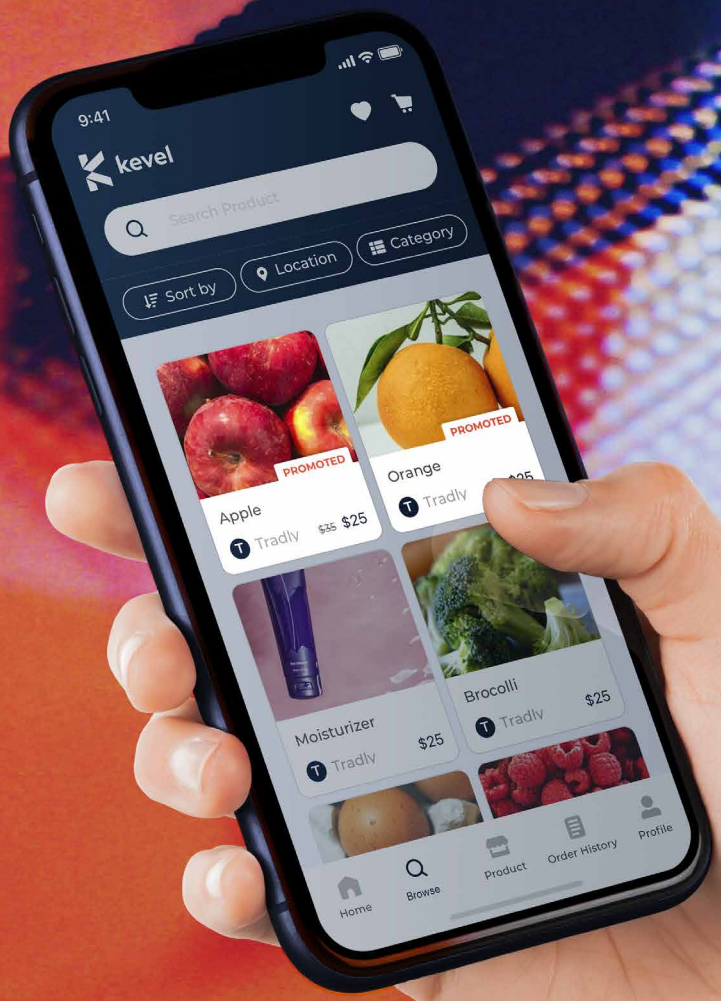
Guide

kevel

What is server-side ad serving?

The 2023 Definitive Guide



What's inside

- How server-side ad serving works
- Problems with client-side ad serving
- Examples of server-side ads
- Transitioning from client-side to server-side ad serving

Executive Summary

CURRENT LANDSCAPE

Many publishers rely on third-party client-side tags or bulky in-app SDKs to monetize their digital platforms and serve ads. These add-ons plague the internet with slow loading webpages, crashing apps, malware, obtrusive banner ads, and PII-leakage.

When you use client-side ad serving, not only are you missing out on untapped revenue, but you may also be setting your platform up for a range of issues down the road.

LOOKING FORWARD

Moving into 2023, the industry is evolving away from traditional client-side ad tags and the issues that come with it. Instead, companies are implementing server-side ad servers. Server-side ads integrate directly into your web application, making for a seamless ad experience that looks and feels like an extension of your platform's content.

SERVER-SIDE AD REQUESTS CAN:

- Monetize ad blocked users (40% of whom make up Internet-users in APAC, 36% in Europe, and 38% in North America).
- Control first-party data and mitigate the risk of data leakage.
- Ensure ads are compliant with international privacy laws (e.g. GDPR, CCPA and LGPD)
- Eliminate hidden third-party cookies.
- Aid with sustainability; faster load times make for more climate-friendly website hosting.
- Enable complete creative customization with unlimited sizes and formats to add value to advertisers and improve publishers' UX.
- Display attractive, high-value native ad units that users (and advertisers) actually enjoy.

This whitepaper explores all things server-side ad serving, to help you understand why it promises a brighter future for platform monetization.

