

The Shift to Server-Side Analytics: Benefits, Considerations, and Comparison to Client-Side Approaches

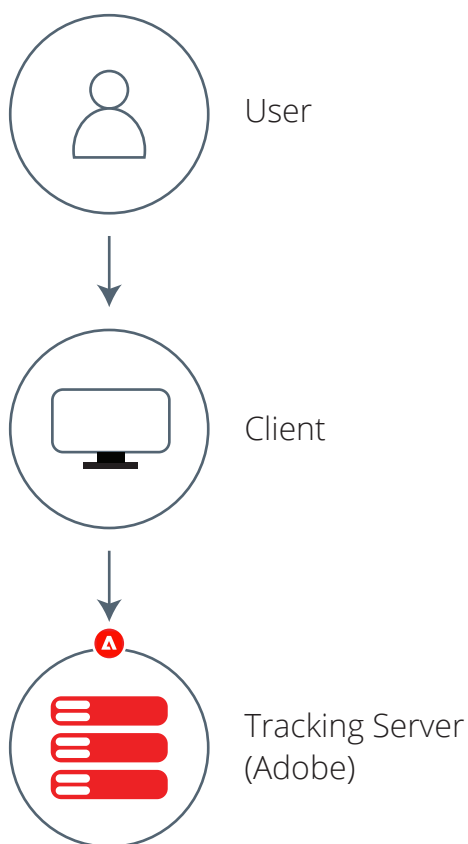


The concept of server-side implementations for MarTech tools is not new. Actually it has been in the spotlight for the past couple of years. But we at Digital Loop have been facing more and more client requests and discussions in the direction of server-side implementations of Adobe Analytics - or at least hybrid models - in the last months, so we decided to take a closer look at the current possibilities these ways of implementation are offering. The most important thing to mention here is that there are a lot of options within the server-side implementation and before deciding to go server-side, we recommend you to reflect on your current challenges and afterwards decide on which kind of setup is the right one for you. The goal of this Whitepaper is to provide you with most relevant aspects to make these above mentioned considerations and to be able to decide on a proper setup for your organization.

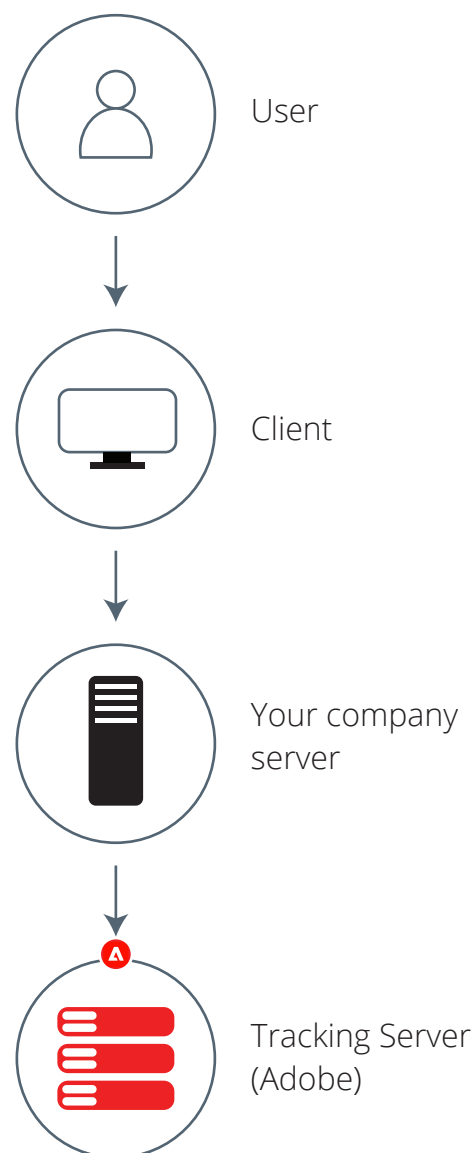
The concept of server-side tracking explained

First of all we would like to briefly explain to you the difference between client- and server-side tracking implementations.

In a client-to-server setup, your user's data is sent directly from the client to the vendor's - in our case Adobe's - server:



In a server-to-server setup, your company is providing a server, to which the data is sent to from the client. From your server, you're then sending the data to the Adobe server:



Sending the data from your own server has the advantage of gaining full control over the data sent to Adobe. You can add or remove data from the analytics payload.

Why companies are moving to server-side Adobe Analytics

Before deepdiving into the different aspects of an Adobe Analytics implementation, we would like to briefly write about the reasons that make companies consider switching from a client- to a server-side implementation.

