Simple Analytics

> *This pdf is generated on April 23, 2024. Go to [the documentation page] for the most up-to-date version.*

Metrics

Not collecting any information would be silly and unrealistic for an analytics tool. We do collect information that is necessary to show you *simple* analytics, but unlike other analytics tools, we don't collect more than absolutely necessary. Here is a list of what we **do and don't** collect from your visitors.

> *Visit our importance overview for a detailed list of metrics we collect*

Cookies

> *We **do NOT set** any cookies (or use similar technologies)*

We care a great deal about the privacy of your visitors. Cookies are something that can track visitors across multiple pages or even multiple websites. For us this is a hard no. This goes for all similar technologies like (but not limited to) local storage, session cookies, fingerprinting, and IP address hashing.

IP addresses

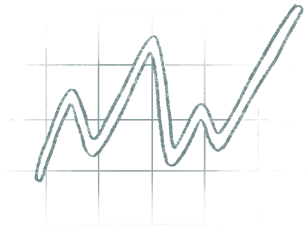
> *We **do NOT collect or store** IP addresses*

We drop the IP address from every single request. Period. We don't save or collect them. We don't hash them with cryptography.

> *Update: Nov 21, 2019. Just to be completely transparent: we found IPs in our logs when requests on our server were failing. We fixed this by filtering all log messages and replacing IPs with zeros using [mmanon](https://www.rsyslog.com/doc/v8-stable/configuration/modules/mmanon.html). From now on, all IPs (like `1.1.1.1` or `2606:4700:4700::1111`) will become `0.0.0.0` or `0:0:0:0:0:0:0:0` before they enter our logs.*

Unique views

> *We **do collect and store** whether visits are unique*



Our unique detection of visits is quite unique itself. Most services use cookies or IP addresses to see if a visitor has visited the website in the past, except we don't use cookies or IP addresses at all. In the UK, for example, websites can't use IP addresses (even hashed) without an active opt-in from each user. This is why Simple Analytics is compatible with all existing privacy laws, including GDPR. You don't need an opt-in for our service.

We detect a unique visit based on the hostname of the *referrer* of the page. When a user comes from one domain to another, their browser shares the previous domain with the next. If the current page's domain is the same as the one in the *referrer*, we know it's a non-unique visit.

Read more on how we register unique page views.

Timestamps

> *We do collect and store timestamps*

We use timestamps to generate the graphs you see on your dashboard, which allows you to analyze changes in your website's performance over various lengths of time.

User agents

> *We do collect and store user agents anonymized*

We detect and exclude bots and spiders based on the visitor's User Agent. We **don't** use User Agents for fingerprinting, only for counting **operating systems, device types, and browsers** in your dashboard. We allow customers to download these counts alongside the User Agent string itself. We do anonymize the User Agent string. For example, when it has a very detailed version number we truncate it from `Chrome/78.0.3904.108` into `Chrome/78.0.0.0`.

Browsers or devices identify themselves to websites. They give themselves some kind of name. For example, a user agent can look like this ([wiki][1]):

```
Mozilla/5.0 (iPad; U; CPU OS 3_2_1) AppleWebKit/531.21.10 (KHTML, like Gecko)
Mobile/7B40598
```

It's not needed, privacy-wise, but we replace long numbers with zeros. After cleanup, it looks like this:

```
Mozilla/5.0 (iPad; U; CPU OS 3_2_0) AppleWebKit/531.21.0 (KHTML, like Gecko)
Mobile/7B40000
```

In some browsers we collect browser name and operating system name without the user agent but via the user agent client hints.

Technical explanation of anonymize function

We drop certain information from the User Agent. Below it the function we use to anonymize the User Agent. Facebook for example sends way more information then just the normal User Agent. We drop all FB related information information from the string. The same goes for `V1_AND_...`.

```
const truncate = (number) => (number + "").slice(0, 5) + (number +
"").slice(5).replace(/[0-9]/g, "0"); const anonymizeAgent = (ua) => ua
.replace(/( \[FB(.*))$/g, "") .replace(/( V1_AND_(.*))$/g, "") .replace(/([0-9]
{5})([0-9]+)/g, (full, first, second) => { return
`${truncate(first)}${second.replace(/[0-9]/g, "0")}`; }) .replace(/([0-9]+)\.
([0-9]+)(\.[0-9]+){1,9}/g, (full, first, second) => { return
`${truncate(first)}.${truncate(second)}${".0".repeat( (full.match(/\./g) ||
[]).length - 1 )}`; });
```

> *Update: Jan 14, 2019. Previously, we didn't store the User Agents, but now we save failed requests to our logs, so we added this as a clarification to the paragraph above.*

> *Update: July 1, 2020. We now store User Agents anonymized. We drop possible identifiers from the User Agents.*

Country

> *We **collect** and **store** country of visitor*

In contrast with most services that collect countries based on IP address, we collect them based on the visitors time zone. This way we don't have to touch their IP address and still are able to define their country. Every country has their own time zone and modern devices automatically update the time zone when the device travels. The time zone is limited to a country, so we can't get data about a city or region within a country. Test this for yourself.

