invalid Traffic Detection and Filtration Standards Addendum*, with the GIVT and SIVT taxonomy. SIVT category 5 names server-side ad insertion spoofing; category 6 names domain and app misrepresentation including app ID spoofing. The 2024 interim update extends app misrepresentation to the CTV store ID and classifies non-CTV apps misrepresented as CTV as SIVT.

[36] Media Rating Council. *Server-side Ad Insertion and OTT Guidance*, August 2021. Requires client-initiated counting for accredited measurement, requires server-fired-only tracking to be segregated and disclaimed as non-compliant, and proposes SSAI provider certification on the stated rationale that invalid traffic perpetrators may disguise themselves as SSAI providers.

[37] Trustworthy Accountability Group. *Certified Against Fraud Guidelines*, v10.1, July 2025. Mandates 100% GIVT and SIVT filtration against an MRC-recognized standard, data-centre IP filtering, ads.txt and app-ads.txt support, ads.cert for SSAI billing notifications, and SSAI headers.

[38] Streaming Video Technology Alliance. *Server-Guided Ad Insertion*, standardized July 2025; taken up by IAB Tech Lab later in 2025. HLS Interstitials and DASH event mechanisms returning ad resolution and measurement to the client player.

[39] AdCP. *Ad Context Protocol* specification, version 3.1.1, repository state at commit `c68d54542`, 3 July 2026.

Industry fraud disclosures. These are the corpus sources. Each scheme's entry in `data/fraud-corpus.json` carries its own citation and a source grade, ten DoubleVerify primaries are preserved in `sources/dv-primaries/` with their document creation dates recorded in `sources/MANIFEST.md`, and Section 11 discusses their evidentiary limits at length. Discoverers are DoubleVerify Fraud Lab, HUMAN Satori (formerly White Ops), Pixalate, Oracle Moat and Method Media Intelligence.

[40] DoubleVerify. *Fraud Follows the Money: DV Is Rooting Out Fraud on CTV Apps*, 11 August 2020. The legitimacy-cloaking category, 1,300+ fraudulent CTV apps in 18 months, and the statement that "because of the way most CTV ad inventory is delivered (via SSAI), information about the placement is often self-declared."

[41] DoubleVerify. *Uncovering MultiTerra*, 15 September 2020. The document creation date settles the disclosure date against secondary coverage placing it in February 2022.

[42] DoubleVerify. *New CTV Fraud Scheme Dwarfs Previous Attacks* (ParrotTerra), 28 January 2021. Also carries the LeoTerra V1-V5 phase history, the Colorius attribution, and the canonical three-phase description of SSAI schemes.

[43] DoubleVerify. *Sophisticated SSAI Scheme Hijacks Real CTV Device Sessions* (SneakyTerra), 4 March 2021. Identification stated as October 2020.

[44] DoubleVerify. *DV Uncovers Long-Game of Fraud With OctoBot*, 20 April 2021. Confirms seven interconnected CTV schemes from November 2019 as one group, names LowerTerra as the first variant, and publishes the decrypted command-server payload pairing a per-request user agent with a matching TLS cipher configuration.

[45] DoubleVerify. *New CTV Scheme Uses Screensavers to Hijack Devices* (SmokeScreen), 11 August 2021.

[46] DoubleVerify. *ViperBot*, 24 March 2022. Verification stripping and redirection. Cited from secondary coverage of the DoubleVerify disclosure; the primary is not among the preserved documents.

[47] DoubleVerify. *Fraudsters Impersonate Smart Refrigerators While Falsifying CTV Traffic* (LeoTerra IoT variant), 24 August 2022.

[48] DoubleVerify. *BeatSting*, 2 February 2023. Headlined as the first large-scale ad impression fraud scheme in audio, which is why this paper treats it as an audio contrast case. Cited from secondary coverage.

[49] DoubleVerify. *New Fraud Scheme, CycloneBot, Hits CTV*, 14 December 2023. The document creation date settles the disclosure date against secondary coverage placing it in February 2024. Also states that Roku's Advertising Watermark validated and confirmed DV's behavioural findings rather than producing them.

[50] DoubleVerify. *ShadowBot*, June 2025. The 35 million spoofed devices figure is specifically mobile.

[51] Pixalate. *Uncovering DiCaprio*, January 2020, with a parallel investigation by Craig Silverman at BuzzFeed News.

[52] Pixalate. *Monarch OTT/CTV ad fraud scheme*, 25 March 2020. Pixalate's hedging language is preserved wherever this row is cited, because the accusations name companies and remain unadjudicated.

