

INTEGRACIONES

API de SendyPOP

Conectá tu CRM, tienda o sistema propio con SendyPOP: envío transaccional, contactos y grupos, campañas, automatizaciones disparadas por eventos, avisos en tiempo real por webhooks y reportes.

Contenido

1. Alta de la conexión
2. Autenticación, IPs permitidas y límites
3. Respuestas y errores
4. Envío transaccional
5. Contactos, grupos y etiquetas
6. Campañas
7. Eventos y automatizaciones
8. Webhooks
9. Reportes y dominios
10. Monitoreo de la conexión
11. Checklist de puesta en producción

DATO	VALOR
URL base	<code>https://sendypop.com/api</code> (todas las rutas empiezan con <code>/v1</code>)
Formato	JSON en UTF-8 (<code>Content-Type: application/json</code>)
Autenticación	<code>Authorization: Bearer <API key></code> o <code>X-API-Key: <API key></code>
Referencia completa	<code>https://sendypop.com/api/docs</code> (OpenAPI, se puede importar en Postman)

1 Alta de la conexión

Cada API key pertenece a una sola empresa y sólo opera sobre sus datos. La crea el equipo de Soporte en **Configuración de API** o, si la empresa tiene la autogestión habilitada, su administrador desde **API** en el panel. Para cada integración:

1. **Una clave por integración y ambiente** (por ejemplo `tienda-produccion` y `tienda-pruebas`). Así se puede revocar una sin cortar las demás. Las claves de **Pruebas** (`cmpk_test_...`) no envían emails reales: ver Modo prueba.
2. **Permisos mínimos**. Elegí sólo lo que la integración usa (tabla abajo).

3. **IPs permitidas.** Si el sistema que llama tiene IPs fijas, cargalas (IP o rango CIDR, IPv4 o IPv6). Un pedido desde otra dirección se rechaza con `403 ip_not_allowed`. Sin lista, la clave funciona desde cualquier IP.
4. **El secreto se muestra una sola vez**, al crearla. Guardalo en el gestor de secretos de tu sistema; nunca en el código, tickets ni correos. Si se pierde o se filtra: revocar y crear otra.
5. Para recibir avisos de entregas, rebotes, aperturas o bajas, configurará un webhook.

PERMISO	PERMITE
<code>messages:send</code> · <code>messages:read</code>	Enviar emails transaccionales · consultar su estado
<code>contacts:read</code> · <code>contacts:write</code>	Ver · crear, editar, importar y eliminar contactos, grupos y etiquetas
<code>campaigns:read</code> · <code>campaigns:write</code> · <code>campaigns:send</code>	Ver · crear y editar · enviar y programar campañas
<code>automations:read</code> · <code>automations:write</code>	Ver · crear, activar y pausar automatizaciones
<code>events:read</code> · <code>events:write</code>	Consultar · registrar eventos que disparan automatizaciones
<code>webhooks:read</code> · <code>webhooks:write</code>	Ver · administrar webhooks y sus entregas
<code>reports:read</code> · <code>reports:export</code>	Generar · exportar reportes (Excel, CSV o PDF), guardarlos y programar envíos
<code>domains:read</code> · <code>domains:write</code>	Ver · agregar y verificar dominios

2 Autenticación, IPs permitidas y límites

```
curl https://sendypop.com/api/v1/lists \
  -H "Authorization: Bearer cmpk_live_XXXXXXXXXXXXXXXXXX"
```

- **Límite de pedidos:** 20 por segundo por clave, con ráfagas de hasta 60. Al superarlo se responde `429 rate_limited` con la cabecera `Retry-After` (segundos).
- **Cupos de envío:** los envíos además respetan los cupos de la empresa y, si se configuraron, los de la clave (mensual, diario y por hora).
- **Precalentamiento:** un dominio nuevo arranca con volumen gradual. SentryPOP encola sin perder mensajes y los entrega según el ritmo seguro para la reputación.
- **IP de origen:** SentryPOP toma la IP pública desde la que llega el pedido. Si tu sistema sale por un proxy o NAT, cargá la IP de salida de ese proxy.