Language

> *We **collect** and **store** the language of the visitor*

Devices are set to a certain language. We collect the language of the device being used by a visitor. We don't show this in our dashboard, but advanced users can get this data out of our APIs. Some language have a different region. For example English is used in the US and in the UK. We also store the region of the language.

ID's

The first ID, data point ID, is a technical ID. When a visitor lands on a page, we send a request to our server. When we measure time on page we need to send a request at the end of a page view. To link those two requests together we match them with an ID. The data point ID. If time on page is not relevant, we can drop this ID.

The second ID, page ID, is used to link multiple events (if a customer collects them) together. Let's say you have 2 events on the same page, you can then combine them together. This is not used by most customers. This page ID will be reset after every page.

The last ID, session ID, is used to link multiple events and pages into one session. It's the same as a page ID, but with multiple pages. When a customer has a SPA (Single Page Application), the session ID is reset when the website is closed. If the customer doesn't have a SPA, the session ID is reset with every navigation.

It's important to note, that all these IDs are not linked to a person or personal data. It's linked to only a page view, page, or session. Another point: these IDs are not stored on the device. Meaning, that if a visitor reloads the page (with F5 for example), the IDs will all be reset.

URLs

> *We partially collect and partially store URLs*

Too much information in the URL can be confusing and can make your stats messy. We only collect and store the first part of the URL. If an URL looks like this `https://example.com/index.html?search=keyword#top` we will only store `https://example.com/index.html`, also known as the protocol (`https`), hostname (`example.com`), and pathname (`/index.html`).

Referrers

> *We do collect and partially store referrers*

Referrers answer the question *"Where did this visitor come from?"*. We have two ways of checking the source of a user visiting your website.

In most cases, browsers send the URL of the previous website as a referrer. We store the referrer the same as URLs (see above). You can find a list of the most popular referrers in your analytics dashboard.

Secondly we check the source of the customer with the UTM-parameters.

UTM-parameters

> *We do collect and store UTM-codes*

UTM codes are bits of text you can add to a link that tell Simple Analytics (as well as other analytics tools) a little bit more information about each link. Here's a sample of what one looks like:

```
https://example.com/landing-page?utm_source=company-x&utm_medium=newsletter&utm_campaign=march
```

Read the UTM guide at Buffer.

We track these UTM codes:

- `utm_source` (e.g.: `duckduckgo.com`)
- `utm_medium` (e.g.: `search`)
- `utm_campaign` (e.g.: `get_customers_02`)
- `utm_content` (e.g.: `sidebar`)
- `utm_term` *(this is deprecated as it is intented to contain user data)*

Website owners can add a URL parameter to links to their website, like `ref=...`, `source=...` or `utm_source=...`. These are all saved as the `utm_source`. Read more on [using URL parameters]().

> *Added: July 20, 2020. We now store all UTM-parameters and not just the `utm_source`. UTM-parameters are not privacy invasive as they are being used for groups of visitors and not for individuals.*

Device dimensions

> *We do collect and store device dimensions*

Collecting the dimensions of a user's browser window (`innerWidth` and `innerHeight` as viewport) and device dimensions (`screen.width` and `screen.height`) allows us to show you the most popular screen sizes. This is useful for making sure your website works great on all screens: phones, tablets, desktops, etc.

> *Updated: July 20, 2020. Previously, we didn't store the device dimensions, but we think they are useful for certain accessibility and design tools, so we added this as a clarification to the paragraph above.*

Time on page

> *We do collect and store how long a page is being viewed*

When a visitor is on a page we collect the amount of seconds a page is viewed. If the page is hidden, we don't count those seconds. [Read this page]() to learn more about how we measure time on page.

Scrolled percentage

> *We do collect and store how far a visitor scrolls on the page*

When a visitor scrolls on a page we record how far they scrolled. We do store this in a percentage with increments of 5%.

Script settings

We send some script settings along with the page view. To identify the embed script and check if the page view comes from a robot. The script settings include things like `version: script_version_2`, `robot: true`.

Do Not Track

> *By default we **do NOT collect or store** any data **if a visitor has Do Not Track** enabled*

The Do Not Track browser setting asks a web application to disable either its own tracking or third-party tracking of an individual user. We never track your users anyway, but by default we also ignore visits with Do Not Track enabled and do not add them to your dashboard. Read more on how to disable this behavior.

> *This pdf is generated on April 23, 2024. Go to the documentation page for the most up-to-date version.*