Table of Contents

What is server-side ad serving?

Server-side ad serving allows publishers to request and place ads or internal promotions directly on their digital properties without the use of coded tags.

HERE'S HOW IT WORKS:

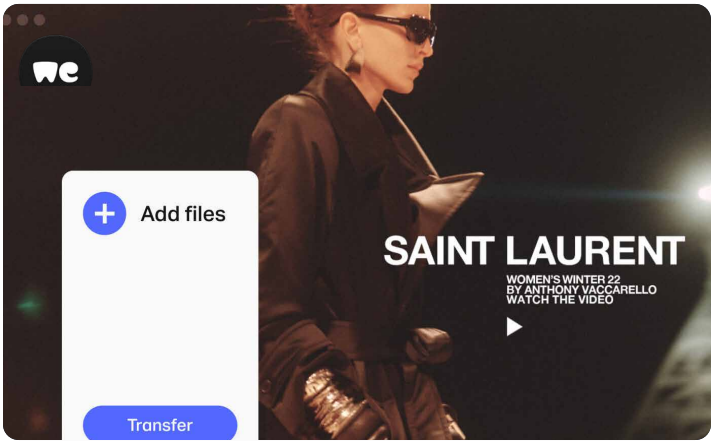
1. **A request is sent** to an internal or partner ad decision engine, like Kevel, which picks the winning ad to display
2. **The decision engine chooses a winning ad** based on various data sets. With Kevel, this includes keyword, context, first party-data, and relevancy scores approved by the publisher
3. **The winning ad's details** are returned in JSON
4. **Data in the response** is parsed
5. **Raw information is inserted** directly into your mobile or web app as it loads

Server-side ads are typically direct-sold or sold through programmatic channels, like PMPs and programmatic guaranteed. This strategy works best for brands who want to deliver an innovative advertising experience through seamless native ads, like sponsored listings, promoted posts, carousel ads, and more.

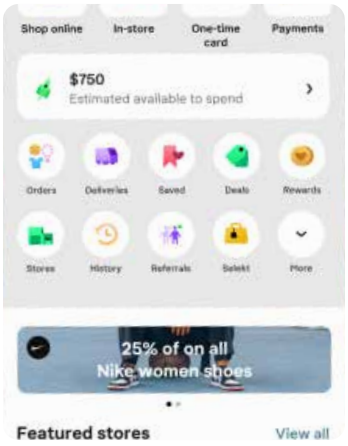
Native ads mean integrating ads seamlessly into your feed, homepage, search results, and more. Any existing content can be replaced with sponsored content. This limits ad obtrusiveness and creates ads that are stylistically identical to your standard content, save for "sponsored" or "promoted" identifiers.

Server-side doesn't necessarily mean no JavaScript, as it's likely your web app uses JavaScript elements to display ads. The key difference regards how ad decisions are requested: through client-side JavaScript ad tags or through server-side ad requests.

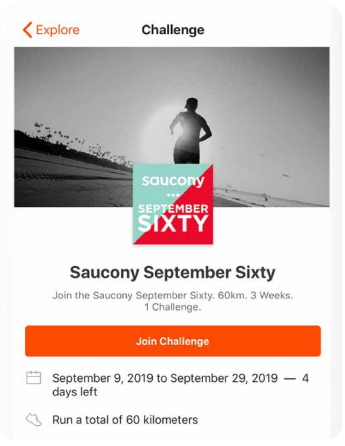
Examples of native ads enabled through server-side implementations include:



WeTransfer



Klarna



Strava

What is client-side ad serving?

Client-side ad serving through JavaScript ad tags or third-party mobile SDKs involves inserting ad code directly onto the platform. These tags ping the external ad tech vendor directly, where the winning ad is picked and then inserted into the site/app where the code is placed.

