Google Tag Manager, the new anti-adblock weapon

The "Server-Side Tagging" version of the Google tool allows you to bypass browser and other adblocker protections

Posted by Pixel de Tracking on Nov 15, 2020

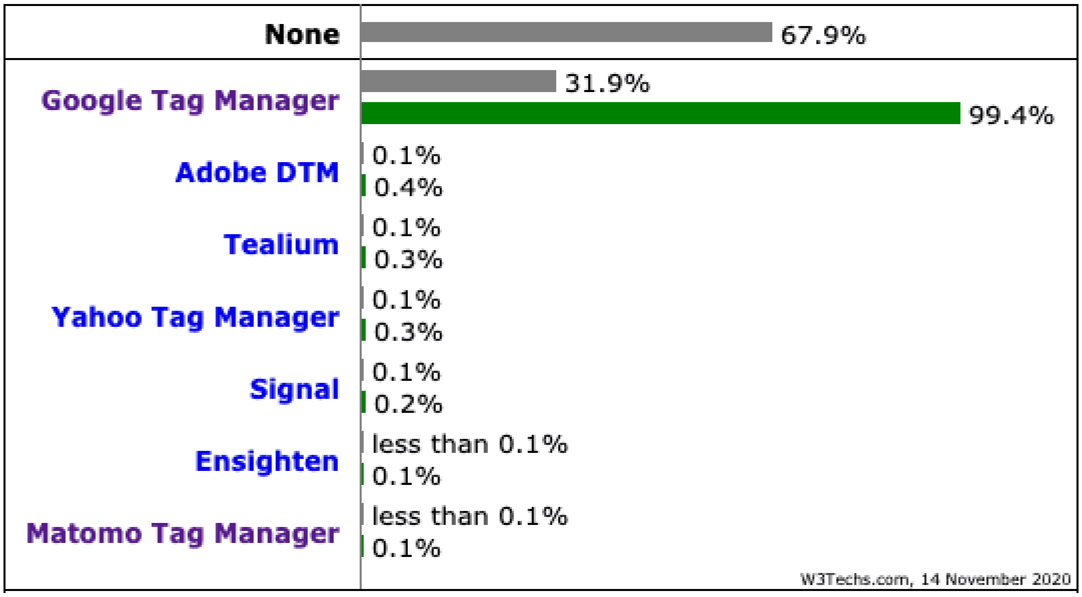
Google Tag Manager, the Trojan horse for marketing teams

Google Tag Manager (https://marketingplatform.google.com/intl/fr/about/tag-manager/) is a TMS (https://fr.wikipedia.org/wiki/Système_de_gestion_de_balises) (Tag Management System): it allows marketing teams to add trackers to a website or application, without having to go through developers. Via a web interface, these teams can decide:

- Tracers to be triggered (analytics, A / B testing, attribution, etc.).
- Trigger conditions (categories of pages, user characteristics, etc.).
- Data to be transmitted to these third-party tools (user characteristics, navigation data, variables present on the page, etc.).

It is not the only one (we can for example quote Segment (https://segment.com) , the French TagCommander (https://www.commandersact.com/fr/solutions/tagcommander/) or

Matomo Tag Manager (https://fr.matomo.org/docs/tag-manager/)) but Google Tag Manager is ultra dominant:

