GUÍA PARA CLIENTES

API DE RAGOSORIO

Cómo instalarla, cómo pedir una clave y qué no cambiar. Para ti y para quien programe tu sitio.

VERSIÓN
23 de noviembre de 2026

EN LÍNEA
docs.ragosorio.com

CONTACTO
studio.ragosorio.com

ÍNDICE

01 **La API de ragosorio**
Qué hace la API, sus tres módulos y por qué ragosorio sigue siendo tu proveedor aunque tengas el código.

02 **Instalar en tu sitio**
La etiqueta para cualquier HTML y el paquete @ragosorio/sitio, con ejemplos para Astro y Next.

03 **Enviar correo**
Enviar desde el servidor de tu sitio con @ragosorio/correo, con el SDK compatible o con fetch. Ejemplos en Node, Workers y Next.

04 **Claves**
La clave pública rsp_ y la secreta rsk_, qué puede cada una, cómo cuidarlas y qué pasa al rotarlas o revocarlas.

05 **Pedir una clave**
Las claves se piden desde tu espacio del estudio. Solo para clientes de ragosorio.

06 **Privacidad y CSP**
Qué se mide y qué nunca sale del navegador, cómo respetar el consentimiento y qué permitir en tu política de seguridad (CSP).

07 **Límites y errores**
Los topes de cada módulo, qué hace el script al llegar a uno y los códigos de error de la API.

08 **Preguntas frecuentes**
Cambios de hosting o de programador, vencimientos, velocidad, varios sitios y más.

01 — QUÉ ES

LA API DE RAGOSORIO

Qué hace la API, sus tres módulos y por qué ragosorio sigue siendo tu proveedor aunque tengas el código.

La API de ragosorio es el servicio con el que cuidamos, medimos y conectamos los sitios que hacemos para ti. Tu sitio lleva un script pequeño (o un paquete) que **solo envía datos**. El trabajo lo hace la API, que es nuestra y vive en `https://api.ragosorio.com`.

Esta guía es para ti y para quien programe o mantenga tu sitio. Explica cómo instalarla, cómo pedir una clave y qué conviene no tocar.

LOS MÓDULOS

La API se organiza por módulos. Cada sitio tiene los suyos y se encienden o se apagan sin cambiar las claves.

MÓDULO	QUÉ HACE	CÓMO SE TIENE
Seguridad (`estado`)	Nos avisa de los errores que ven tus visitas: errores de JavaScript, scripts, estilos o imágenes que no cargan y páginas que tardan más de 8 segundos en cargar. Con la Seguridad vigente, ves el estado de tu sitio en tu espacio del estudio.	Todo sitio de cliente la trae. El primer año es de cortesía (o lo que diga tu contrato). Después va incluida en el mantenimiento o se paga anual.
Métricas (`medicion`)	Visitas, de dónde llegan, qué páginas ven, hasta dónde bajan, qué tocan, mapas de calor y grabaciones de la visita.	Al habilitarla tienes 30 días de prueba. Después va en el mantenimiento o se paga anual.
Correo (`correo`)	Envía correo desde el servidor de tu sitio, con tu propio dominio.	Solo si lo contrataste. No tiene prueba.

Las métricas cubren hasta su fecha (de prueba o de pago) **inclusive**. Si no se renuevan, al día siguiente la API deja de guardar datos y tu sitio sigue funcionando igual. El correo no vence solo: si llega su fecha, el estudio lo revisa contigo y un formulario de contacto nunca deja de llegar por eso.

DÓNDE VES TUS DATOS

- **En tu espacio del estudio**, en studio.ragosorio.com: «Estado» aparece cuando tienes un sitio activo con la Seguridad vigente, y «Métricas» cuando tu sitio es una página web (no una aplicación propia) con las métricas vigentes.

- **Si tu sitio es una aplicación propia**, tu servidor puede consultar la API con la clave secreta. Lo ves en Límites y errores.

Ragosorio sigue siendo tu proveedor

Aunque tengas el código de tu sitio, la API es un servicio que presta ragosorio. El código que recibes solo envía datos: sin la API, esas funciones no hacen nada (y tampoco rompen tu sitio). Por eso hay tres cosas que conviene **no cambiar**:

1. **La dirección de la API.** Es `https://api.ragosorio.com`, tanto en la etiqueta del script como en el correo. El paquete ya la trae, así que nadie tiene que escribirla a mano.
2. **El script o el paquete.** Cárgalo siempre desde la API, tal como te lo damos. El script toma la dirección de la API del lugar desde donde se cargó: si alguien lo copia a otro servidor, deja de funcionar.
3. **Las claves.** Las crea, las rota y las revoca solo el estudio. No las compartas, no las reutilices en otro sitio y no las subas al repositorio. Todo está en Claves.