PROBLEMS WITH CLIENT-SIDE AD TAGS

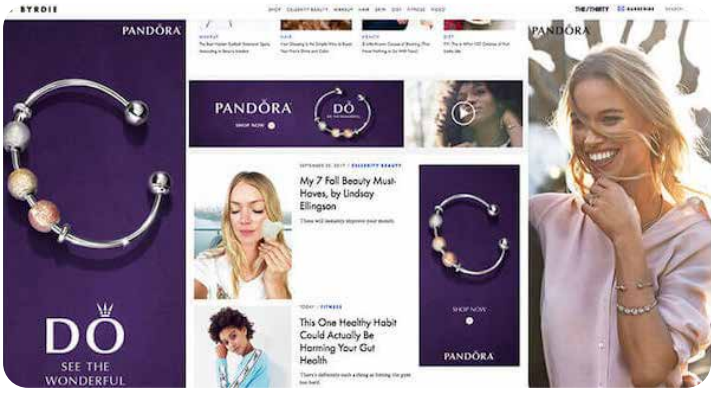
JavaScript client-side ad tags come with drawbacks for platform performance and security. You could encounter issues like:

Slow sites and apps

Ad tags are notoriously slow, hindering page load times by seconds. The number of ad slots and load method – including asynchronous and lazy loading – can cause even more lag. Slow load times lead to jumpy content, poor browsing experience, and user attrition.

No revenue from ad block users

Ad blocking tools can identify and block major ad tech tags. Even direct-sold ads and internal promotions will be blocked if served through a particular ad tag. With 36% of Europeans and 38% of North Americans using an ad blocking service, client-side advertising severely limits your ads' reach and value to advertisers.



Hidden Cookies

All of your platform's third-party script drops cookies, which can harvest and sell user data. This violates user trust, and is illegal under some international privacy laws (a UK city council learned this the hard way).

Not GDPR/CCPA compliant

Ad tags aren't inherently non-compliant; however, the tag decides what data to send the vendor. If the tag's code pulls user data or sends data to another partner without consent, you're on the hook for any incurring privacy law fines. If there's any code on your page or app that you don't exclusively own, you're always at risk for privacy law non-compliance.

What is client-side ad serving?

Malware

Your ad tech partners aren't immune to malware infiltration, which can drop auto-redirect ads or fake ad calls so you don't get the revenue. This costs publishers nearly $1 billion a year according to Fast Company. Even renowned publishers like The New York Times and The Atlantic aren't immune. Unless you own all of the code on your site, malware will always pose a threat.

Obtrusive ads that ruin user experience

Client-side ad tags exist to provide scalable revenue through streamlining the sale and purchase of standardized units. Because of this inflexibility, these ads are awkwardly placed, tend to "pop-up", and distract users' browsing experience. Though profitable, these ads are notorious for frustrating users and creating bad brand experiences.

Being at the whim of Google, Apple, and others

Google's Chrome is the latest browser to announce the sunsetting of third-party cookies, following Apple's announcement that Safari would do the same. While Google's third-party cookie offramp will be lengthy, we're moving towards a world where publishers can only rely on monetizing their first-party data. Companies that built their own ad platforms, like Facebook, Pinterest, and Amazon, don't rely on ad tags for revenue, and can rest easy without the negative effects of a third-party cookie depreciation. A tag-free ad platform also means avoiding hiccups that come with sudden changes like Google's recent Ad Manager outage.

Client-side vendor tags aren't the future of ad monetization – they're too susceptible to outside factors, while also neglecting user experience and privacy expectations. So, why do companies continue to use client-side Javascript ad tags?

WHY DO COMPANIES USE CLIENT-SIDE AD SERVING?

For smaller blogs, websites, and apps, it stems from necessity. Many businesses don't have the resources to build a direct-sold, server-side ad platform. Client-side tags are easy to insert and drive revenue almost immediately, making for an attractive, if imperfect, solution.

Yet larger companies with the capital to invest in more innovative ad monetization strategies oftentimes don't, and continue to resort to client-side ads out of convenience. But once the honeymoon period of ad revenue stagnates, these publishers are left with a platform that isn't efficient, user-friendly or convenient.