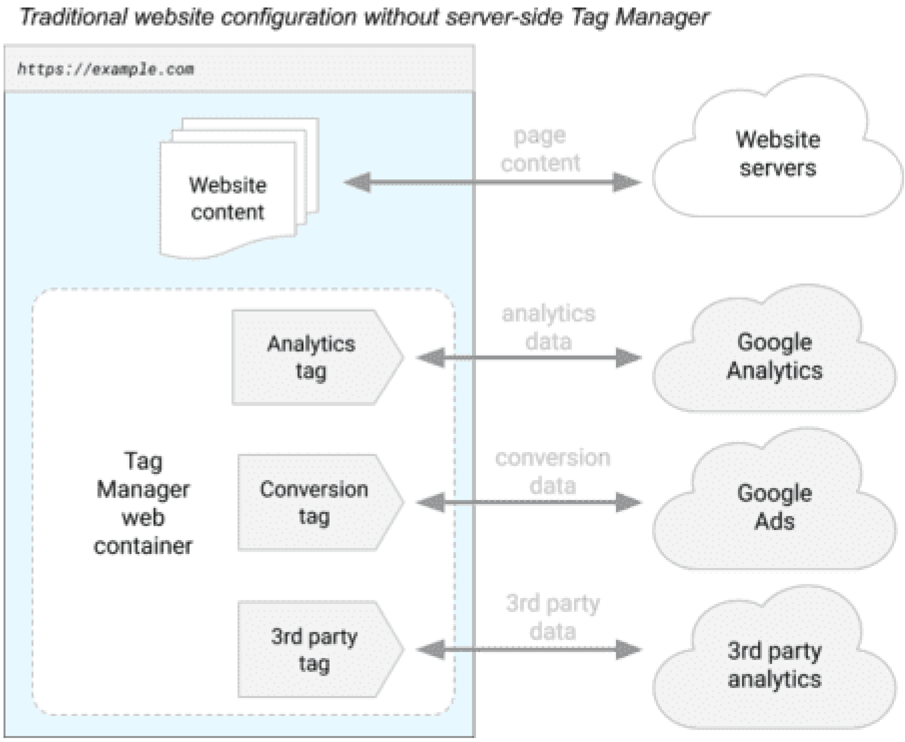


Google Tag Manager is present on 31.9% of the top 10 million Alexa websites, according to W3Techs (https://frama.link/6sE8rTVq) , but above all Google Tag Manager has a 99.4% market share on TMS (!)

How has Google been able to impose itself again? As with Google Analytics, the standard version of Google Tag Manager is free (market solutions are generally paid), it is very well integrated with other Google solutions and it is well done.

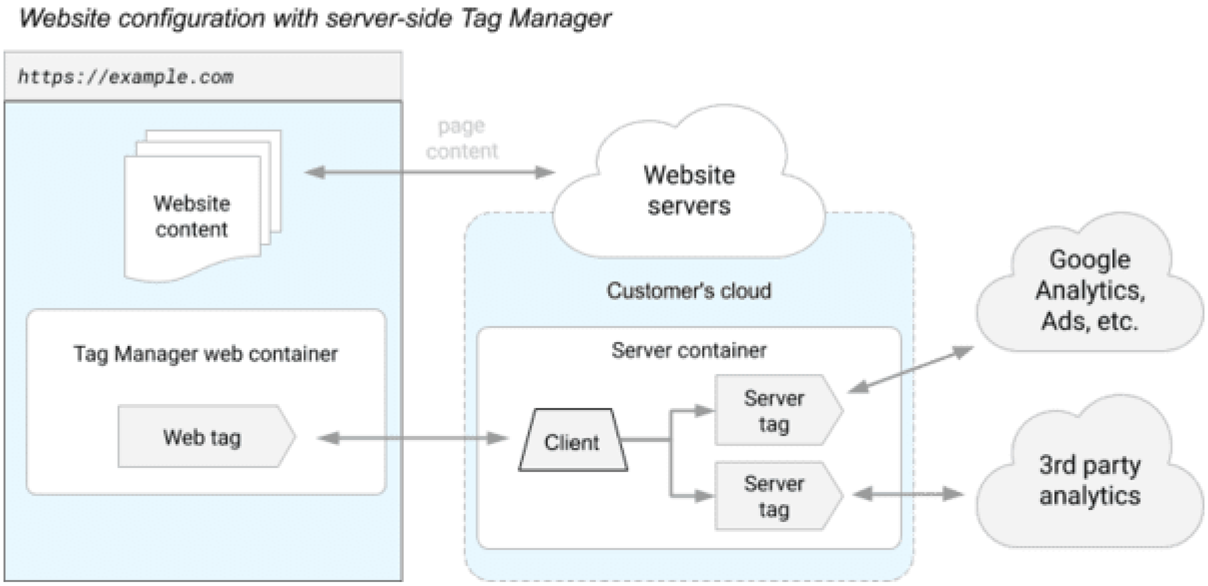
Trackers that are no longer called from your browser

Last August, Google announced a new version of Google Tag Manager (https://blog.google/products/marketingplatform/360/improve-performance-and-security-server-side-tagging/) , called Server-Side Tagging. Here is a diagram from Google (https://developers.google.com/tag-manager/serverside/intro) to explain how Google Tag Manager works in Client-Side Tagging version (the "historical" version):



Google Tag Manager will allow the triggering of various third-party tracers (on the diagram: Google Analytics, Google Ads, *and an analytics tool*), *directly on your browser.*

In the new Server-Side version (https://developers.google.com/tag-manager/serverside/intro) , third-party trackers are no longer run from your browser but from a " Proxy (https://fr.wikipedia.org/wiki/Proxy) " server called "Server container" on the diagram below (and hosted by Google):



The javascript library (called "Tag Manager web container" in the diagram) is always run on your browser in order to collect your interactions and your personal data, but the execution of the various third-party tracers takes place on the server side.