Si cambias de dominio, de hosting o de programador, avísanos antes: la clave pública solo funciona desde las direcciones que tenemos registradas para tu sitio.

A quién escribir

Escríbenos desde tu espacio en studio.ragosorio.com, en tu conversación con el estudio. Ahí también se piden las claves (ver Pedir una clave). Si quien programa tu sitio no tiene acceso a tu espacio, pídele que te escriba a ti o pásale esta guía.

02 — INSTALAR

INSTALAR EN TU SITIO

La etiqueta para cualquier HTML y el paquete @ragosorio/sitio, con ejemplos para Astro y Next.

Un mismo script carga la Seguridad y las Métricas según los módulos que tenga tu sitio. Solo necesita la **clave pública** de tu sitio, la que empieza por `rsp_`.

ANTES DE EMPEZAR

- **La clave pública del sitio** (`rsp_…`). Si no la tienes, mira Pedir una clave.
- **Las direcciones de tu sitio registradas.** La clave pública solo funciona desde los orígenes que el estudio registró para tu sitio, y el origen tiene que coincidir exacto: `https://tusitio.com` y `https://www.tusitio.com` son dos orígenes distintos. Si tu sitio se ve en más de una dirección (con y sin `www`, o un dominio de pruebas), pide que las registremos todas.
- En tu computadora (`localhost`) no se mide nada mientras ese origen no esté registrado. Es lo esperado.

OPCIÓN A: LA ETIQUETA (CUALQUIER HTML)

Pega esta línea en el `<head>` de todas tus páginas, con la clave de tu sitio:

```HTML
<script defer src="https://api.ragosorio.com/v1/sitio.js" data-key="rsp_…"></script>
```

Pesa pocos KB y se carga con `defer`, así que no frena tu página. La grabación de la visita se descarga aparte y solo si tu sitio la tiene activa.

Astro

En el layout que usan todas tus páginas (por ejemplo `src/layouts/Layout.astro`):

ASTRO

```
<head>
  <!-- … -->
  <script is:inline defer src="https://api.ragosorio.com/v1/sitio.js" data-key="rsp_…">
</script>
</head>
```

`is:inline` le dice a Astro que deje la etiqueta tal cual, sin procesarla.

Next.js (App Router)

En `app/layout.tsx`, con el componente `Script` de Next:

TSX

```
import Script from "next/script";

export default function RootLayout({ children }: { children: React.ReactNode }) {
  return (
    <html lang="es">
      <body>
        {children}
        <Script src="https://api.ragosorio.com/v1/sitio.js" data-key="rsp_…"
strategy="afterInteractive" />
      </body>
    </html>
  );
}
```

OPCIÓN B: EL PAQUETE `@ragosorio/sitio`

Para proyectos con npm (Next, Astro, Vite, Svelte…). El paquete ya trae la dirección de la API: solo le pasas la clave.

JAVASCRIPT

```
import { iniciar } from "@ragosorio/sitio";

iniciar({ clave: "rsp_…" });
```

> NOTA
>
> **Todavía no está publicado en npm.** Mientras tanto, usa la etiqueta (opción A) o el paquete que te entrega el estudio junto con el código de tu sitio. Cuando se publique, el nombre seguirá siendo `@ragosorio/sitio`.

`iniciar` es seguro de llamar en cualquier lugar: en el servidor no hace nada y, si se llama dos veces, solo cuenta la primera.

Next.js con el paquete

Un componente cliente que se monta una vez en el layout:

TSX

```tsx
// app/medicion.tsx
"use client";

import { useEffect } from "react";
import { iniciar } from "@ragosorio/sitio";

export function Medicion() {
  useEffect(() => {
    iniciar({ clave: "rsp_…" });
  }, []);
  return null;
}
```

TSX

```tsx
// app/layout.tsx
import { Medicion } from "./medicion";

export default function RootLayout({ children }: { children: React.ReactNode }) {
  return (
    <html lang="es">
      <body>
        {children}
        <Medicion />
      </body>
    </html>
  );
}
```

Astro con el paquete

ASTRO

```astro
<script>
  import { iniciar } from "@ragosorio/sitio";
  iniciar({ clave: "rsp_…" });
</script>
```

Vite, React, Svelte u otros

En el archivo de entrada (por ejemplo `src/main.ts`):

```javascript
import { iniciar } from "@ragosorio/sitio";

iniciar({ clave: "rsp_…" });
```

La clave pública puede ir escrita en el código: está hecha para verse en el HTML. La secreta nunca (ver Claves).

Comprobar que funciona

1. Abre tu sitio en una de sus direcciones registradas.
2. En las herramientas del navegador, pestaña **Red**, busca `api.ragosorio.com`.
3. Deberías ver `sitio.js` y `/v1/config` con respuesta **200**. Si tu sitio tiene Métricas, enseguida aparece `/v1/m` con **202**, y luego se repite de vez en cuando, cuando hay algo nuevo que contar (un clic, otra página, al salir).