How does a server-side approach differ?

Server-side monetization has two key differences from a client-side strategy:

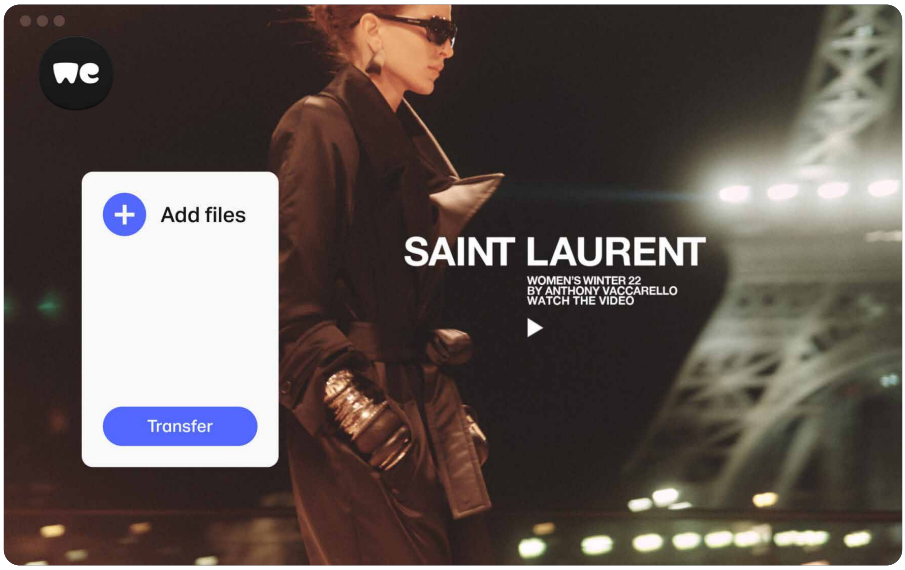
- Revenue is driven by directly-sold, PMP, or programmatic guaranteed ads, not third-party traffic
- The ad request process is server-side, eliminating on-site JavaScript ad tags that slow load times

Ad details are inserted directly into your content management system, with ads using the same CSS and layout as your organic content. Because of this, server-side ads are less intrusive, and enhance rather than detract from the user experience.

With server-side ad serving in place, some, or all, of third-party ad tags can be removed. This leads to faster page load time, privacy law compliance, ad block monetization, and happier users.

Potential server-side ad units include native ads, sponsored listings, content- and intent-based banner targeting, personalized internal promotions, digital-out-of-home, and more. Server-side ad serving also supports more innovative advertising strategies, like metaverse or other blended in-game ad units.

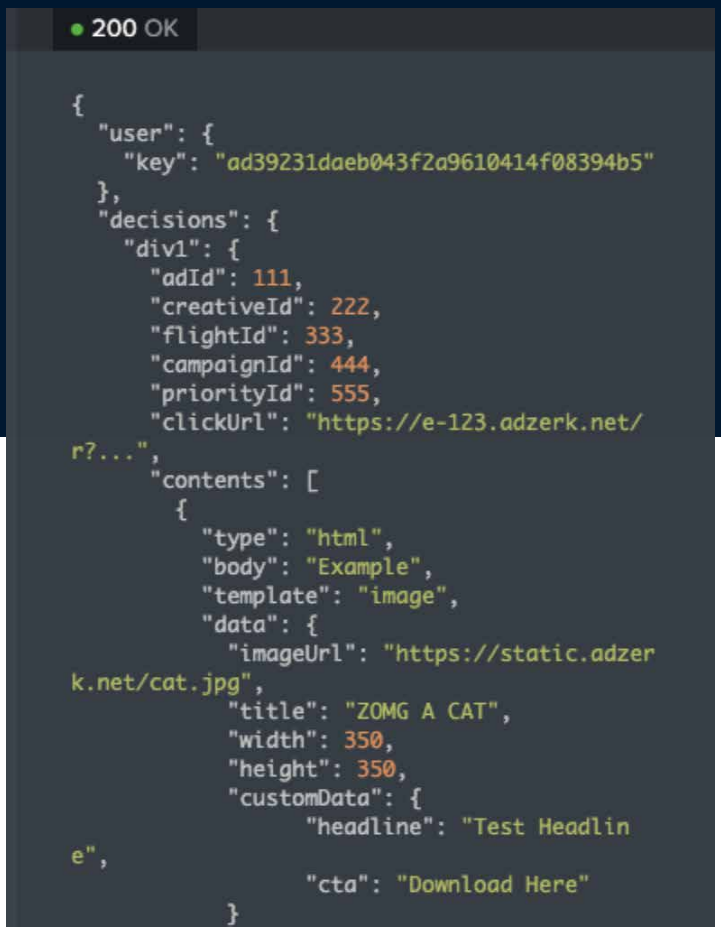
Moving towards a native ad strategy allows the removal of some (or all) of third-party ad tags - leading to faster pages, privacy law compliance, ad block monetization, and happier users.



