PROTOCOLO REALNODE

Guía de Integración B2B — React y Next.js (NPM)

Manual de implementación técnica para entornos modernos en JavaScript.

Arquitectura de Componentes (React)
Este documento detalla la integración del protocolo RealNode a través de nuestro paquete oficial NPM. Diseñado específicamente para React y Next.js, este paquete proporciona componentes listos para usar (botones de pago, insignias de cuotas, gestores de dispositivos) para asegurar su túnel de transacciones sin la necesidad de diseñar interfaces complejas.

1 Paso 0: Creación de Cuenta y Recuperación de Claves API

El primer paso es crear una cuenta en el portal de RealNode y recuperar sus claves de acceso. Estas claves identifican de forma segura su aplicación frente a nuestra infraestructura.

1. **Registro:** Visite el portal de administración (`https://app.realnode.emkaylabs.tech/signup`) para crear su cuenta de empresa. Deberá proporcionar el nombre de su empresa, un correo electrónico profesional y una contraseña de administrador.
2. **Recuperación de Claves:** El portal genera un par de claves único para su entorno:
   - **Public Key** (formato: `pk_live_...`): Para ser utilizada en su código frontend de React. Es completamente seguro exponer esta clave en el navegador.
   - **Secret Key** (formato: `sk_live_...`): Para ser utilizada estrictamente en su servidor backend para validar criptográficamente las transacciones. Nunca la comparta ni la empaquete dentro de su código frontend.

2 Paso 1: Instalación del Paquete NPM

Abra su terminal en la raíz de su proyecto React o Next.js y ejecute el siguiente comando:

```
npm install @emkaylabs/realnode-sdk
```

¿Qué hace este comando? Descarga e instala el paquete `@emkaylabs/realnode-sdk` en su proyecto. Este paquete incluye:

- El cliente HTTP principal que se comunica con la API de RealNode (`/verify`, `/consume`, `/check-session`, etc.).
- Componentes de React listos para producción (`RealNodeProvider`, `RealNodeCheckoutButton`, `RealNodeDeviceManager`, `RealNodeQuotaBadge`).
- Utilidades para la gestión de sesiones y el almacenamiento local seguro (ciclo de vida del token de dispositivo).

Una vez instalado, el paquete estará disponible en su directorio `node_modules` y referenciado en su archivo `package.json`.

3 Paso 2: Configuración Global — El Provider

El componente `RealNodeProvider` actúa como el núcleo central. Inicializa el motor de seguridad y expone el contexto del SDK a todos los componentes anidados. Debe ser montado **exactamente una vez**, en lo más alto posible de su árbol de componentes.

¿Dónde colocar este código? Dentro de su archivo de entrada principal: `App.jsx` para Create React App, o `_app.jsx` / `layout.jsx` para Next.js.

```
import { RealNodeProvider } from '@emkaylabs/realnode-sdk/react';

export default function App({ children }) {
  return (
    <RealNodeProvider apiKey="pk_live_SU_CLAVE_PUBLICA">
      {children}
    </RealNodeProvider>
  );
}
```

Secuencia de inicialización interna:

1. El Provider emite automáticamente una solicitud `GET /sdk/config?api_key=pk_live_...` hacia nuestros servidores de borde (edge servers).
2. Los servidores devuelven la configuración activa correspondiente a su plan de suscripción: el nivel de protección y las características habilitadas.
3. En base a esta respuesta, el SDK monta dinámicamente los módulos adecuados sin requerir cambios de código por su parte:
   - **Insight:** Solo análisis conductual pasivo. Sin interferencia visual en la UI.
   - **Sentinel:** Se activa en segundo plano la matriz conductual cinética completa.
   - **Vault:** Se activan la matriz cinética y el modal biométrico (FaceID, TouchID, Windows Hello). El modal se inyecta automáticamente en el DOM.
4. El SDK intenta restaurar de forma silenciosa una sesión existente (si el usuario ya ha completado una verificación anteriormente durante la misma ventana de evento) para evitar avisos biométricos redundantes.

Seguridad de la Public Key
Su clave pública (`pk_live_...`) puede residir de manera segura en su bundle público de React. No otorga ningún acceso a acciones administrativas y sirve estrictamente para identificar su propiedad frente a nuestra API de verificación. Por el contrario, su Secret Key (`sk_live_...`) debe permanecer completamente aislada en la infraestructura de su backend.

4 Paso 3: Asegurando el Botón de Pago (Checkout)

Esta es la única modificación necesaria dentro de su flujo de pago. Simplemente reemplace su botón de compra estándar en HTML con el componente `RealNodeCheckoutButton`.

```
import { RealNodeCheckoutButton } from '@emkaylabs/realnode-sdk/react';

export default function CheckoutPage() {

  const handleSuccess = (proof) => {
    // proof contiene: { idh, device_hash, device_token, quantity, status
        }
```

```
    // Transmita este payload a su backend para finalizar el pedido.
    fetch('/api/su-endpoint-de-checkout', {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify(proof)
    });
  };

  const handleDenied = () => {
    alert("Acceso Denegado: Automatizaci n sospechosa detectada.");
  };

  return (
    <RealNodeCheckoutButton
      quantity={2}
      onSuccess={handleSuccess}
      onDenied={handleDenied}
      className="su-clase-css-existente"
    >
      Confirmar Pedido
    </RealNodeCheckoutButton>
  );
}
```

Flujo de ejecución tras el clic del usuario:

1. Se intercepta el evento de clic. La transacción se detiene temporalmente.
2. Dependiendo del plan activo detectado por el Provider:
   - **Insight:** Se ejecuta una verificación silenciosa en segundo plano. El usuario no nota nada.
   - **Sentinel:** El puntaje de comportamiento se calcula y verifica. El usuario no nota nada.
   - **Vault:** El modal biométrico toma el foco. El usuario verifica su identidad a través de FaceID, TouchID o Passkey. La operación de hardware se resuelve en menos de 1 segundo.
3. Tras una atestación exitosa, se dispara `onSuccess`, inyectando la prueba criptográfica (`proof`).
4. Si la atestación falla o se detecta una firma de bot, se activa `onDenied` y la secuencia de la transacción es abortada.