Si `/v1/config` responde **403** con `origin_not_allowed`, esa dirección no está registrada en tu sitio: escríbenos. Los demás códigos están en Límites y errores.

Marcar partes de tu página (opcional)

Con estos atributos las Métricas entienden mejor tu página. Ninguno es obligatorio.

Atributo	Qué hace
`data-rs-section="nombre"`	Mide la atención en esa sección (el tiempo que estuvo en pantalla). Las `section` con `id` se miden solas.
`data-rs-click="nombre"`	Le da un nombre estable a un botón o enlace, aunque cambie su texto o su diseño.
`data-rs-mask`	Tapa ese texto en la grabación y no lo envía como texto de un clic.
`data-rs-block`	Oculta ese bloque entero en la grabación.

```html
<section data-rs-section="precios">
  <a href="/reservar" data-rs-click="reservar-hero">Reservar</a>
  <p data-rs-mask>Datos que no deben verse en una grabación</p>
</section>
```

Sitios de una sola página (SPA)

Si tu sitio cambia de página sin recargar (Next, Astro con transiciones, React Router…), el script lo detecta solo y cuenta cada página. No tienes que hacer nada.

El mapa de calor

Para dibujar el mapa de calor, el estudio abre tu página dentro de un marco con `?rs_mapa=1` en la dirección. En ese modo el script **no mide ni graba**: solo se desplaza al ritmo del mapa. Si tu sitio usa otra herramienta de medición, apágala cuando la dirección lleve `rs_mapa=1`, para que esas aperturas no cuenten como visitas.

03 — CORREO

ENVIAR CORREO

Enviar desde el servidor de tu sitio con @ragosorio/correo, con el SDK compatible o con fetch. Ejemplos en Node, Workers y Next.

El módulo **Correo** envía correo desde el servidor de tu sitio, con tu dominio: formularios de contacto, reservas, confirmaciones. Funciona en Node, Workers, Deno y Bun, y desde cualquier lenguaje que pueda hacer una petición HTTP.

LO QUE NECESITAS

- **El módulo Correo contratado** para tu sitio.
- **La clave secreta** del sitio, la que empieza por `rsk_`, guardada como variable de entorno `RAGOSORIO_KEY` en tu servidor.
- **Tu dominio verificado.** El remitente (`from`) tiene que ser de un dominio verificado en tu cuenta de correo. El estudio te da los registros DNS y los revisa contigo.

> CUIDADO
>
> **Solo desde el servidor.** La clave secreta nunca va en el navegador: ni en el HTML, ni en un componente cliente, ni en una variable que empiece por `NEXT_PUBLIC_`, `PUBLIC_` o `VITE_`. La clave pública (`rsp_`) no envía correo.

CON `@ragosorio/correo`

JAVASCRIPT

```
import { Correo } from "@ragosorio/correo";

const correo = new Correo(process.env.RAGOSORIO_KEY); // rsk_…, solo en el servidor

const { data, error } = await correo.emails.send({
  from: "Reservas <reservas@tusitio.com>",
  to: "cliente@ejemplo.com",
  subject: "Tu reserva",
  html: "<p>¡Listo! Te esperamos.</p>",
});

if (error) {
  console.error(error.statusCode, error.name, error.message);
}
```

Sin argumentos, `new Correo()` lee `RAGOSORIO_KEY` del entorno (en Node). El paquete no lanza errores: siempre devuelve `{ data, error }`, así tu formulario decide qué hacer si un correo no salió.

> NOTA
>
> **Todavía no está publicado en npm.** Mientras tanto, usa el paquete que te entrega el estudio con el código de tu sitio, el SDK compatible o `fetch` (los dos de abajo).

Workers

En un Worker las variables llegan en `env`. Guarda la clave como **secreto** del Worker, nunca en su archivo de configuración.

JAVASCRIPT

```
import { Correo } from "@ragosorio/correo";

export default {
  async fetch(request, env) {
    const correo = new Correo(env.RAGOSORIO_KEY);
    const { error } = await correo.emails.send({
      from: "Web <hola@tusitio.com>",
      to: "equipo@tusitio.com",
      subject: "Nuevo mensaje",
      text: await request.text(),
    });
    return new Response(error ? "No se pudo enviar" : "Enviado", { status: error ? 502 : 200
});
  },
};
```

Next.js: route handler

TYPESCRIPT

```
// app/api/contacto/route.ts
import { Correo } from "@ragosorio/correo";

const correo = new Correo(process.env.RAGOSORIO_KEY);

export async function POST(request: Request) {
  const datos = await request.formData();
  const nombre = String(datos.get("nombre") ?? "").slice(0, 100);
  const mensaje = String(datos.get("mensaje") ?? "").slice(0, 5000);

  const { data, error } = await correo.emails.send({
    from: "Web <hola@tusitio.com>",
    to: "equipo@tusitio.com",
    subject: `Mensaje de ${nombre || "la web"}`,
    text: mensaje,
  });

  if (error) return Response.json({ ok: false }, { status: 502 });
  return Response.json({ ok: true, id: data.id });
}
```

