

Results for **gettr.com**[↻ Check again](#)

🕒 2022-01-06 17:26:13 Etc/UTC

HTTPS by default:	✓ Yes
Content Security Policy:	✗ Not implemented
Referrer Policy:	✗ Referrers leaked
Cookies:	4 (2 first-party; 2 third-party)
Third-party requests:	18 requests to 9 unique hosts
IP address:	18.66.97.11 Look up

Checked URL: <http://gettr.com>

Final URL:

<https://gettr.com/onboarding>

✓ HTTPS by default

[gettr.com](#) uses HTTPS by default.

Chromium reports the following:

State	Title	Summary	Description
✓	Certificate	valid and trusted	The connection to this site is using a valid, trusted server certificate issued by Amazon.
✓	Connection	secure connection settings	The connection to this site is encrypted and authenticated using TLS 1.3, X25519, and AES_128_GCM.
✓	Resources	all served securely	All resources on this page are served securely.

More information about the site's TLS/SSL configuration:

- [Analyze gettr.com on SSL Labs](#)
- [Observatory by Mozilla](#)
- [Mozilla TLS Observatory](#)
- [testssl.sh](#)

HTTPS encrypts nearly all information sent between a client and a web service. Properly configured, it guarantees three things:

- **Confidentiality.** The visitor's connection is encrypted, obscuring URLs, cookies, and other sensitive metadata.
- **Authenticity.** The visitor is talking to the "real" website, and not to an impersonator or through a "man-in-the-middle".
- **Integrity.** The data sent between the visitor and the website has not been tampered with or modified.

A plain HTTP connection can be easily monitored, modified, and impersonated. Every unencrypted HTTP request reveals information about a user's behavior, and the interception and tracking of unencrypted browsing has become commonplace.

The goal of the Internet community is to establish encryption as the norm, and to phase out unencrypted connections.

🔗 GDPR: [Rec. 83](#), [Art. 5.1.f](#), [Art. 25](#), [Art. 32.1](#)

By GDPR [Art. 25](#), a controller is responsible for implementing state of the art data protection by design and by default. Encrypted connections are a well-established technology to protect the privacy of web visitors against eavesdroppers on the wire.

[🕒 How to implement](#)

✗ Strict Transport Security

HTTP Strict Transport Security (HSTS) not implemented.

[🕒 How to implement](#)

[HTTP Strict Transport Security](#) (HSTS) is a simple and [widely supported](#) standard to protect visitors by ensuring that their browsers always connect to a website over HTTPS. HSTS exists to remove the need for the common, insecure practice of redirecting users from `http://` to `https://` URLs.

When a browser knows that a domain has enabled HSTS, it does two things:

- Always uses an `https://` connection, even when clicking on an `http://` link or after typing a domain into the location bar without specifying a protocol.
- Removes the ability for users to click through warnings about invalid certificates.

A domain instructs browsers that it has enabled HSTS by returning an HTTP header over an HTTPS connection.

✗ Content Security Policy

Content Security Policy set in HTTP header:

Content Security Policy (CSP) header not implemented.

[🕒 How to implement](#)

The Content Security Policy tests are based on the ones from the [Mozilla HTTP Observatory](#) scanner/grader project ([Mozilla Public License 2.0](#)) by April King, reimplemented by us for Webbkoll. The explanatory texts are from the [Observatory by Mozilla](#) website, [CC-BY-SA 3.0](#). Any mistake or inaccuracy in the results is our fault.

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including Cross Site Scripting (XSS) and data injection attacks. These attacks are used for everything from data theft to site defacement to distribution of malware.

A primary goal of CSP is to mitigate and report XSS attacks. XSS attacks exploit the browser's trust of the content received from the server. Malicious scripts are executed by the victim's browser because the browser trusts the source of the content, even when it's not coming from where it seems to be coming from.

CSP makes it possible for server administrators to reduce or eliminate the vectors by which XSS can occur by specifying the domains that the browser should consider to be valid sources of executable scripts. A CSP compatible browser will then only execute scripts loaded in source files received from those whitelisted domains, ignoring all other script (including inline scripts and event-handling HTML attributes).

— MDN: [Content Security Policy \(CSP\)](#), Mozilla Contributors, [CC BY-SA 2.5](#)

🔗 GDPR: [Rec. 83](#), [Art. 5.1.f](#), [Art. 25](#), [Art. 32.2](#)

GDPR [Art. 32.2](#) makes clear that measures should be taken against unauthorized disclosure of, or access to, personal data transmitted, stored or otherwise processed. CSP is a relatively simple way of ensuring that your web visitors do not end up being put in contact with someone that either they - or you - did not anticipate for them to contact.