Note that this new version also applies to applications and to "offline" data collection (to transmit in-store purchases for example):

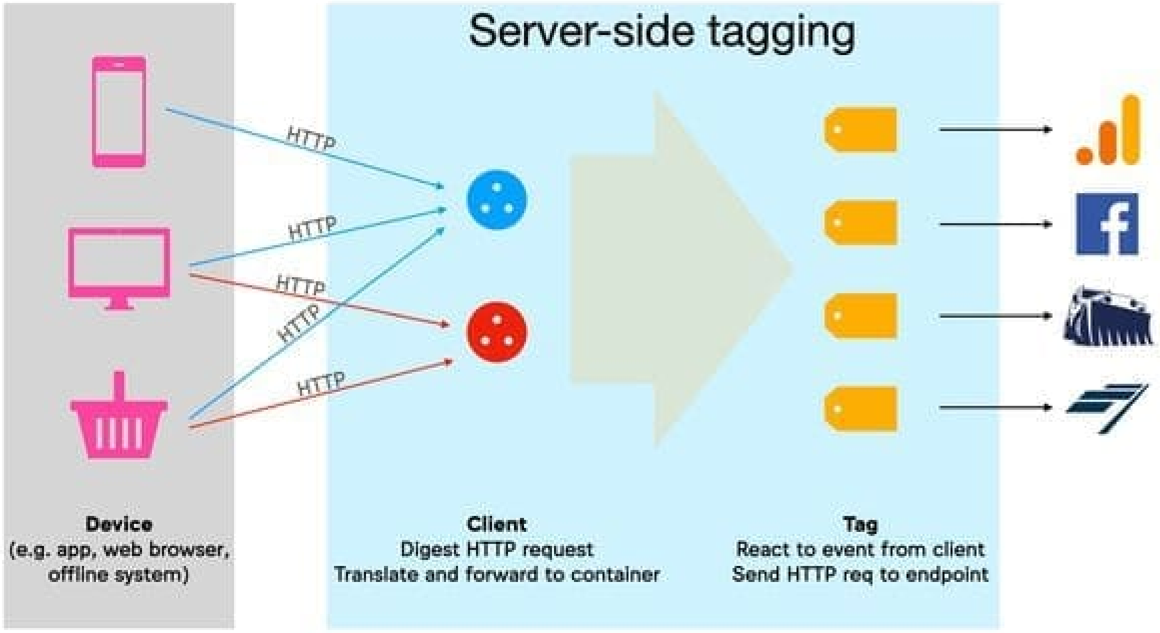

Server-side tagging
HTTP
HTTP
HTTP
HTTP
HTTP
Device
(e.g. app, web browser, offline system)
Client
Digest HTTP request
Translate and forward to container
Tag
React to event from client
Send HTTP req to endpoint

Diagram of Simo Ahava's blog (https://frama.link/CgwLQt6F) : on the server side, the "Clients" are there to translate the HTTP requests received into "events", the "Tags" react to these events to send "hits" to third-party marketing companies.

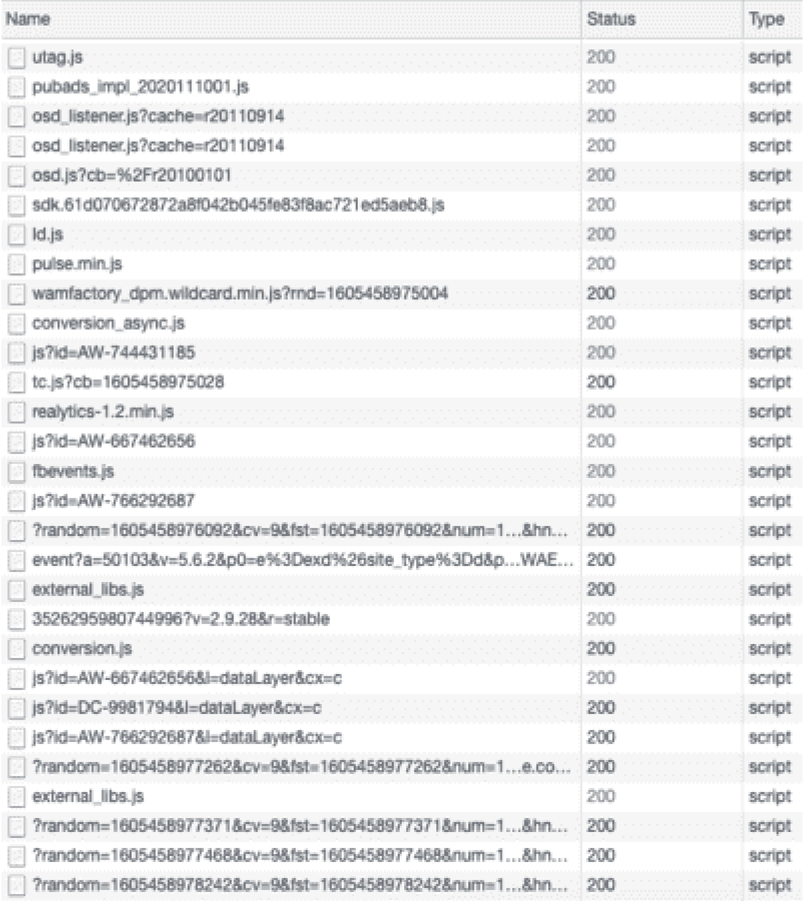
This logic of triggering third-party tracers on the server side is a game-changer. Simo Ahava has detailed the different impacts in an excellent article (https://www.simoahava.com/analytics/server-side-tagging-google-tag-manager/) , for my part I will summarize the advantages and focus on the problems for your privacy (operating on the server side can allow you to bypass your choices and leak your personal data, without being unmasked).