Next.js: server action

TYPESCRIPT

```
// app/contacto/acciones.ts
"use server";

import { Correo } from "@ragosorio/correo";

const correo = new Correo(process.env.RAGOSORIO_KEY);

export async function enviarContacto(datos: FormData) {
  const { error } = await correo.emails.send({
    from: "Web <hola@tusitio.com>",
    to: "equipo@tusitio.com",
    subject: "Nuevo mensaje desde la web",
    text: String(datos.get("mensaje") ?? "").slice(0, 5000),
  });
  return { ok: !error };
}
```

CON EL SDK `resend` DE NPM

La API de correo tiene la misma forma que el SDK `resend` de npm. Si tu sitio ya lo usa, puede seguir con él cambiando solo dos variables de entorno:

TERMINAL

```
RESEND_BASE_URL=https://api.ragosorio.com/v1/correo
RESEND_API_KEY=rsk_…
```

JAVASCRIPT

```
import { Resend } from "resend";

const resend = new Resend(process.env.RESEND_API_KEY);
const { data, error } = await resend.emails.send({
  from: "Web <hola@tusitio.com>",
  to: ["equipo@tusitio.com"],
  subject: "Nuevo mensaje",
  html: "<p>…</p>",
});
```

CON `fetch` O CUALQUIER LENGUAJE

JAVASCRIPT

```
const r = await fetch("https://api.ragosorio.com/v1/correo/emails", {
  method: "POST",
  headers: {
    authorization: `Bearer ${process.env.RAGOSORIO_KEY}`,
    "content-type": "application/json",
  },
  body: JSON.stringify({
    from: "Web <hola@tusitio.com>",
    to: ["equipo@tusitio.com"],
    subject: "Nuevo mensaje",
    text: "…",
  }),
});
const resultado = await r.json(); // 200 → { id }; si no, { statusCode, name, message }
```

RUTAS

Todas bajo `https://api.ragosorio.com/v1/correo`, con `Authorization: Bearer rsk_…`.

RUTA	QUÉ HACE
`POST /emails`	Envía un correo. Responde `200 { "id": "…" }`.
`POST /emails/batch`	Envía de 1 a 100 correos en una lista, sin adjuntos. Se validan todos antes de enviar ninguno. Responde `200 { "data": [{ "id": "…" }, …] }`.
`GET /emails/{id}`	Consulta un envío de tu cuenta: remitente, destinatarios, asunto, contenido y su último estado (`last_event`).

Con `@ragosorio/correo` son `correo.emails.send`, `correo.emails.batch` y `correo.emails.get`.

CAMPOS DE UN CORREO

CAMPO	QUÉ ES
`from`	Remitente, de un dominio verificado de tu cuenta: `"Nombre <hola@tusitio.com>"`. Obligatorio.
`to`	Una dirección o una lista. Obligatorio.
`cc`, `bcc`, `reply_to`	Una dirección o una lista.
`subject`	Asunto. Obligatorio (salvo con plantilla).
`html`, `text`	El contenido. Hace falta al menos uno. Si solo mandas `html`, el texto se genera solo.
`headers`	Cabeceras adicionales.
`tags`	Hasta 20 etiquetas `{ name, value }` (letras, números, `_` y `-`).
`attachments`	Adjuntos con `filename` y `content` en base64.
`template`	Con `fetch` o el SDK compatible: `{ "id": "…", "variables": { … } }`, una plantilla guardada en tu cuenta. No se combina con `html` ni `text`. Las variables `{{nombre}}` se escapan en el HTML.

Límites de un envío:

- Hasta **50 destinatarios** entre `to`, `cc` y `bcc`.
- Hasta **10 adjuntos** y 10 MB en total. No se aceptan ejecutables ni adjuntos por `path` (URL).
- La petición completa no puede pasar de **2 MB**, adjuntos en base64 incluidos. En la práctica, los adjuntos deben ser pequeños.
- Todavía no se pueden programar envíos (`scheduled_at`): se rechazan con 422.

LO QUE HACE LA API POR TI

- **Supresiones.** Si una dirección rebotó de forma permanente o marcó un correo como no deseado, queda suprimida en tu cuenta y no se le vuelve a enviar. El envío responde igual con su `id`.
- **Cuota del mes.** Tu cuenta tiene un tope de destinatarios al mes. El estudio recibe un aviso al acercarse y al llegar. Pasado el tope, la API responde **429** `monthly_quota_exceeded`.
- **Registro.** En «Correo» de tu espacio del estudio ves lo que salió, a quién, sus eventos y las supresiones. Nadie del estudio lee el contenido de tus correos.