3 Respuestas y errores

Los errores devuelven JSON con un código estable y un mensaje legible:

```
{"error": "codigo", "message": "..."} .
```

HTTP	SIGNIFICADO	QUÉ HACER
200 / 201 / 202	OK · creado · aceptado (se procesa en segundo plano)	—
400	Datos inválidos (<code>invalid_body</code> , con <code>details</code> por campo)	Corregir el pedido; no reintentar igual
401	Falta la clave o no es válida (<code>unauthorized</code> , <code>invalid_api_key</code>)	Revisar la clave y sus permisos
403	IP no permitida (<code>ip_not_allowed</code>), otra empresa o falta de verificación (por ejemplo <code>sender_domain_not_verified</code>)	Ver el mensaje; no reintentar sin cambios
404	Recurso inexistente en esta empresa	—
409	Estado incompatible (por ejemplo una campaña ya enviada, o un bloqueo anti-spam). <code>idempotency_request_in_progress</code> : el pedido original con esa <code>Idempotency-Key</code> todavía se procesa	Revisar el estado del recurso · reintentar en unos segundos con la misma clave
422	<code>idempotency_key_reused</code> : la misma <code>Idempotency-Key</code> se usó con otro contenido	Usar una clave nueva por cada envío distinto
429	Límite de pedidos o cupo de envío	Reintentar después de <code>Retry-After</code>
5xx	Error temporal	Reintentar con espera exponencial

4 Envío transaccional

Para emails individuales disparados por tu sistema (confirmaciones, facturas, recuperos). El remitente debe ser un remitente activo de un dominio verificado de la empresa.

```

POST /v1/messages          permiso messages:send
{
  "from": "novedades@tuempresa.com",
  "to": "cliente@ejemplo.com",
  "subject": "Tu pedido 1234 fue confirmado",
  "html": "<p>Hola Ana, tu pedido está en camino.</p>",
  "text": "Hola Ana, tu pedido está en camino.",
  "tags": { "pedido": "1234" }
}

202 Accepted
{ "messageId": "6f1c...", "status": "queued", "tenantId": "a7b3...", "reputation": { ... }
}

```

- `GET /v1/messages/{messageId}` (`messages:read`): estado actual del mensaje. Pasado el plazo de retención de contenido (90 días por defecto), el mensaje conserva estado, fechas, destinatario, remitente, asunto y etiquetas, pero no el cuerpo: `contentPurgedAt` indica cuándo se borró.
- `GET /v1/messages/{messageId}/events` (`events:read` o `messages:read`): entrega, demoras, rebotes, aperturas y clics.
- Las bajas y bloqueos se respetan siempre: un destinatario dado de baja no recibe el mensaje.

Reintentos sin duplicados: Idempotency-Key

Si tu sistema reintentará un envío por un corte de red o un timeout, envía la cabecera `Idempotency-Key` con un valor único por email (por ejemplo el número de pedido). SentryPOP procesa el primer pedido y, ante un reintento con la misma clave y el mismo contenido, devuelve la misma respuesta sin crear otro mensaje.

```

curl -X POST https://sendypop.com/api/v1/messages \
  -H "Authorization: Bearer cmpk_live_XXXXXXXXXXXXXXXXXX" \
  -H "Idempotency-Key: pedido-1234-confirmacion" \
  -H "Content-Type: application/json" \
  -d '{"from":"novedades@tuempresa.com","to":"cliente@ejemplo.com","subject":"Tu pedido 1234","text":"..."}'

```

- La clave vale 24 horas, por API key, y admite hasta 255 caracteres.
- La respuesta repetida trae la cabecera `Idempotent-Replayed: true`.
- La misma clave con otro contenido responde `422 idempotency_key_reused`; mientras el primer pedido se procesa, `409 idempotency_request_in_progress`.
- Las respuestas transitorias (`429` y `5xx`) no se guardan: el reintento con la misma clave se procesa de nuevo.