[53] White Ops (HUMAN). *ICEBUCKET* disclosure, 16 April 2020. 1.9 billion ad requests per day in January 2020; 2 million spoofed consumers across 30 countries; at peak "28 percent of all programmatic CTV traffic that it had visibility into", a qualifier that must travel with the figure.

[54] Oracle. *Oracle Exposes Largest CTV Ad Fraud Operation Ever* (StreamScam), 17 December 2020. Oracle has never confirmed DoubleVerify's identification of StreamScam with LeoTerra, and Oracle Moat no longer exists.

[55] HUMAN. *PARETO* botnet disruption, 22 April 2021, with the Human Collective (Omnicom Media Group, The Trade Desk, Magnite) and support from Google and Roku. Roughly one million infected Android devices running 29 Android apps, impersonating more than 6,000 CTV apps across Fire OS, tvOS and Roku OS, at an average of 650 million ad requests per day.

[56] Method Media Intelligence. *RapidFire*, 23 August 2021, identified late 2020. methodmi.com/ctv-ott-report-rapidfire, with coverage in AdExchanger, "MMI Says It Uncovered CTV Fraud Costing Advertisers $20 Million Per Month". A Python script generating counterfeit bid requests in JSON across multiple supply-side platforms; $10 million a month to the operators and $20 million a month in advertiser cost, both MMI's models. The operators are reported as a five-member team of former adtech professionals based overseas, running an ad network MMI dubbed HyperCast through a registered corporate entity in Nevada. MMI's separate claim that 50% of CTV real-time bidding requests are counterfeit was disputed at the time and is not used in this paper.

[57] HUMAN. *VASTFLUX* takedown, January 2023. Mobile in-app; retained as the VAST-layer contrast case, not as a CTV corpus member.

[58] HUMAN. *Trojans All the Way Down: BADBOX and PEACHPIT*, October 2023. 39 apps installed more than 15 million times, comprising 20 Android apps, 16 iOS apps and 3 CTV channels, each hard-coded to a fake supply-side platform. Hidden WebViews on infected devices spoof the app, the referring website and the model of phone, tablet or connected TV. Four billion fraudulent requests a day at peak, across 227 countries. No CTV bundle identifiers or channel names were published, which is why this paper grades the CTV component low-confidence.

[59] HUMAN. *HUMAN Security Disrupts CTV Device Spoofing Operation "NewsJunkie"*, 7 July 2026. A coordinated group of sellers transacting inventory masquerading as premium CTV content, at "hundreds of millions to nearly two billion invalid CTV bid requests per day, per individual seller". HUMAN's stated reason the environment is hard to protect is that a substantial share of CTV inventory is delivered with no JavaScript, so far fewer signals are available, and that no single signal revealed the fraud.

[60] CleanTap. Spoofed-device demonstration study, October 2025. Industry, not peer reviewed, and a manufactured demonstration rather than an observed operation.

[61] Pixalate. CTV app-ads.txt adoption measurements, 2020-2021: 80% of the top 500 Roku apps and 62% of the top 500 Fire TV apps. Corroborates the per-domain completion rate in Section 7.3 from a different sampling frame. Also the source for the transparent versus non-transparent SSAI distinction used in Sections 4 and 11.

Availability

Artifacts: https://github.com/aleksUIX/ctv-verification-gap

That repository holds the field inventory generated from the OpenRTB 2.6 object catalog with specification citations, the field codings, the fraud corpus with per-mapping grades and a rejection reason for every excluded candidate, the mitigation matrix with per-cell justifications under both readings, the VAST layer codings, the crawl code with frozen store-listing and compliance-file snapshots, a manifest of ten preserved primary disclosure documents (the PDFs themselves carry vendor copyright and are not redistributed), and every analysis script.

Each number in this paper is produced by a script in that repository: `compute_index.py` for Sections 5 and 5.3, `compute_corpus.py` for Section 6, `compute_matrix.py` for Sections 7.1 and 8, `reliability.py` for

Section 5.4, the `crawl/` scripts for Section 7.3, and `figures.py` for every figure. The counts in Sections 6 and 7 were hand-derived in an earlier draft and were wrong; they are now derived mechanically, and a reader who changes a coding in the published JSON will see every dependent figure in the paper move.

Conflict of interest

The author develops and maintains open-source validators for the protocols analyzed here (VASTlint, RTBlint). The OpenRTB field inventory used in this paper is extracted from RTBlint's object catalog. No funding was received for this work and no vendor reviewed it before publication.