

GDPR by proxy, and who is responsible?

2023-01-10 [en] Joaquim Homrighausen, gdprtech@webbplatsen.se

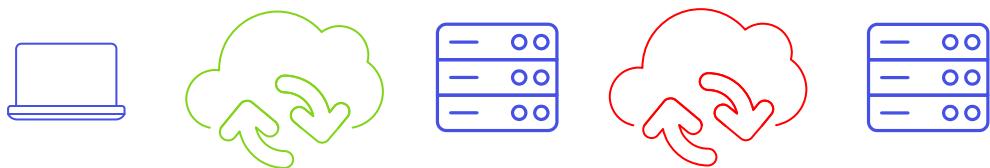
With GDPR receiving increased focus and attention, it is vital to keep track of where data is processed, how data is processed, and where data is stored, if said data can be affected by GDPR.

Many websites and cloud services are in direct violation of GDPR. This is, probably, not intentional in many cases, but some use a proxy mechanism to hide or mask where and how data from and to the visitor/user is actually processed. This is nearly impossible to detect, unless you have a direct and clear insight into the architecture of said website or Internet service. Using external resources on a website or cloud service is another way to get into GDPR trouble.

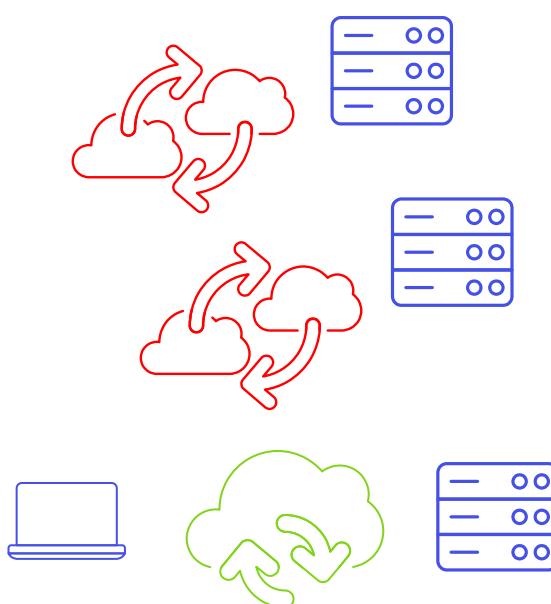
I often analyze websites, for clients, and because I'm curious by nature. I am amazed at the number of GDPR and privacy issues that could be resolved by some very minor changes to these websites, but yet they keep on leaking personal or other sensitive information year after year.

This will most likely be too technical for many, but I'll try to keep it as simplistic as possible. Feel free to ask me, and, naturally, feel free to point out errors. Feedback is much appreciated!

This article does not intend to discuss cookies, hidden pixels, and other ways of tracking users such as "fingerprinting". The article furthermore makes the assumption that the client is governed, and protected, by the laws of the European Union (EU).



Accessing cloud services within the EU (green), that then perform a data exchange outside the EU (red). The inside-EU cloud service acts as a proxy.



Accessing a cloud service within the EU (green), that then instructs the user's web browser to fetch external resources from outside-EU servers (red).

There are two different scenarios here:

1. **Backend websites proxied by a frontend:** A website that doesn't contain any external references, but instead passes data back and forth to an underlying "backend", including data from the visitor/user.
2. **External references:** An actual website containing external references to things like CSS, images, JavaScript frameworks or snippets, and/or web fonts will request the user's web browser to fetch these resources from the given URL. One common situation is where developers use a so called CDN, or *Content Distribution Network*.

Backend websites proxied by a frontend

This is a setup that is more common than you might think, and it's extremely hard, not to say impossible, to detect.

From a technical standpoint, it may make sense to built services like this, and it's not necessarily the wrong way to go about things. The first server may be a web server, the second server a database server, or the first server is a web server and the second server an API server with which the web server exchanges data.

So what's the problem? Well, if the first server is inside the EU, and the second server is outside of the EU, there may be a transfer of personal data to a third country. That's the problem.

External references

Website and cloud service developers often use external frameworks and other types of pre-packaged collections of code, images, CSS, and web fonts, for what they're building. Not because it's needed, nor because makes that much sense in 2023, but because, well, they think it's a great idea when developing websites. It's not. At least not when such frameworks and other resources are retrieved from external infrastructure.

All of this can be hosted "locally" on the website, or the cloud service, without much difference to the user. It may, technically, increase the load on the local hosting infrastructure, and that is probably one of the few valid points among the barrage of excuses and explanations web developers will give you.

Arguments like "it will increase the time it takes to render/display the website", and "it becomes a pain to maintain over time" are not, in most cases, true. Nor is the argument for using a CDN that sounds something like "it will be cached in the client already"; that argument is based on the assumption that the client will have visited another website with the exact same external reference to the exact same resource. While this can happen, the probability is far lower than developers claim.

Every time the client's web browser goes to fetch such a resource via an external reference, it will pass the client's IP address (among other things) to the service hosting the external resource. There's very little the client can do about it, short of blocking the loading of said resources. Blocking the loading of said resources will more often than not completely break the website and/or service.

(continued on the next page)

What does "inside the EU" really mean?

You will often see cloud service providers claim that their services are “hosted inside the EU”, or that their services are served from “GDPR safe locations”. This, in itself, is not instrumental to whether or not a service can be considered “GDPR compliant” by nature.

What matters more is where the company that owns the infrastructure is incorporated.

For example, a US corporation, providing services from an Amsterdam (Netherlands) location, still falls under the jurisdiction of the USA. This leaves room for a potential transfer of personal information to a “third country”, should the US authorities request data from that US corporation.

Who's responsible?

You, as a business owner, and/or corporate executive, will ultimately be responsible for GDPR incidents and violations by your company. You cannot outsource or blame “the web people” or the “website developer”.

It may be a good idea to ask before you commit. And it really isn't that hard to stay on the right side of GDPR, true story.

Feel free to get in touch with me if you have questions, suggestions, or other feedback. My contact details are underneath the document title.

Credits: The illustrations use graphics from the Streamline collection, www.streamlinehq.com