Modo prueba

Las claves creadas con ambiente **Pruebas** sirven para integrar sin enviar emails reales. Validan todo igual que producción (clave, permisos, IPs, dominio verificado y remitente activo), pero la entrega se simula: no se envía ningún email, no se consume cupo y el resultado llega a tus webhooks con `"sandbox": true` en `data`.

```
202 Accepted
{ "messageId": "9b2e...", "status": "simulated", "outcome": "delivered", "sandbox":
true, "webhookDeliveries": 1 }
```

DESTINATARIO	RESULTADO SIMULADO	WEBHOOK
cualquier dirección	entregado	message.delivered
termina en +bounce (por ejemplo ana+bounce@ejemplo.com)	rebote	message.bounced
termina en +complaint	queja	message.complained

GET /v1/messages/{messageId} y /events también responden para los mensajes simulados. Para pasar a producción, crea una clave de ambiente Producción: no hace falta cambiar nada más.

5 Contactos, grupos y etiquetas

```
POST /v1/contacts contacts:write
{ "email": "ana@ejemplo.com", "firstName": "Ana", "lastName": "Pérez",
  "listId": "<id del grupo>", "attrs": { "ciudad": "Rosario" } }

POST /v1/contacts/import-batch contacts:write (hasta 5.000 por pedido)
{ "listId": "<id del grupo>", "contacts": [ { "email": "a@ejemplo.com", "firstName":
"A" }, ... ] }

POST /v1/contacts/lookup contacts:read (hasta 500 emails)
{ "emails": ["ana@ejemplo.com"] }
```

OPERACIÓN	RUTA
Grupos: listar · crear · renombrar · eliminar	GET/POST /v1/lists · PATCH/DELETE /v1/lists/{listId}
Miembros de un grupo	GET/POST /v1/lists/{listId}/contacts · POST /v1/lists/{listId}/memberships
Editar · eliminar un contacto	PATCH/DELETE /v1/contacts/{contactId}
Grupos y etiquetas de un contacto	PUT /v1/contacts/{contactId}/lists · PUT /v1/contacts/{contactId}/tags
Etiquetas	GET/POST /v1/tags · PATCH/DELETE /v1/tags/{tagId} · POST /v1/contacts/bulk-tags

Las eliminaciones masivas (por ejemplo un grupo grande) se aceptan de inmediato y la limpieza física sigue en segundo plano.

6 Campañas

```
POST /v1/campaigns campaigns:write
{ "name": "Novedades octubre", "subject": "Lo nuevo de octubre",
  "senderEmail": "novedades@tuempresa.com", "listIds": ["<id del grupo>"],
  "contentHtml": "<h1>Hola {{nombre}}</h1>..." }
```

```
POST /v1/campaigns/{campaignId}/send campaigns:send
POST /v1/campaigns/{campaignId}/schedule campaigns:send
{ "scheduledAt": "2026-11-02T13:00:00Z" }
```

- Antes de lanzar, SentryPOP analiza el contenido con SpamAssassin (`POST /v1/campaigns/{id}/spam-check`). Desde 5 puntos pide confirmación (`acknowledgeContentIssues: true`); desde 8, o con un link en una lista negra, el envío se bloquea (`409 campaign_spam_blocked`) con los motivos.
- El lanzamiento se acepta enseguida (`202`) y la preparación sigue en segundo plano. El avance se ve en `GET /v1/campaigns` y en los webhooks.

7 Eventos y automatizaciones

Registrá lo que pasa en tu sistema (una compra, un alta) y SentryPOP dispara las automatizaciones cuyo disparador es **Evento recibido por API** con ese nombre.

```
POST /v1/events events:write
{ "email": "ana@ejemplo.com", "name": "compra_confirmada",
  "properties": { "pedido": "1234", "total": 15990 },
  "idempotencyKey": "pedido-1234" }
```

- `name` : minúsculas, números, guion, guion bajo o punto (hasta 64).
- `idempotencyKey` : el mismo valor nunca dispara dos veces. Usalo siempre que reintentas.
- Si el contacto no existe se crea (salvo `"createContact": false`).
- `GET /v1/events?email=...` (`events:read`): historial de eventos de un contacto.