#1

Impact of browser restrictions

With the release of Intelligent Tracking Prevention (ITP), Enhanced Tracking Prevention (ETP) and other browser features to restrict cookies from tracking users, companies were able to see a decrease in their traffic within Adobe Analytics. Also the rising popularity of ad blockers leads to more and more companies losing share of users that can be tracked. As most of these technologies are looking for tracking cookies in the user's browser, server-side implementations became a real alternative to the client-side integrations.

Most client-side integrations rely on the usage of Third-Party Cookies to ensure the user identification and basic functionalities. With the browser restrictions & ad blockers, blocking of third party cookies can lead to no user data being tracked.

Adobe came up with a solution for this by providing the CNAME setup which allows you to host the cookies in a first party context to avoid tracking prevention. This solves at least the problem with third party cookies, though ad blockers can also cause first-party cookies to be blocked.

And at this point, server-side implementations can solve this problem by not using the cookie-based Experience Cloud ID (ECID) for identity resolution but to use an identifier owned by the company to track the data or moving to a completely server-side ECID integration.

#2

Performance benefits: Reducing client-side load

A second reason why your company might consider switching to a server-side implementation is the positive impact on page performance. Moving your Analytics setup away from the client leads to less resources being loaded and therefore to higher pagespeed and performance. This not only has a positive impact on your general user experience but is a SEO ranking factor as well.

#3 Data quality

Third, a server-side integration affects your data quality and integrity in a positive way. Relying on JavaScript for executing your Analytics functionalities involves a lot of dependencies with the code on the rest of your webpage. Moving this code to your server allows you on the one hand to match it perfectly with your environment. On the other hand, this way of implementation allows you to fully control what is executed and in which way. This independence of external factors leads to a higher quality of your data due to less issues in your tracking.

#4 Data security

A fourth reason for you to migrate your Analytics implementation server-to-server is that it also causes a higher level of security for your customer's data. By not executing any data collection in the client, your customers' data is not accessible to third parties. And additionally the ways of server-side Analytics implementation we will present to you in this whitepaper will allow you to use an authentication for the APIs. Adobe offers a Server to Server authentication for all API endpoints within Adobe Experience Platform and Experience Cloud. This can be an important factor for companies in financial services or insurance working with sensitive customer data especially in logged-in areas.

#5 Data Enrichment

If you decide to send your website data to Adobe Analytics via your own servers, you are also free to add more data in the payload to enrich your view on the customer. You can for example collect a stock-keeping unit (SKU) in your payload from the client. On your server, you can then use this SKU to add additional product-related information like the price or product name before sending it to Adobe Analytics. Another use case could be to add customer-related information like loyalty status or Customer Lifetime Value (CLV). This data can be used for additional analysis purposes within Adobe Analytics.

Key Considerations for Adobe Analytics Server-Side Implementation

Now if one of the reasons above makes you consider to switch to a server-side analytics implementation, keep in mind that there are some important considerations to make, before deciding on a specific solution. First, with the introduction of Adobe WebSDK, there are now two options of implementation available which differ regarding code bases and data distribution. We want you to understand the differences to decide which way to go. The second big topic to think about is how you want to identify your users. The goal is to present you the different IDs available including the advantages and disadvantages in the context of a server-side integration.

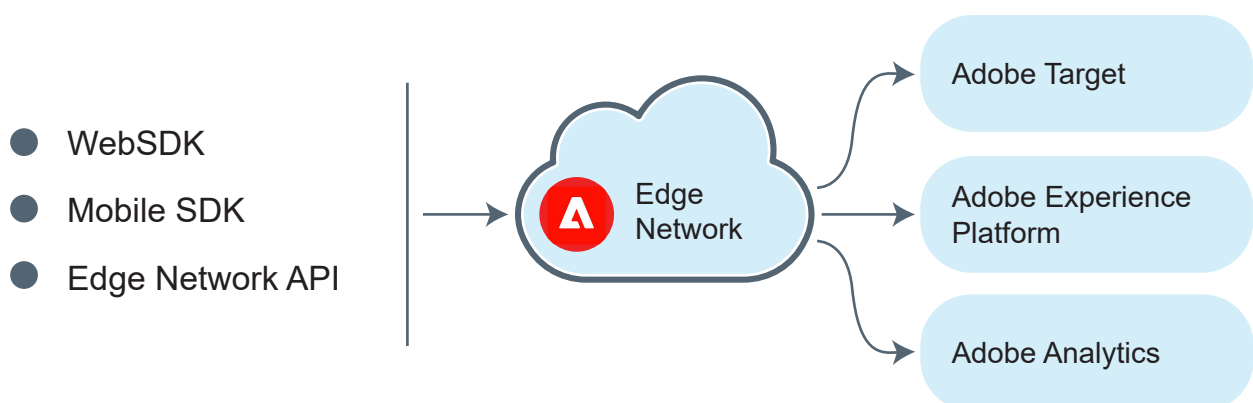
From AppMeasurement.js to Edge Network: The Evolution of Adobe Analytics Implementation Methods