Better user experience

On most websites, the number of javascript libraries loaded by third parties (for analytics, advertising, A / B testing, etc.) is impressive. Loading and running these libraries is often the main cause of a bad user experience: site slowness and lack of interactivity.

Consequences for websites offering a bad user experience: less satisfied Internet users, who will directly abandon browsing or will not return.

Here is an example with Le Bon Coin, it calls an innumerable number of javascript libraries (https://www.pixeldetracking.com/fr/le-bon-coin-donnees-personnelles-rgpd) :



Name	Status	Type
utag.js	200	script
pubads_impl_2020111001.js	200	script
osd_listener.js?cache=r20110914	200	script
osd_listener.js?cache=r20110914	200	script
osd.js?cb=%2Fr20100101	200	script
sdk.61d070672872a8f042b045fe83f8ac721ed5aeb8.js	200	script
ld.js	200	script
pulse.min.js	200	script
wamfactory_dpm.wildcard.min.js?rnd=1605458975004	200	script
conversion_async.js	200	script
js?id=AW-744431185	200	script
tc.js?cb=1605458975028	200	script
realytics-1.2.min.js	200	script
js?id=AW-667462656	200	script
fbevents.js	200	script
js?id=AW-766292687	200	script
?random=1605458976092&cv=9&fst=1605458976092&num=1…&hn…	200	script
event?a=50103&v=5.6.2&p0=e%3Dexd%26site_type%3Dd&p…WAE…	200	script
external_libs.js	200	script
3526295980744996?v=2.9.28&r=stable	200	script
conversion.js	200	script
js?id=AW-667462656&l=dataLayer&cx=c	200	script
js?id=DC-9981794&l=dataLayer&cx=c	200	script
js?id=AW-766292687&l=dataLayer&cx=c	200	script
?random=1605458977262&cv=9&fst=1605458977262&num=1…e.co…	200	script
external_libs.js	200	script
?random=1605458977371&cv=9&fst=1605458977371&num=1…&hn…	200	script
?random=1605458977468&cv=9&fst=1605458977468&num=1…&hn…	200	script
?random=1605458978242&cv=9&fst=1605458978242&num=1…&hn…	200	script

A small part of the javascript scripts called on the home page of Le Bon Coin, this one leaks your personal data to many third parties (https://frama.link/RSFS9ZYh) .

In the best case scenario, the website will only install one javascript library (events can be very different between tools that do not have the same purposes, the website will sometimes use more than a single library). This could be that of Google Tag Manager but not necessarily: it is possible to create your own library or to use other libraries on the market such as Snowplow (https://docs.snowplowanalytics.com/docs/collecting-data/collecting-from-own-applications/javascript-tracker/web-quick-start-guide/) , Matomo (https://developer.matomo.org/guides/tracking-javascript-guide) , AT Internet, etc.

Then instructs this library to send the "hits" with the parameters required during key interactions. Then the "client" of the server container will have to translate these "hits" into events, these will be read by the "Tags" which will send "hits" to the third party marketing companies. Note that if the javascript library installed on the site is provided by Google, the "client" is already pre-configured in Google Tag Manager. If the website uses another library, it will have to create its own "client" in Google Tag Manager (example with AT Internet (https://levelup.gitconnected.com/google-tag-manager-server-side-how-to-manage-custom-vendor-tags-21bef51bc89e)), while waiting to have "clients" pre-configured for the main javascript tracking libraries.