8 Webhooks

SentryPOP avisa a tu sistema, con un `POST` firmado, cada evento elegido. Se configuran desde Soporte o el panel (**Configuración de API**) o por API (`POST /v1/webhooks` , permiso `webhooks:write`).

EVENTO	CUÁNDO
<code>message.delivered</code>	El servidor de destino aceptó el mensaje
<code>message.deferred</code>	Demora temporal; SentryPOP reintenta
<code>message.bounced</code>	Rebote definitivo (incluye <code>reason</code> , <code>smtpCode</code>)
<code>message.failed</code>	No se pudo entregar (vencido o rechazado)
<code>message.complained</code>	El destinatario lo marcó como spam
<code>message.opened</code> · <code>message.clicked</code>	Apertura · clic (incluye <code>url</code>)
<code>contact.unsubscribed</code>	El destinatario se dio de baja
<code>webhook.test</code>	Prueba enviada con el botón Probar

POST `https://tu-sistema.com/webhooks/sendypop`

Content-Type: `application/json`

SentryPOP-Event: `message.bounced`

SentryPOP-Delivery: `3c9e...` (id de la entrega)

SentryPOP-Signature: `t=1792610000,v1=5f2a...`

```
{ "id": "b81d...", "type": "message.bounced", "createdAt": "2026-10-21T18:20:01.000Z",
  "tenantId": "a7b3...",
  "data": { "messageId": "6f1c...", "email": "cliente@ejemplo.com", "campaignId": null,
    "subject": "Tu pedido 1234", "tags": { "pedido": "1234" },
    "occurredAt": "2026-10-21T18:20:01.000Z",
    "reason": "550 5.1.1 user unknown", "smtpCode": 550 } }
```

Verificar la firma

Calculá un HMAC-SHA256 con el secreto del webhook sobre `<t>.<cuerpo exacto recibido>` y compará en tiempo constante con `v1` . Rechazá firmas con más de 5 minutos de antigüedad.

```
// Node.js
import { createHmac, timingSafeEqual } from "node:crypto";
function validSentryPopSignature(rawBody, header, secret) {
  const parts = Object.fromEntries(header.split(",").map((p) => p.split("=")));
  if (Math.abs(Date.now() / 1000 - Number(parts.t)) > 300) return false;
  const expected = createHmac("sha256",
secret).update(`${parts.t}.${rawBody}`).digest("hex");
  return timingSafeEqual(Buffer.from(expected), Buffer.from(parts.v1 ?? ""));
}
```

```
# Python
import hmac, hashlib, time
def valid_sendypop_signature(raw_body: bytes, header: str, secret: str) -> bool:
    parts = dict(p.split("=", 1) for p in header.split(","))
    if abs(time.time() - int(parts["t"])) > 300: return False
    expected = hmac.new(secret.encode(), f"{parts['t']}".encode() + raw_body,
hashlib.sha256).hexdigest()
    return hmac.compare_digest(expected, parts.get("v1", ""))
```

```
// PHP
function valid_sendypop_signature(string $rawBody, string $header, string $secret):
bool {
    parse_str(str_replace(',', '&', $header), $parts);
    if (abs(time() - (int)$parts['t']) > 300) return false;
    $expected = hash_hmac('sha256', $parts['t'] . '.' . $rawBody, $secret);
    return hash_equals($expected, $parts['v1'] ?? '');
}
```

Reintentos y buenas prácticas

- Respondé **2xx en menos de 10 segundos** y procesá en segundo plano. Cualquier otra respuesta, o ninguna, cuenta como falla.
- Reintentos: 1 min, 5 min, 30 min, 2 h, 6 h, 12 h y 24 h (8 intentos). Después la entrega queda **fallida** y se puede reintentar a mano desde el historial.
- Un webhook que falla de forma continua durante más de 3 días se desactiva solo. Se reactiva desde la configuración.
- Las entregas pueden llegar repetidas o fuera de orden: usá el `id` del evento para no procesarlo dos veces y `occurredAt` para ordenar.
- Sólo `https://` a direcciones públicas. Hasta 10 webhooks por empresa.
- Rotó el secreto con `POST /v1/webhooks/{id}/rotate-secret`: el anterior deja de valer de inmediato.