One very fundamental decision to make before deciding on a server-side setup is the one between legacy “Analytics-only” implementation and the new one via the Adobe Experience Platform Edge Network. For most Analytics integrations, the general recommendation would most likely be to go with the Edge Network. Though there can be cases where the legacy integration still can be a valid option. The good news is: both options offer a server-side implementation. But: there are restrictions, especially when it comes to identity usage. But before deepdiving into this topic, we would like to first focus on the difference between legacy and Edge Network integration.

Legacy methods include the AppMeasurement.js and the Bulk Data Insertion API. The legacy implementation works very straightforward: data is collected - either client- or server-side and sent to Adobe Analytics.



Using the Adobe Experience Platform Edge Network on the other hand is a bit more complex, but it also offers you more flexibility and scalability. The Edge Network includes the Adobe Experience Platform WebSDK, Mobile SDK and the Edge Network Server API. The interesting thing about the Edge Network is that there is only one payload for all tools integrated on your website via the Edge Network (Adobe Analytics, Adobe Target, Adobe Experience Platform). So data is being collected (client- or server-side) and then distributed to the different tools from the Edge Network:



If you already have plans to also add other Adobe tools like Adobe Target or the AEP to your integration, it makes sense to use the Edge Network implementation. As mentioned above, both options provide an API for server-side data collection. Before deciding between the two, there is another important thing to consider: how are you going to handle the identity resolution within your implementation?

Identity Resolution: ECID vs. First-Party IDs

In Adobe Analytics users are per default identified with the Experience Cloud ID (ECID). The ECID is stored in a cookie within the user's browser. It is collected with other context data and sent to Adobe Analytics and other applications from the Adobe Experience Cloud, as Adobe Target, or used as an identifier in Adobe Experience Platform (AEP). The ECID is stored in a first-party context to ensure third-party cookie blocking does not affect your ID-setup. Adobe additionally offers a CNAME implementation to store the cookie on a custom domain that matches your company's site domain. Unfortunately even this CNAME implementation underlies browser restrictions which leads to a shorter cookie lifetime. Having a shorter cookie lifetime for your ECID cookie leads to one single user (or more specifically: one single browser) being tracked as multiple Unique Visitors in Adobe Analytics, as the ECID is being renewed after the cookie's expiry. In order to provide a more stable way of identifying your users, Adobe introduced the usage of First Party Device IDs (FPIDs) within the Adobe Web SDK. These are identifiers generated by your own web servers and stored in a first-party cookie in the client's browser. This sounds very similar to the ECID cookie. The big difference here is that a server-side cookie is immune to web browser restrictions which makes the identifier more persistent and your tracking more accurate. After the FPID is sent to the Edge Network through the WebSDK, it is used to seed an ECID. Then the ECID is used to distribute data to Adobe Analytics and other Adobe Tools. If you want to have a persistent solution for identifying your visitors within Adobe Analytics, the First Party Device ID is a safe way to go. It is possible to either collect the FPID via Adobe Web SDK and go with a partial server-side integration (at least when it comes to the cookies), or to send it server-side within the Identity Map for a full server-side approach.

There is in fact a third option for identity resolution which can be used in order to be independent of the client and cookies: the server-side ECID. Adobe provides the possibility to send server-to-server calls to their Data Collection Servers (DCS) to directly receive the ECID from here. Briefly summarized, you manually follow the steps for ECID creation by first requesting a Unique User ID (UUID) from the DCS, and, within a second request, use this UUID to get an ECID created. We want to highlight three important things here: first, this solution requires you to build your own table to store the IDs and connected information as there is no cookie available. Second, this solution includes Adobe Audience Manager (AAM) functionality. If not licensed yet, it might require you to check this with Adobe regarding additional cost. And third, if you have any other solution from the Adobe Experience Cloud, like for example Adobe Target or Adobe Audience Manager, running client-side, this server-side ECID creation will lead to the fact that you're not able to match your users between client- and sever-side solutions. As your server will not be able to read an already existing UUID or ECID from the browser, your server-side setup will always generate a new ECID, which is different from the cookie-based one.

What degree of server-side works for your needs?