Advantage therefore: a single javascript tracking library is installed on the website and a single "flow" of data from the browser, the user should see the difference.

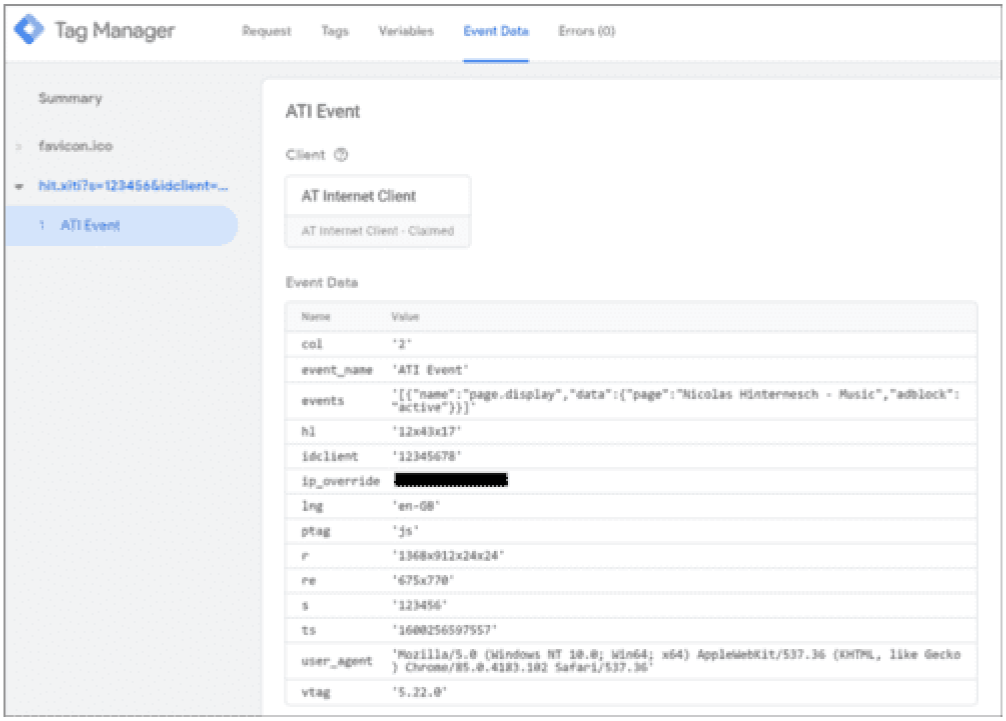
Better control over data transmitted to third parties

Having a "proxy" on the server side makes it possible to control the data transmitted to third parties (which is much more difficult when the trackers are directly executed by the user's browser):

- By default and unlike the "client-side" version, the IP address and User-Agent (https://fr.wikipedia.org/wiki/User_agent) (browser name, version, operating system, language, etc.) of the user do not leak (which avoids user identification via " fingerprinting (https://www.cnil.fr/fr/definition/fingerprinting) "). The publisher using the Server-Side Tagging version of Google Tag Manager may decide to transmit this information to third parties, but this is not automatic.
- It often happens that personal information is leaked to third parties via URL parameters (read for example the article " Google Tag Manager

Server-Side - How To Manage Custom Vendor Tags (https://medium.com/@thezedwards/the-2020-url-querystring-data-leaks-millions-of-user-emails-leaking-from-popular-websites-to-39a09d2303d2) "), Server-Side Tagging makes it possible to avoid that.

- In general, the publisher has control over the personal data and cookies sent by its "proxy" to third parties (read Google's technical documentation (https://developers.google.com/tag-manager/serverside/permissions) , note for example the get_cookies and set_cookies methods). It can therefore "clean" the information and send to third parties only what is strictly necessary.



ATI Event

Client

AT Internet Client

AT Internet Client - Claimed

Event Data

Name	Value
col	'2'
event_name	'ATI Event'
events	'[{"name":"page.display","data":{"page":"Nicolas Hinternesch - Music","adblock":"active"}}]'
hl	'12x43x17'
idclient	'12345678'
ip_override	[redacted]
lng	'en-GB'
ptag	'js'
r	'1368x912x24x24'
re	'675x770'
s	'123456'
ts	'1600256597557'
user_agent	'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/85.0.4183.102 Safari/537.36'
vtag	'5.22.0'

Example with an AT Internet hit "seen" by the "proxy" server, the website may decide not to transmit the user's IP address and User-Agent to AT Internet.