✅ Reporting (CSP, Certificate Transparency, Network Error Logging)

Reports are not sent to a third-party.

The Content Security Policy (CSP) directive [report-uri](#) , or [report-to](#) in combination with a `Report-To` header, instructs the user's browser to send a [violation report](#) to specified URI(s) if the CSP is violated. Each report is a JSON object containing information about the violation, including, among other things, the URL of the document where it occurred, and referrer information. While reporting is useful for developers to find and fix bugs, it can also be used for tracking purposes.

The [Expect-CT](#) header can be used to enforce [Certificate Transparency](#) requirements, and/or optionally send reports of Certificate Transparency violations to a specified URI.

The [NEL](#) (Network Error Logging) header instructs the user's browser to send reports about network errors (e.g. DNS resolution error, TCP or TLS connection failure, 4xx or 5xx HTTP responses) to a specified URI. It can also be configured to send reports about successful network requests. See [\[1\]](#), [\[2\]](#) for privacy considerations.

❌ Referrer Policy

Referrer Policy not set. This means that the default value `no-referrer-when-downgrade` , leaking referrers in many situations, is used.

👉 [How to implement](#)

When you click on a link, your browser will typically send the HTTP referer [sic] header to the webserver where the destination webpage is at. The header contains the full URL of the page you came from. This lets sites see where traffic comes from. The header is also sent when external resources (such as images, fonts, JS and CSS) are loaded.

The referrer header is privacy nightmare as it allows websites and services to track you across the web and learn about your browsing habits (and thus possibly private, sensitive information), particularly when combined with cookies.

By setting a Referrer Policy, it's possible for websites to tell browsers to not leak referrers. Referrer Policy lets you specify a policy that's applied to all links clicked, as well as all other requests generated by the page (images, JS, etc.).

🔑 GDPR: [Rec. 83](#), [Art. 5.1.c](#), [Art. 25](#), [Art. 32.2](#)

Setting referrer policy is an easy way to do data minimization ([Art. 5.1.c](#)) and help ensure that you don't transfer or disclose personal data needlessly ([Art. 32.2](#)).

✗ Subresource Integrity (SRI)

Subresource Integrity (SRI) not implemented, but all external resources are loaded over HTTPS

The following third-party resources are not loaded using SRI:

Type URL

script	https://static.zdassets.com/ekr/snippet.js?key=3d0ea9b2-3214-4304-9f69-a64536198d21
script	https://www.googletagmanager.com/gtag/js?id=AW-10782555182
script	https://websdk.appsflyer.com?st=pba&
script	https://connect.facebook.net/en_US/fbevents.js
script	https://connect.facebook.net/signals/config/1271570993296195?v=2.9.48&r=stable

🕒 How to implement

The Subresource Integrity test is based on the one from the [Mozilla HTTP Observatory](#) scanner/grader project ([Mozilla Public License 2.0](#)) by April King, reimplemented by us for Webbkoll.

HTTP headers

Pass	Header	Value	Result
✗	X-Content-Type-Options		X-Content-Type-Options header not implemented

Subresource Integrity (SRI) is a security feature that enables browsers to verify that resources they fetch (for example, from a CDN) are delivered without unexpected manipulation. It works by allowing you to provide a cryptographic hash that a fetched resource must match.

Using Content Delivery Networks (CDNs) to host files such as scripts and stylesheets that are shared among multiple sites can improve site performance and conserve bandwidth. However, using CDNs also comes with a risk, in that if an attacker gains control of a CDN, the attacker can inject arbitrary malicious content into files on the CDN (or replace the files completely) and thus can also potentially attack all sites that fetch files from that CDN.

Subresource Integrity enables you to mitigate some risks of attacks such as this, by ensuring that the files your web application or web document fetches (from a CDN or anywhere) have been delivered without a third-party having injected any additional content into those files — and without any other changes of any kind at all having been made to those files.

— MDN, [Subresource Integrity](#), Mozilla Contributors, [CC BY-SA 2.5](#)

GDPR: [Rec. 83](#), [Art. 5.1.f](#), [Art. 25](#), [Art. 32.2](#)

This is an easy measure to take against unauthorized disclosure of, or access to, personal data transmitted, stored or otherwise processed.

The **X-Content-Type-Options** response HTTP header is a marker used by the server to indicate that the MIME types advertised in the Content-Type headers should not be changed and be followed. This allows to opt-out of MIME type

Pass	Header	Value	Result
❗	X-Frame-Options		X-Frame-Options (XFO) header not implemented
❗	X-XSS-Protection		X-XSS-Protection header not implemented

🔗 How to implement

The header tests are based on the ones from the [Mozilla HTTP Observatory](#) scanner/grader project ([Mozilla Public License 2.0](#)) by April King, reimplemented by us for Webb koll. The explanatory texts are from the [Observatory by Mozilla](#) website, [CC-BY-SA 3.0](#).

sniffing, or, in other words, it is a way to say that the webmasters knew what they were doing.