By now you should have noticed that there are a lot of options to decide on a final implementation. We would like to summarize at this point the most important things to think about:



- What are my requirements for an Analytics Solution (Data Privacy, Data Integrity, Page Performance)?
- Which data and areas on my website do I want to track? Is there any sensitive user data included?
- What are my requirements towards user identification? Do I need to stitch Analytics data with data from other tools (Adobe Target, Adobe Experience Platform, Adobe Audience Manager)? Do I require cross-domain-tracking?
- When it comes to implementation and maintenance, how many resources do I have?
- Are in general other tools from the Adobe Experience Cloud part of my MarTech architecture? Are there tools implemented client- or server-side? Is it necessary to have a shared data collection via Adobe Edge Network?

After answering those questions for you and your company, you will soon be aware that everyway of implementation might include sacrifices in another area. For example: going fully server-side (including a server-side ECID setup) might cause huge efforts for the initial setup or migration and the following maintenance as well. It also leads to implications for the integration of other Adobe Experience Cloud Applications within your Tech Stack. Defining the final solution will for sure be a compromise. Is it more important to collect the sensible data from your logged-in area which requires a server-side data collection? Or do you want to have a flexible and easy maintainable setup with a client-side approach without collecting data in your account areas?

Below you will find a comparison of all implementation methods (client- as well as server-side) including its pros and cons. What we would like to highlight here again is that these methods are not complete. There are various ways to combine those methods for hybrid integrations if needed, like for example using a client-side approach on the first pageview of a visitor and then proceeding with client-side data collection. We might tend to say that whatever fits your needs is possible - from fully-fledged server-side to a hybrid approach.

Conclusion & Future Outlook

The most important takeaway we want you to provide is that there is not 'the' server-side implementation of Analytics. There are many shades of grey when it comes to how much server-side there is and from my experience all of those options come with compromise. With this Whitepaper we wanted to give some more input and clarity about the possibilities as well as advantages and disadvantages of the different approaches. However, transitioning to a server-side setup requires strategic planning and technical expertise - internal or external. Additionally this most often comes with more effort for setup and maintenance than the client-side integrations. Looking at how privacy regulations have impacted the Digital Analytics landscape in the last years, we can definitely expect more to come. Therefore it will become more and more important to consider shifting away from a client-side tracking setup to ensure data availability. Also we are very sure that vendors will come up with more solutions that guarantee a privacy conform tracking in the future. Either way, we will keep an eye on what the MarTech landscape provides and will provide new updates and support to stay ahead of the changes.

	 Client-Side				 Server-Side	
	Manually load AppMeasurement.js on webpages	AppMeasurement.js via Tag Manager	Manually load WebSDK on webpages	WebSDK via Tag Manager	Edge Network Server API	Bulk Data Insertion API
Ease of Implementation	✗ More Manual Setup	✓ Easier Implementation with Tag Rules	✗ More Manual Setup	✓ Easier Implementation with Tag Rules	✗ Complex Server Setup	✗ Complex Server Setup
Data Security and Integrity					✓ Allows full control of the data that is being sent	✓ Allows full control of the data that is being sent
Performance	✗ Can cause slow page load	✓ Optimized through TMS	✗ Can cause slow page load	✓ Optimized through TMS	✓ Faster as it removes client dependency	✓ Faster as it removes client dependency
Ad Blocker Resistance	✗ Can be blocked	✗ Can be blocked	✗ Can be blocked	✗ Can be blocked	✓ Harder to block	✓ Harder to block
Maintenance & Ongoing Effort	✗ High effort for updates	✓ Easier via TMS	✗ Requires ongoing maintenance	✓ Optimized through TMS	✗ Requires backend maintenance, cost for server capacities (cloud, on-prem), harder to debug	✗ Requires backend maintenance, cost for server capacities (cloud, on-prem), harder to debug
Multi-Device-Tracking	✗ Challenging without identity resolution	✗ Challenging without identity resolution	✓ Can be resolved via Identity Graph in AEP	✓ Can be resolved via Identity Graph in AEP	✓ Can be resolved via Identity Graph in AEP	✗ Challenging without identity resolution
First Party ID Usage Possible	✗	✗	✓	✓	✗	✗
XDM Schema Creation in AEP necessary	✗	✗	✓	✓	✓	✗
Use Cases	Legacy Implementations	Legacy Implementations, Customization	Future-proof solution, AEP as an option, direct control over tracking-code	Future-proof solution, AEP as an option, easy deployment & maintenance via TMS	High-security environments, data governance priority, page performance optimization, IoT device implementations	High-security environments, data governance priority, page performance optimization, Analytics Stand-alone implementation without XDM formatting