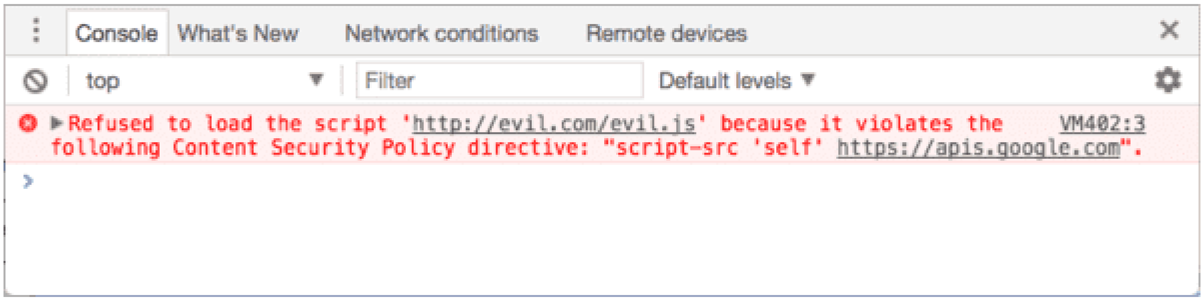
A better secure website

Setting up a Content-Security Policy (https://developer.mozilla.org/fr/docs/Web/HTTP/CSP) (CSP) allows a publisher to better protect itself against different types of threats including XSS (https://developer.mozilla.org/fr/docs/Glossaire/Cross-site_scripting) (Cross-Site Scripting) attacks and content injections. By adding a header to responses from the web server, the site can indicate to browsers which resources (scripts, css, etc.) are allowed.

Here is an example of CSP documented by Google (https://developers.google.com/web/fundamentals/security/csp) :

> *Content-Security-Policy: script-src 'self' https://apis.google.com.*

This means: the browser is only allowed to execute scripts that come directly from the site consulted (' *self* ') or from *apis.google.com* . And here's how your browser will react if a malicious script tries to run from the visited site:



```
Refused to load the script 'http://evil.com/evil.js' because it violates the     VM402:3
following Content Security Policy directive: "script-src 'self' https://apis.google.com".
```

The evil.js script is not hosted on the visited site, nor on apis.google.com: its execution is blocked by the browser.

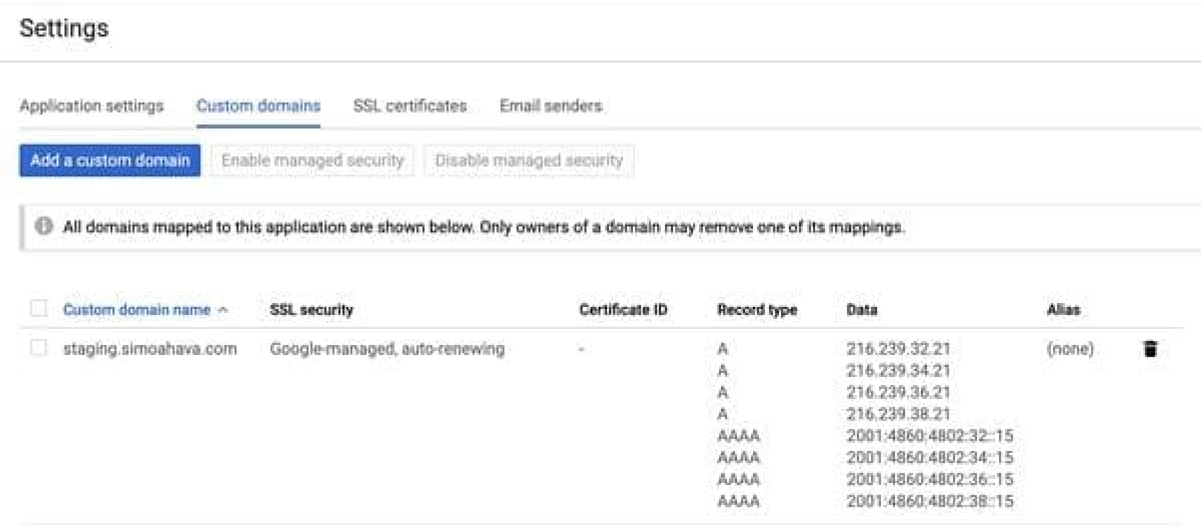
By greatly reducing the third-party domains allowed to execute javascript code, the CSP becomes more robust.

While Server-Side Tagging has advantages for users who consent to marketing surveillance (speed, security), it jeopardizes the protections of non-consenting users.