— MDN, [X-Content-Type-Options](#), Mozilla Contributors, [CC BY-SA 2.5](#)

The **X-Frame-Options** HTTP response header can be used to indicate whether or not a browser should be allowed to render a page in a `<frame>`, `<iframe>` or `<object>`. Sites can use this to avoid [clickjacking](#) attacks, by ensuring that their content is not embedded into other sites.

Note: The [Content-Security-Policy](#) HTTP header has a [frame-ancestors](#) directive which [obsoletes](#) this header for supporting browsers.

— MDN, [X-Frame-Options](#), Mozilla Contributors, [CC BY-SA 2.5](#)

The **X-XSS-Protection** header has been deprecated by modern browsers and its use can introduce **additional** security issues on the client side. As such, it is recommended to set the header as `X-XSS-Protection: 0` in order to disable the XSS Auditor, and not allow it to take the default behavior of the browser handling the response.

— OWASP Cheat Sheet Series, [Cross Site Scripting Prevention Cheat Sheet](#), OWASP CheatSheets Series Team, [CC BY 3.0](#)

🔗 GDPR: [Art. 5.1.c](#), [Art. 5.1.f](#), [Art. 25](#), [Art. 32.1-2](#).
These headers can help minimize data disclosures.

Cookies

First-party cookies (2)

Domain	Name	Value	Expires on	HttpOnly	Secure	SameSite
.gettr.com	AF_SYNC	1641489961419	2022-01-13 17:26:01Z	❌	❌	❌
.gettr.com	afUserId	9093b9f4-5e9a-47c9-9...	2024-01-06 17:26:01Z	❌	❌	❌

Third-party cookies (2)

Domain	Name	Value	Expires on	HttpOnly	Secure	SameSite
.appsflyer.com	af_id	9093b9f4-5e9a-47c9-9...	2024-01-06 17:26:01Z	❌	✅	✅ (None)

Domain	Name	Value	Expires on	HttpOnly	Secure	SameSite
.onelink.me	af_id	9093b9f4-5e9a-47c9-9...	2024-01-06 17:26:01Z	✗	✓	✓ (None)

HttpOnly means that the cookie can only be read by the server, and not by JavaScript on the client. This can mitigate XSS (cross-site scripting) attacks.

Secure means that the cookie will only be sent over a secure channel (HTTPS). This can mitigate MITM (man-in-the-middle) attacks.

SameSite can be used to instruct the browser to only send the cookie when the request is originating from the same site. This can mitigate CSRF (cross-site request forgery) attacks.

 GDPR: [Rec. 60](#), [Rec. 61](#), [Rec. 69](#), [Rec. 70](#), [Rec. 75](#), [Rec. 78](#), [Art. 5.1.a](#), [Art. 5.1.c](#), [Art. 5.1.e](#), [Art. 21](#), [Art. 22](#), [Art. 32](#).
[e-PD \(2002/58/EC\)](#). Rec. 24, 25, Art. 5.2.
[e-PD revised \(2009/136/EC\)](#). Rec. 65, 66.
 [More information](#)

localStorage


localStorage used:

Key	Value
AF_SESSION	1641489960935
FIRST_TIME_VISIT	false
ZD-buid	f961dce5b60e4t
ZD-store	{"activeEmbed"
ZD-suid	{"id":"97f297dc1
e5981ddee1f82d7f0e2591defb529f948af08804/cly_event	[]
e5981ddee1f82d7f0e2591defb529f948af08804/cly_id	202a422e-9f22-4409-83ef-90975478f145
e5981ddee1f82d7f0e2591defb529f948af08804/cly_queue	[]
e5981ddee1f82d7f0e2591defb529f948af08804/cly_sessi...	1641491760
tusSupport	null

Like with cookies, [web storage](#) can be used to store data in a user's browser. Unlike cookies, web storage data is not sent with HTTP requests: it can only be directly set and accessed by the user's browser (through JavaScript). Compared to cookies, the storage capacity is much larger.

There are two types: `localStorage` data is persistent (not removed when the browser is closed) and never expires, while `sessionStorage` data is removed when the page session ends (unlike with session cookies, a `sessionStorage` session is *per window/tab*).

This can be used to track and profile users by simply using JavaScript to read a user's storage data and send it to a server.

 GDPR: Same as for [cookies](#) above.