Server-side in action

HERE'S WHAT HAPPENS AT THE TIME OF AN AD REQUEST:

- Your backend system sends an API call to the ad decision engine
- The engine picks the winning ad and sends back details about it in a JSON response (see below), including image URL, ad name, meta data, click URL, etc
- You then take the info and insert it into your Content Management System / web app / mobile app.



A server-side ad platform addresses many of the problems plaguing a client-side ad strategy.

With this set-up, the only data the vendor collects is information you choose to send, and the only content that appears on the site/app is what you choose to place.

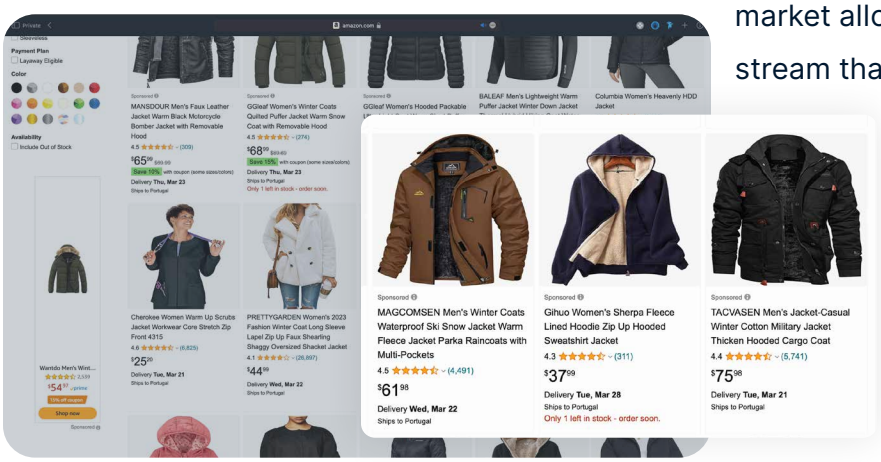
Server-side vs. Client-side ad platforms

WHAT	CLIENT-SIDE TAGS	SERVER-SIDE API REQUESTS
AD RESPONSE TIMES (impact page load speeds)	Slow (depending on set-up, could take seconds)	Fast (a recent Kevel client check found 40x faster GAM fetch times)
AD BLOCKING	Ads will be blocked automatically - no revenue from visitors using ad blockers	Since the ad is requested server-side without tags, it's tougher for ad blockers to identify and block your ads - enabling you to monetize 30%-40% more users.
HIDDEN THIRD-PARTY COOKIES	Can be dropped unbeknownst to you	Can't be dropped as the vendor doesn't have any code on your site
MALWARE	Can infect your device without your knowledge	Cannot infect your device
PRIVACY LAW COMPLIANCE (GDPR/CCPA/etc)	On page tags or mobile SDKs could lead to vendors collecting/using your user data illegally	The API calls include only the user data you want to send
OBTRUSIVENESS	Standard rectangular banner ads stand out and annoy users	Integrated native ads could have same CSS, style, etc of organic content
FULL CONTROL OVER REVENUE	Beholden to decisions you can't control - like Google introducing a new AdSense algorithm or web browsers auto-blocking third-party cookies	You can build your own walled garden where you make the rules