A bypass of browser protections

The "proxy" server is hosted in the Google cloud (App Engine (https://cloud.google.com/appengine?hl=fr) instance) but Google advises (https://developers.google.com/tag-manager/serverside/custom-domain) to link the App Engine domain to a subdomain of its customers' site (without explaining the reasons):

> *The default server-side tagging deployment is hosted on an App Engine domain. We recommend that you modify the deployment to use a subdomain of your website instead.*



Settings

All domains mapped to this application are shown below. Only owners of a domain may remove one of its mappings.

□	Custom domain name ^	SSL security	Certificate ID	Record type	Data	Alias
□	staging.simoahava.com	Google-managed, auto-renewing	-	A A A A AAAA AAAA AAAA AAAA	216.239.32.21 216.239.34.21 216.239.36.21 216.239.38.21 2001:4860:4802:32::15 2001:4860:4802:34::15 2001:4860:4802:36::15 2001:4860:4802:38::15	(none)

The link between the App Engine domain and the customer's subdomain, documented by Google (https://frama.link/DKtyuDee) .

Google does not recommend a CNAME type DNS record (alias), but a type A or AAAA type DNS record (https://fr.wikipedia.org/wiki/Liste_des_enregistrements_DNS) , directly linked to the IP addresses of Google App Engine, which acts as a host. The "proxy" server is therefore well considered by browsers as the 1st party, and the consequences are therefore important.

In particular, the cookies deposited by the "proxy" server are not third-party cookies, nor cookies created via javascript, nor cookies deposited by a CNAME domain. They are therefore authorized, without restriction:

- Safari via Intelligent Tracking Prevention (https://webkit.org/blog/9521/intelligent-tracking-prevention-2-3/) (ITP) restricts the lifespan of cookies created in javascript to 7 days (example: 1st-party cookies created by Google Analytics). Thanks to the "proxy" server, third-party plotters now overcome this limitation.
- Always Safari via ITP now restricts cookies placed via a CNAME domain to 7 days (https://webkit.org/blog/11338/cname-cloaking-and-bounce-tracking-defense/) . Thanks to the "proxy" server, third-party tracers are not affected by this limitation.
- Brave for its part blocks CNAME requests to known tracers (https://brave.com/privacy-updates-6/) . Again thanks to the "proxy" server, third-party tracers avoid this blocking.

A bypass of adblockers

Your adblocker (uBlock Origin on Firefox (https://addons.mozilla.org/fr/firefox/addon/ublock-origin/) for example), your content blocker (Firefox Focus (https://apps.apple.com/fr/app/firefox-focus/id1055677337) or Adguard (https://apps.apple.com/us/app/adguard-adblock-privacy/id1047223162) on iOS for example) or your DNS blocker (NextDNS (https://www.pixeldetracking.com/fr/nextdns-mon-nouveau-bloqueur-de-publicites-prefere) for example) works on your device. It can thus detect third-party trackers and block them before your personal data leaks.

None of this with the Server-Side Tagging version of Google Tag Manager: personal data leaks take place from the client's proxy server (hosted in the Google cloud) to third parties. You no longer have the hand to avoid these leaks.

You could say to yourself: just block the first call, that of your browser to the javascript library in charge of collecting the data and communicating to the "proxy" server. Except that this javascript library can very well be accessible on the domain of the website (and not on a Google domain for example). Also, Google already advises (https://developers.google.com/tag-manager/serverside/send-data#update_the_gtagjs_source_domain) its customers to change their *gtag.js* scripts *in* order to *enter* the domain of the proxy server. This manipulation already makes the blocking via domain name inoperative.

If *gtag.js* is a javascript script whose name is known to the main adblockers, they will have difficulty functioning when the name of the javascript library has been changed or when sites have created their own libraries.

uBlock Origin, effective against CNAME cloaking on Firefox (https://frama.link/VR9XsofE) , powerless against Server-Side Tagging?

How can adblockers react? The subject is not obvious, here are some ideas but I'm not sure they are feasible:

- Automatically detect these "1st party" calls to the "proxy" server via the URL parameters sent. Except that these URL parameters will change from one site to another, depending on the library used, the page viewed, etc.
- Detect the javascript library responsible for calls to the "proxy" server to block its execution. Except that you should not simply detect the javascript library provided by Google, but potentially all the javascript tracking libraries, even home libraries.
- Block the IP addresses of these proxy servers. Except that it will be necessary to manually find the thousands of IP addresses behind these "proxy" servers, to update them ... Or to decide to block all the IP addresses of Google App Engine (https://cloud.google.com/appengine/kb) , at the risk of blocking many applications. having nothing to do with tracking. Not to mention that Google could decide to open the "proxy" server to other hosts.
- Never run javascript on your browser, for example with the NoScript extension (https://addons.mozilla.org/fr/firefox/addon/noscript/) , drastically configured. Effective option, except that many sites will no longer work.