Third-party requests

18 requests (18 secure, 0 insecure) to **9** unique hosts.

A third-party request is a request to a domain that's not `gettr.com` or one of its subdomains.

Host	IP	Classification	URLs
connect.facebook.net	157.240.205.11	Social, FingerprintingGeneral (Facebook)	🔍 Show (2)
ekr.zdassets.com	104.18.72.113		🔍 Show (1)
gettr.count.ly	35.223.82.96		🔍 Show (2)
gettr.zendesk.com	104.16.53.111	Content (Zendesk)	🔍 Show (2)
static.zdassets.com	104.18.70.113		🔍 Show (6)
wa.appsflyer.com	52.210.64.118	Advertising (AppsFlyer)	🔍 Show (1)
wa.onelink.me	34.248.93.25		🔍 Show (2)
websdk.appsflyer.com	62.115.253.195	Advertising (AppsFlyer)	🔍 Show (1)
www.googletagmanager.com	64.233.164.97		🔍 Show (1)

We use [Mozilla's version](#) of Disconnect's [open source list of trackers](#) to classify hosts.

🔍 GDPR: [Rec. 69](#), [Rec. 70](#), [Art. 5.1.b-c](#), [Art. 25](#).


IP address

The server **gettr.com** had the IP address **18.66.97.11** during our test.

You can find information about this IP address using third-party tools such as the following:

- bgp.he.net
- [KeyCDN](https://keycdn.com)
- iplocation.io

When using tools that do geolocation, please note that the estimated country can be wrong, especially for websites that use CDNs.

ⓘ Some sites use CDNs – [content delivery networks](#)  – in which case the server location might vary depending on the location of the visitor. This tool, Webbkoll, is currently on a server in Finland.

🔍 Under the GDPR, all EU/EEA countries are considered equally trustworthy, so there is no particular reason under the GDPR to consider any EU country more or less reliable or desirable than any other. The importance of the location of a server comes into play only under GDPR [Art. 23](#), Restrictions, where member states may invoke a number of reasons, notably national security, that enable them to void protections for visitors or web service providers.

For non-EU/EEA territories, it depends (GDPR [Art. 44](#)). For a website, transfers will probably have to rely on adequacy decisions ([Art. 45](#)) made by the European Commission when a third territory has been deemed to have appropriate data protection safe guards in its legislation. However, adequacy decisions cannot always be trusted, as demonstrated in the EU Court of Justice 2015 (C-362/14). Binding corporate rules ([Art. 47](#)) or standard clauses ([Art. 46](#)) may also be used to transfer data, but in the absence of rulings by courts and data protection authorities this is still legally uncertain territory.

Raw headers

Header	Value
--------	-------

accept-ranges	bytes
---------------	-------

cache-control	max-age=0,no-cache,no-store,must-revalidate
---------------	---

content-length	5555
----------------	------

content-type	text/html
--------------	-----------

date	Thu, 06 Jan 2022 17:26:00 GMT
------	-------------------------------

etag	"46e53eccfcad827c7fb9cf73817e1f0b"
------	------------------------------------

last-modified	Thu, 06 Jan 2022 02:54:27 GMT
---------------	-------------------------------

server	AmazonS3
--------	----------

Header	Value
via	1.1 21c2c1b3872c539a34b64bcf45f4054c.cloudfront.net (CloudFront)
x-amz-cf-id	xF2qt7B2aDNRsNGzYcVRHjzkiUfR-Ozu5vYMIrZhofAK6r2OR6aSdw==
x-amz-cf-pop	FRA56-P2
x-cache	Error from cloudfront

What this tool checks (and doesn't check)

This tool attempts to simulate what happens when a user visits a specified page with a typical browser. The browser has no addons/extensions installed, and Do Not Track (DNT) is not enabled, since this is the default setting in most browsers.

External files such as images, scripts and CSS are loaded, but the tool performs no interactions with the page — no links are clicked, no forms are submitted.

Disclaimer: The results presented here might not be 100% correct. Bugs happen. This tool is meant to be used by site owners as a starting point for improvements, not as a rigorous analysis.

Text about HTTPS partly adapted from the CIO Council's [The HTTPS-Only Standard](#) (public domain). [See here](#) for more information.

Test results are stored in memory on our server for 24 hours. We don't show a list of tested URLs. We don't use URLs or test results. We don't log IP addresses. We use one essential session cookie, `_webbkoll_key`, to prevent CSRF attacks.

Developed by [dataskydd.net](#).

The code is [available on GitHub](#).

Feedback? Questions? info@dataskydd.net

Twitter: [@dataskyddnet](#)

[Support us](#)

e1f569b 2021-11-24 16:00:04 +0100