Los errores del correo y qué hacer con cada uno están en Límites y errores.

Si tu sitio ya envía con una clave `rs_`

Las claves anteriores, que empiezan por `rs_`, siguen funcionando igual, con la dirección que te dimos. No cambies nada hasta que el estudio te lo proponga.

04 — CLAVES

CLAVES

La clave pública rsp_ y la secreta rsk_, qué puede cada una, cómo cuidarlas y qué pasa al rotarlas o revocarlas.

Cada sitio tiene dos tipos de clave. Los módulos se encienden y se apagan en el sitio, sin cambiar ninguna de las dos.

	PÚBLICA `RSP_…`	SECRETA `RSK_…`
Dónde va	En el HTML (`data-key`) o en `iniciar({ clave })`.	En las variables de entorno secretas del servidor de tu sitio (`RAGOSORIO_KEY`).
Qué puede	Solo enviar datos (Seguridad y Métricas), y solo desde las direcciones registradas de tu sitio.	Lo que habiliten los módulos de tu sitio: enviar correo y consultar tu sitio desde tu servidor.
Cuántas	Una vigente por sitio.	Una o varias (por ejemplo, una por entorno o por servidor).
Si alguien la ve	No pasa nada grave: está hecha para verse.	Es como la llave de tu correo: hay que rotarla.

Las dos tienen el mismo formato: `rsp_` o `rsk_` seguido de 40 letras y números.

CUIDAR LA CLAVE SECRETA

La clave secreta se muestra **una sola vez**. Nosotros guardamos solo una huella de ella (un resumen que no se puede revertir), así que no podemos volver a leerla: si se pierde, se crea otra.

> CUIDADO
>
> **Nunca pongas la clave secreta:**
>
> - en el navegador: ni en el HTML, ni en un componente cliente, ni en una variable `NEXT_PUBLIC_` , `PUBLIC_` o `VITE_` ;
> - en el repositorio, ni siquiera en un repositorio privado;
> - en un archivo `.env` que se suba (revisa que `.env` esté en tu `.gitignore`);
> - en capturas de pantalla, chats, correos, tickets o documentos compartidos.

Guárdala en el gestor de secretos de tu hosting. Si el estudio administra tu sitio, la ponemos nosotros directamente en sus secretos y no tienes que tocarla.

Si una clave se filtra

Escríbenos enseguida desde tu espacio en studio.ragosorio.com y pide que la roten. No esperes a confirmar si alguien la usó.

Rotar

Rotar crea una clave nueva y deja la anterior funcionando un tiempo, para que la cambies en tu sitio sin cortar nada:

- Por defecto, la anterior sigue sirviendo **7 días** (se puede acordar entre 0 y 30).
- Al terminar ese plazo, la anterior se revoca sola.
- Cambia la clave en tu sitio antes de esa fecha.

Revocar

Revocar deja la clave sin servir **al instante** para el correo, y en pocos minutos para el script del navegador (que guarda la configuración unos minutos). Es lo que se hace cuando una clave se filtró y no hace falta un plazo, o cuando un sitio deja de usar la API.

Con una clave revocada, el script deja de enviar sin romper tu sitio, y el correo responde con un error claro (ver Límites y errores).

Quién administra las claves

Solo el estudio crea, rota y revoca claves. No hay un panel para crearlas tú: se piden (ver Pedir una clave). Las claves de correo anteriores, que empiezan por `rs_`, siguen funcionando.

05 — PEDIR UNA CLAVE

PEDIR UNA CLAVE

Las claves se piden desde tu espacio del estudio. Solo para clientes de ragosorio.

Las claves de la API **se piden**: no hay un panel para crearlas tú. Por ahora están disponibles solo para clientes de ragosorio.

PASO A PASO

1. Entra a tu espacio en studio.ragosorio.com.
2. En el inicio, busca el bloque **«API de ragosorio»**.
3. Toca **«Pedir una clave de la API»**.
4. Tu solicitud queda en tu conversación con el estudio. Ahí te escribimos si nos falta algo.
5. El estudio prepara la clave y te la entrega por un canal seguro. Si administramos tu sitio, la ponemos nosotros directamente en sus secretos.

QUÉ CONVIENE DECIRNOS

Así la preparamos de una vez:

- **El sitio**: su nombre y su dirección.
- **Todas las direcciones** desde donde se abre (con y sin `www`, un dominio de pruebas…). La clave pública solo funciona desde las direcciones registradas.
- **Qué clave necesitas**:
  - la **pública** (`rsp_`) para la Seguridad y las Métricas en tu página;
  - la **secreta** (`rsk_`) para enviar correo o consultar la API desde tu servidor. El correo tiene que estar contratado.
- **Para qué entorno**, si necesitas más de una secreta (por ejemplo, producción y pruebas).