Escape your personal data in the most total opacity

While many websites today leak your personal data, often without your consent, it is still possible to audit the sites, prove the consent violation and document the leaks. The CNIL could, for example, do its job and sanction faults. None of this with Server-Side Tagging, a site can now very easily:

- Give an appearance of consent by letting you respond to a consent banner.

- While leaking your personal data to multiple third parties, without an external auditor being able to realize it (it will simply see the call "1st-party" to the server "proxy", without knowing if the personal data is used , shared or sold behind).

Your data in the Google cloud

By default, the "proxy" server logs all the requests it receives (https://developers.google.com/tag-manager/serverside/script-user-guide) :

> *By default, App Engine logs information about every single request (eg request path, query parameters, etc) that it receives.*

But the personal data contained in these queries is not the only information leaking to Google. As with CNAME cloaking (https://www.laquadrature.net/2020/10/05/le-deguisement-des-trackers-par-cname/) , cookies associated with the domain of the site consulted are sent to the subdomain of the "proxy" server. So, if your session cookies are associated with the site domain (and not a separate subdomain), they will be sent to Google's cloud.

This declares (https://cloud.google.com/security/privacy) that the data hosted on its cloud belongs to the customer, and not to Google. You still have to trust Google.

Server-Side Tagging, probably soon to be widely adopted

If Server-Side solutions have existed on the market for a long time, and if it was already possible to develop your own "proxy", the launch of the Google solution will probably have a huge impact on the adoption of Server-Side Tagging :

- Google Tag Manager is present on a considerable number of websites, it is ultra-dominant.
- Google presents this version as an evolution (https://blog.google/products/marketingplatform/360/improve-performance-and-security-server-side-tagging/) of TMS tools, improving the performance and security of websites.

Even if a Google Tag Manager client can continue to use the Client-Side version, even if the Server-Side version still has limits (few third-party libraries, some solutions will have difficulty being supported, etc.), even if the learning the solution is complex and even if it does pay off (yes, you have to pay the Google App Engine bill for the "proxy" server), we can therefore bet that Google Tag Manager clients will gradually migrate to this version.

Bypass adblockers and other browser protections, a selling point

As we have seen, Google does not explain (https://developers.google.com/tag-manager/serverside/custom-domain) the reason for creating a subdomain of the website for its "proxy" server:

> *The default server-side tagging deployment is hosted on an App Engine domain. We recommend that you modify the deployment to use a subdomain of your website instead.*

It doesn't need it, browser and adblocker protection bypasses have already been listed as "benefits" by many publications:

- Simo Ahava's " Server-side Tagging In Google Tag Manager (https://www.simoahava.com/privacy/intelligent-tracking-prevention-ios-14-ipados-14-safari-14/) ", the article indicates the benefit of being able to bypass Safari's limitations regarding the lifespan of javascript cookies. To his credit, the author does not want to give details on the fact that Server-Side Tagging makes it possible to bypass adblockers and indicates that data collection must be done after obtaining consent.
- “ GTM Server Side - The Natural Evolution for Your Tagging? (https://converteo.com/blog/gtm-server-side-levolution-naturelle-pour-votre-tagging/) ” From Converteo. The article lists the advantages of being able to bypass browser limitations such as those of Safari and Firefox, as well as bypassing adblockers.
- " Introduction to Google Tag Manager Server-side Tagging (https://www.analyticsmania.com/post/introduction-to-google-tag-manager-server-side-tagging/) ", from the Analytics mania blog. Here too, the browsers and adblockers limitations workarounds are listed as a benefit.
- " Google introduces server-side tagging, good news? (https://www.journaldunet.com/ebusiness/publicite/1494723-google-introduit-le-tagging-cote-serveur-une-bonne-nouvelle/) " By Nicolas Jaimes on the JDN. The angle of the article is advertising, and therefore the bypassing of browser protections is listed as a benefit (although for the moment, the lack of third-party libraries means that Server-Side Tagging remains complex to implement).

Unfortunately, it's a safe bet that many sites will also be drawn to these "benefits", in addition to the performance, security and control gains. The inability to audit websites will also be a big loss for privacy advocates. We hope that browsers and adblockers find solutions so that Internet users concerned about their privacy can continue to defend themselves.

● (https://twitter.com/pixeldetracking)