Importante: La biometría se solicita solo una vez por sesión.
El SDK conserva la prueba de atestación en memoria. Si un usuario ha superado con éxito un desafío biométrico (por ejemplo, durante una compra anterior), no se le volverá a preguntar en transacciones posteriores en la misma página, siempre y cuando la sesión siga siendo válida.

5 Paso 4: Componentes de Interfaz de Usuario (Recomendado)

RealNode proporciona componentes preconstruidos y estilizables que se integran perfectamente en su interfaz de usuario existente, requiriendo cero wireframes o CSS adicional.

5.1. 4.1. Visualización de Cuotas en Tiempo Real

Disponible en todos los planes. Esta insignia muestra el inventario de entradas restantes disponibles para la sesión de usuario actual. El valor numérico se sincroniza automáticamente con nuestros servidores perimetrales a través de una conexión persistente Server-Sent Events (SSE).

```
import { RealNodeQuotaBadge } from '@emkaylabs/realnode-sdk/react';

// Coloque este componente junto al bot n de pago.
// Se renderiza autom ticamente: "Entradas restantes: 3 / 5"
<RealNodeQuotaBadge label="Entradas restantes" />
```

5.2. 4.2. Panel de Usuario — Gestión de Dispositivos

Este componente inyecta un panel completo e interactivo en el área "Mi Cuenta"de sus usuarios, permitiéndoles:

- Auditar la lista de dispositivos de hardware autorizados (teléfonos inteligentes, portátiles) vinculados a su identidad.
- Revocar remotamente el acceso a un dispositivo perdido o robado.
- Desvincular asociaciones de hardware antes de la reventa del dispositivo.
- Transferir el acceso de la cuenta a un usuario diferente de forma segura.
- Recuperar el acceso utilizando un Código Maestro de emergencia proporcionado por su equipo administrativo.

```
import { RealNodeDeviceManager } from '@emkaylabs/realnode-sdk/react';

export default function MyAccount() {
  return (
    <div>
      <h2>Configuraci n de Seguridad</h2>
      {/* Genera autom ticamente toda la interfaz de gesti n. */}
      {/* No se requiere l gica adicional en HTML o CSS.      */}
      <RealNodeDeviceManager apiKey="pk_live_SU_CLAVE_PUBLICA" />
    </div>
  );
}
```

6 Paso 5: Validación Final en el Lado del Servidor (Backend)

La prueba criptográfica (`proof`) emitida por el navegador debe ser validada por su infraestructura backend antes de completar el pedido. Este es un punto de control de seguridad obligatorio: su servidor realiza una llamada autenticada a la API de RealNode utilizando su **Secret Key** para garantizar que la prueba no ha sido manipulada y que la cuota de inventario sigue siendo válida.

```
// Ejemplo: Backend Node.js / Express
app.post('/api/su-endpoint-de-checkout', async (req, res) => {
  const proof = req.body;
  // proof = { idh, device_hash, device_token, quantity, status }

  const response = await fetch('https://api.emkaylabs.tech/consume', {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json',
      'X-RealNode-API-Key':    'pk_live_SU_CLAVE_PUBLICA',
      'X-RealNode-Secret-Key': 'sk_live_SU_SECRET_KEY'
    },
    body: JSON.stringify({
      quantity:     proof.quantity,
      idh:          proof.idh,
      device_hash:  proof.device_hash,
      device_token: proof.device_token,
      client_id:    'su-id-de-sitio-web',
      // event_id: 'CONCIERTO-2024-MADRID' // Opcional: para cuotas
         limitadas
    })
  });

  if (response.ok) {
    // RealNode valid  con  xito  la atestaci n. Proceda con el pago.
    await processPayment();
    res.status(200).json({ success: true });
  } else {
    // Cuota excedida, o firma criptogr fica inv lida.
    res.status(403).json({ error: "Transacci n denegada por RealNode
       Engine." });
  }
});
```

Respuestas Estándar de la API `/consume`:

Código HTTP	Significado	Acción Recomendada
200	Transacción validada, cuota deducida	Finalizar el pedido
403	Firma inválida o bot detectado	Rechazar la transacción
429	Cuota agotada para este IDH de usuario	Mostrar un mensaje de espera
401	Credenciales de clave API inválidas	Verificar sus claves en el panel

Soporte B2B Dedicado
Como cliente empresarial de RealNode, usted recibe soporte de integración prioritario. Para cualquier consulta sobre la arquitectura, póngase en contacto con nuestro equipo de ingeniería en **security@emkaylabs.tech**. Garantizamos una respuesta técnica en menos de 24 horas laborables.

7 Apéndice: Configuraciones Avanzadas

7.1. Gestión de Cuotas por Evento (Plan Vault)

Para lanzamientos altamente controlados (entradas de edición limitada, lanzamientos exclusivos de zapatillas, etc.), puede adjuntar un identificador de evento interno al payload `/consume` (ver Paso 5).

¿Cómo funciona?

- No es necesaria ninguna configuración previa en el panel de RealNode.
- Simplemente agregue su propia cadena arbitraria (por ejemplo, `CONCIERTO-2024-MADRID`).
- El RealNode Engine aislará y rastreará automáticamente las cuotas de hardware para ese usuario específico de manera estricta dentro del contexto de ese evento.