CUANDO LA RECIBAS

- **La pública** va en la etiqueta o en `iniciar` (ver Instalar).

- **La secreta** va directo al gestor de secretos de tu hosting, como `RAGOSORIO_KEY`. No la copies en ningún otro lugar (ver Claves).

Si algo no funciona, revisa Límites y errores o escríbenos en la misma conversación.

PRIVACIDAD Y CSP

Qué se mide y qué nunca sale del navegador, cómo respetar el consentimiento y qué permitir en tu política de seguridad (CSP).

QUÉ SE MIDE

Con las Métricas activas, el script cuenta:

- Las páginas vistas (también en sitios de una sola página), el tiempo activo en cada una y hasta dónde se bajó.
- La atención por sección y por franja de la página (para el mapa de calor).
- Los clics: en qué elemento, su texto corto y su posición. También los enlaces a otros sitios y los clics repetidos con enojo.
- Los formularios empezados y enviados. **Nunca lo que se escribe en ellos.**
- De dónde llega la visita: campaña (UTM), sitio de origen y tipo de dispositivo.
- La ubicación aproximada (país, región y ciudad), deducida de la conexión. No es exacta, sobre todo la ciudad.
- La grabación de la visita, si tu sitio la tiene activa. Por defecto se graba una parte de las visitas, no todas.

Con la Seguridad, el script avisa de los errores que ven tus visitas: como mucho 10 por página y cada uno una sola vez.

QUÉ NUNCA SALE DEL NAVEGADOR

- **Lo que se escribe en un campo.** En la grabación, todos los campos van tapados.
- **El texto marcado con** `data-rs-mask`: se tapa en la grabación y no se envía como texto de un clic.
- **Los bloques con** `data-rs-block`: no aparecen en la grabación.
- **El query string de las direcciones** (lo que va después de `?`). De la dirección de entrada solo se conservan las UTM, `ref` y los identificadores de clic de anuncios; los enlaces a otros sitios se guardan sin query string. En los avisos de errores, las direcciones van sin query string y los correos que aparezcan se reemplazan por `[correo]`.

Además, la API no guarda la dirección IP ni usa huellas del navegador.

Cookies y almacenamiento

El script **no usa cookies**. Guarda en el navegador de la visita:

Dónde	Nombre	Qué es
`localStorage`	`rs_v`	Un identificador al azar de la visita, para distinguir visitas nuevas de las que vuelven. No identifica a la persona.
`sessionStorage`	`rs_visita`	La visita en curso (se borra al cerrar la pestaña).

Si el navegador no deja guardar nada, la visita se mide solo en esa página.

Consentimiento: `window.rsConsent`

Si tu sitio pide permiso antes de medir, pon `window.rsConsent = false` **antes** de que cargue el script. Así no mide ni graba nada, y tampoco envía avisos de errores.

HTML

```
<script>
  // Por ejemplo, según la respuesta guardada de tu aviso de privacidad.
  if (localStorage.getItem("mi-sitio-consentimiento") !== "si") window.rsConsent = false;
</script>
<script defer src="https://api.ragosorio.com/v1/sitio.js" data-key="rsp_…"></script>
```

Si tu sitio tiene CSP (más abajo), pon ese código en un archivo propio en lugar de escribirlo en línea. El script lo lee al cargar la página. Si la persona cambia de opinión, aplica desde la siguiente vez que se cargue la página. Con el paquete, en lugar de poner `rsConsent`, también puedes llamar a `iniciar` solo cuando haya permiso.

> **NOTA**
>
> Esta guía no es asesoría legal. Si tu sitio necesita un aviso de privacidad o de cookies, revisa con quien te asesora lo que aplica en tu país.

Cuánto se guarda

Qué	Cuánto
Visitas (páginas, tiempo, scroll, origen)	13 meses
Resumen diario de tu sitio	Sin vencimiento (no tiene datos de personas)
Clics y atención por sección	3 meses

QUÉ	CUÁNTO
Grabaciones	30 días

POLÍTICA DE SEGURIDAD (CSP)

Si tu sitio envía una cabecera `Content-Security-Policy`, agrega la API:

DIRECTIVA	CON LA ETIQUETA	CON EL PAQUETE
`script-src`	`https://api.ragosorio.com`	No hace falta: el código va dentro de tu sitio.
`connect-src`	`https://api.ragosorio.com`	`https://api.ragosorio.com`

Con la etiqueta, la grabación también se descarga de `https://api.ragosorio.com`, así que `script-src` ya la cubre.

HTTP

```
Content-Security-Policy: default-src 'self'; script-src 'self' https://api.ragosorio.com;
connect-src 'self' https://api.ragosorio.com; frame-ancestors 'self'
https://studio.ragosorio.com
```

El mapa de calor muestra tu página dentro de tu espacio del estudio. Si tu sitio no se deja abrir en un marco (`frame-ancestors` o `X-Frame-Options`), permite `https://studio.ragosorio.com` en `frame-ancestors`. Si no, el mapa se ve igual, pero sin tu página debajo.