9 Reportes y dominios

- **Reportes de la empresa** (los mismos seis del panel): `period_summary` (Resumen del período), `campaign` (Reporte de una campaña), `campaign_comparison` (Comparativo de campañas), `deliverability` (Entregabilidad y reputación), `audience` (Audiencia) y `api_sends` (Envíos por API).
- `POST /v1/reports/company/{type}/run` devuelve el reporte en JSON: `kpis` (indicadores) y `tables` (columnas y filas). `POST /v1/reports/company/{type}/export` lo descarga con `"format": "xlsx"`, `"csv"` o `"pdf"`.
- Filtros en `"filters"`: `period` (`today`, `7d`, `30d`, `90d`, `this_month`, `last_month` o `custom` con `from` y `to` en formato `AAAA-MM-DD`, hasta 400 días), `groupId` (carpeta), `sender` (remitente) y `campaignId` (para `campaign`). Los días son de la zona horaria de la empresa. `GET /v1/reports/company` lista carpetas, remitentes y campañas válidos.
- Si la empresa todavía está cargando su historial por primera vez, la respuesta es `202` con `{"preparing": true}` y `Retry-After`.

- Envíos programados: `POST /v1/reports/company/schedules` con `reportType`, `filters`, `format`, `frequency` (`daily`, `weekly` con `weekday` 1 a 7, `monthly` con `monthDay` 1 a 28 o `-1` para el último día), `hour` (0 a 23, hora de la empresa) y `recipients` (hasta 20). Envía el período que acaba de cerrar: el día anterior, los 7 días anteriores o el mes anterior. `GET /v1/reports/company/schedules` muestra los últimos envíos de cada uno; `PATCH` pausa o reanuda y `DELETE` lo borra.
- Siguen disponibles `GET /v1/reports/definitions`, `POST /v1/reports/run` y `POST /v1/reports/export` (reporte por campañas de versiones anteriores).
- `GET /v1/domains`, `GET /v1/domains/{id}/dns` y `POST /v1/domains/{id}/verify` muestran los registros DNS a publicar y su estado. Ver la guía de alta de dominio.

10 Monitoreo de la conexión

En **Configuración de API** (Soporte) y en **API** (panel de la empresa) se ve, por clave y por hora de las últimas 48 horas:

- Pedidos, errores 4xx y 5xx, rechazos por límite y por IP no permitida (con la última IP rechazada).
- Último uso y última IP desde la que se usó cada clave.
- Por webhook: último éxito, última falla, fallas seguidas y el historial de entregas con su respuesta HTTP.

Toda alta, cambio de IPs o permisos, revocación y cambio de webhook queda en la auditoría con quién lo hizo y cuándo.

11 Checklist de puesta en producción

1. Dominio verificado (SPF, DKIM, DMARC y tracking) y remitente activo.
2. Clave de producción con permisos mínimos y, si es posible, IPs permitidas.
3. Secreto guardado en un gestor de secretos; nunca en el repositorio.
4. Manejo de 429 con `Retry-After` y reintentos exponenciales para 5xx.
5. Integración probada de punta a punta con una clave de modo prueba, incluidos rebotes y quejas simulados.
6. Cabecera `Idempotency-Key` en cada envío; `idempotencyKey` en eventos; deduplicación por `id` en webhooks.
7. Webhook probado con el botón **Probar** y firma validada.
8. Prueba real a casillas propias y revisión del monitoreo después de la primera hora.