Challenges of switching to server-side

Whether you build a server-side platform yourself, or use infrastructure APIs like Kevel to do so, the technical requirements can be a hurdle. It isn't rocket science, especially as API services continue to grow in popularity, and fixes like combining campaign manager tools with server-side ad decisioning have gained traction. Even so, switching to server-side ad serving isn't a one man job.

Making the switch will need buy-in from multiple departments, including Revenue, Product, and Engineering. A server-side strategy will also require direct-sold placements procured by a sales team or a self-serve platform. Both of these options need additional resources from multiple teams across your business. For help building out this strategy, see our guide on needed roles for building a successful ad product team.



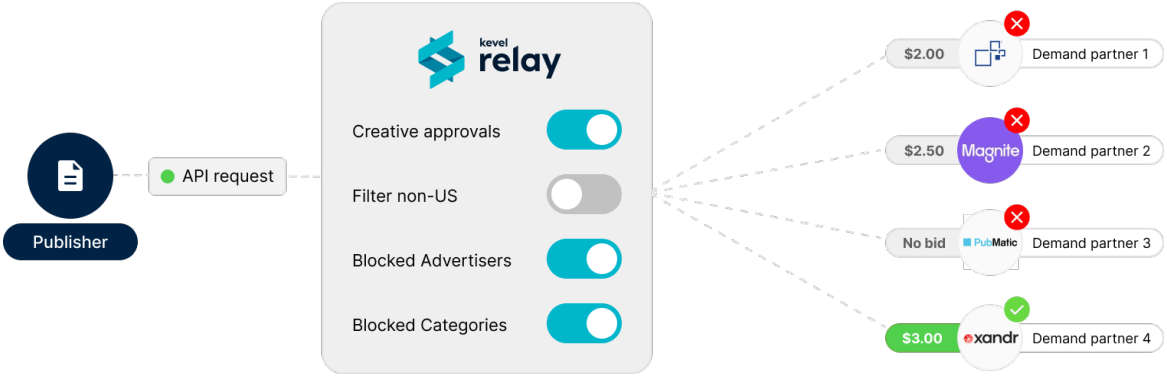
IN GOOD COMPANY

Many innovative server-side ad platforms exist in the market. Amazon and Facebook's server-side offerings, for example, are seeing substantial revenue growth.

These successes come as no surprise: both of these brands recognize the importance of great ad experiences, and invest to achieve them. While utilizing client-side ad tags would've been the easiest way to monetize, they weren't willing to play by Google or another ad tech player's rules. They pursued this vision early on as they were growing, not as an afterthought.

Amazon, whose Q4 2022 ad revenue is reported at $11.6 billion, utilizes its native sponsored products to provide capital which offsets potential losses from their competitive pricing and two-day shipping. Their ability to think innovatively and pursue an ad strategy different from 99% of the market allows them to tap into an auxiliary revenue stream that their competitors can't.

Transitioning from client-side to server-side ad serving



If you have no other monetization strategy in place, we're not asking you to immediately pitch your programmatic ad tags. There is a middle ground between the two options.

As a first step, migrate any direct-sold demand to server-side requesting while keeping your header bidding wrapper on the page. With your UX team, identify and convert standard organic content into sponsored content, and upsell these native placements at premium prices to your advertisers. These steps alone will ramp up your page load times, combat ad blocking, and increase your control over ad look and feel.

Once you've launched server-side ad serving with direct-sold, you can manage your existing programmatic placements with a server-side solution like Relay; Kevel's suite of programmatic APIs. Relay lets you mediate and manage your programmatic partners with a server-side setup that keeps you in control of your inventory. You can also consider scaling your direct offering with PMPs or Programmatic Guaranteed Ads with Relay.