NO CONTAR TUS PROPIAS VISITAS

El script no trae un interruptor para esto. Tu sitio puede poner `window.rsConsent = false` cuando lo necesites (por ejemplo, según una preferencia guardada solo en tu navegador). Si quieres, el estudio lo deja preparado.

07 — LÍMITES Y ERRORES

LÍMITES Y ERRORES

Los topes de cada módulo, qué hace el script al llegar a uno y los códigos de error de la API.

La regla de fondo: **la API nunca rompe tu sitio.** Sin clave, con una clave revocada, con un módulo apagado o vencido, o sin cupo, el script deja de enviar en silencio y tu página sigue igual. El correo, en cambio, responde con un error claro para que tu formulario sepa que no salió.

TOPES DE LAS MÉTRICAS

Son los topes por defecto de cada sitio. El estudio puede ajustarlos dentro de un máximo.

QUÉ	TOPE POR SITIO
Visitas medidas al mes	50,000
Eventos al mes (clics, atención)	800,000
Visitas grabadas	10 % de las visitas, hasta 300 al mes
Espacio de grabaciones	300 MB al mes y 5 MB por visita

Qué pasa al llegar a un tope:

- **Sin cupo de visitas**, el script deja de medir las visitas nuevas hasta el mes siguiente. Tu sitio no cambia.
- **Sin cupo de grabación**, se deja de grabar (la medición sigue).
- **Si una petición recibe 429 o 402**, el script deja de enviar y de grabar durante esa visita.

TOPES DE LA SEGURIDAD

- Como mucho **10 errores por página**, y cada error distinto una sola vez.
- Como mucho **300 errores por sitio al día**. Pasado ese tope, la API responde 429 y el script deja de enviar errores en esa visita.

Errores del script y de la API

Estos son los códigos que puedes ver en la pestaña **Red** del navegador. Todos responden `{ "ok": false, "code": "…" }`.

Código	`code`	Qué significa	Qué hacer
401	`missing_key`	Falta la clave.	Revisa `data-key` o `iniciar({ clave })`.
403	`invalid_key`	La clave no existe, está mal copiada o fue revocada.	Copia la clave otra vez o pide una nueva.
403	`origin_not_allowed`	La página se abrió desde una dirección no registrada en tu sitio.	Pide que registremos esa dirección.
403	`module_not_enabled`	El módulo no está activo en tu sitio.	Escríbenos si debería estarlo.
402	`module_expired`	El módulo venció (la prueba o lo pagado terminó).	Renueva con el estudio.
429	`budget` / `replay_budget`	Se llegó a un tope del mes o del día.	Nada: se retoma solo en el siguiente periodo.
400 / 413	`invalid_batch`, `too_large`…	Un envío mal formado o demasiado grande.	No debería pasar con el script oficial; escríbenos.
500	`internal`	Un problema de nuestro lado.	Nada: el script no reintenta y tu sitio sigue igual.

Si la clave no tiene el formato correcto (`rsp_` y 40 caracteres), el script ni siquiera llama a la API.

Correo

El correo responde con la misma forma que el SDK `resend` de npm: `{ "statusCode", "name", "message" }`.

Código	`name`	Cuándo sale
401	`missing_api_key`	Falta la cabecera `Authorization: Bearer …`.
401	`restricted_api_key`	Usaste la clave pública (`rsp_`), el módulo Correo no está activo en tu sitio o el sitio no tiene una cuenta de correo.
403	`invalid_api_key`	La clave no existe, está mal copiada o fue revocada.
403	`validation_error`	El dominio del remitente no está registrado o no está verificado en tu cuenta, o la clave solo puede enviar desde otro dominio.
403	`account_paused`	El envío de tu cuenta está en pausa. Escríbenos.
403	`tenant_paused`	El envío se pausó por la reputación de tu cuenta (por ejemplo, muchos rebotes o quejas). Escríbenos.

CÓDIGO	NAME	CUÁNDO SALE
404	`not_found`	El correo que consultas no es de tu cuenta, o la plantilla no existe.
400	`invalid_idempotency_key`	La cabecera `Idempotency-Key` no tiene entre 1 y 256 caracteres.
413	`validation_error`	La petición pasa de 2 MB.
422	`validation_error`	Un campo no es válido: dirección mal escrita, más de 50 destinatarios, adjunto no permitido, `scheduled_at` …
422	`missing_required_field`	Falta un campo obligatorio (`from`, `to`, `subject`, `html` o `text`).
429	`monthly_quota_exceeded`	Tu cuenta llegó a su tope de destinatarios del mes.
429	`rate_limit_exceeded`	Demasiados envíos seguidos. Espera un momento y vuelve a intentar.
500	`application_error`	Un problema de nuestro lado, o el correo no se pudo entregar (queda registrado como fallido).
503	`application_error`	El servicio de correo no está disponible en ese momento.

Con `@ragosorio/correo`, un problema de red (sin respuesta de la API) llega como `statusCode: 0` y `name: "network_error"`.

El correo no vence solo: no verás un 402 al enviar. Si se llega a la fecha de tu plan, el estudio lo revisa contigo.

CONSULTAS DESDE TU SERVIDOR

Con la clave secreta, tu servidor puede preguntar por su sitio. Sirve sobre todo para aplicaciones propias.

TERMINAL

```
curl https://api.ragosorio.com/v1/yo -H "Authorization: Bearer $RAGOSORIO_KEY"
```

JSON

```
{
  "ok": true,
  "site": { "slug": "tusitio", "name": "Tu sitio", "kind": "platform" },
  "modules": { "estado": "active", "medicion": "active", "correo": "active" }
}
```

Cada módulo viene como `active`, `expired`, `paused` u `off`. Con la clave pública, esta ruta responde 403 `public_key_not_allowed`, y si la Seguridad de tu sitio venció, 402 `module_expired`.

Para saber si la API está en línea, sin clave:

TERMINAL

```
curl https://api.ragosorio.com/v1/salud
# { "ok": true, "api": "ragosorio", "version": 1 }
```

08 — PREGUNTAS

PREGUNTAS FRECUENTES

Cambios de hosting o de programador, vencimientos, velocidad, varios sitios y más.

¿PUEDO CAMBIAR DE HOSTING O DE PROGRAMADOR?

Sí. El script, el paquete y las claves no dependen de dónde esté alojado tu sitio. Lo único que importa es la **dirección** desde donde se abre: si cambia el dominio, avísanos antes para registrar la nueva. Pásale esta guía a quien vaya a trabajar en tu sitio.

¿PUEDO COPIAR `sitio.js` A MI SERVIDOR?

No. El script toma la dirección de la API del lugar desde donde se cargó: copiado en tu servidor, enviaría los datos a tu propio dominio y dejaría de funcionar. Cárgalo siempre desde `https://api.ragosorio.com/v1/sitio.js`, o usa el paquete.

¿QUÉ PASA SI VENCE UN MÓDULO?

Tu sitio sigue funcionando igual. La API responde 402 y el script deja de enviar en silencio. Las Métricas dejan de verse en tu espacio al día siguiente de su fecha, sin días de gracia; si renuevas tarde, esos días quedan sin datos. El correo no vence solo.

¿Y SI LA API NO RESPONDE?

Tu sitio sigue igual: el script se carga con `defer`, no bloquea la página y, si algo falla, no hace nada. Esos minutos quedan sin medir.

¿HACE MÁS LENTO MI SITIO?

No de forma apreciable. El script pesa pocos KB, se carga después del contenido y agrupa lo que envía en pocas peticiones. La grabación se descarga aparte y solo si tu sitio la tiene activa.

¿Puedo usar la misma clave en otro sitio?

No. Cada sitio se registra aparte, con su clave pública y sus direcciones. Pide una clave para cada sitio (ver Pedir una clave).

¿Puedo probarlo en mi computadora?

Solo si el estudio registra esa dirección (por ejemplo `http://localhost:4321`) en tu sitio. Si no, el script no mide nada en local, y es lo esperado.

¿Por qué no encuentro los paquetes en npm?

Todavía no están publicados. Mientras tanto, usa la etiqueta del script o el paquete que te entrega el estudio con el código. Cuando se publiquen, los nombres seguirán siendo `@ragosorio/sitio` y `@ragosorio/correo`.

¿Mi sitio necesita un aviso de cookies?

El script no usa cookies, pero guarda un identificador al azar en el navegador (ver Privacidad y CSP). Si tu sitio pide consentimiento, usa `window.rsConsent`. Qué aviso necesitas depende de tu país: revísalo con quien te asesora.

¿Puedo enviar correo desde el navegador?

No. El correo necesita la clave secreta, y la clave secreta solo vive en el servidor. Desde el navegador, envía el formulario a tu servidor (una ruta o una acción de servidor) y que este llame a la API (ver Enviar correo).

Perdí la clave secreta. ¿Me la pueden volver a enviar?

No podemos: solo guardamos una huella de ella. Pide una rotación y te damos una nueva (ver Claves).

¿CON QUIÉN HABLO?

Con el estudio, desde tu espacio en studio.ragosorio.com. Ahí también pides claves y nos cuentas cualquier cambio en tu sitio.

¿Dudas? Escríbenos desde tu espacio en studio.ragosorio.com.

ragosorio · Actualizado el 23 de noviembre de 2026