

SlicePay gateway security recheck

Full report · slicechain.io/security/

Date: 1 October 2026 **Scope:** Pay widget and gateway invoice controls from the September 2026 audit, re-read in the current code, including fiat currency conversion. **Method:** Source review of the gateway, checkout, webhook, and merchant settings code. This is not a penetration test of the whole platform.

This document does not claim that SlicePay has zero vulnerabilities. It records which checkout controls are in place, and which residual issues remain.

Checkout is safe to use when invoices are created on your server

Shoppers pay a hosted invoice. The USD amount is stored when that invoice is created. A later payment link cannot change it. Merchant API keys are required for normal orders, for fulfillment, and for private invoice metadata. Payment notices to your store are signed.

Use `data-invoice-id` in the embed. Create the invoice on your server. Do not put the API key in the store HTML.

Controls that hold

	orderId\	status\

Fiat conversion did not reopen the amount lock. The shopper still cannot change the USD charge from the checkout URL.

Residual issues

These are still true in the code that was re-read.

- 1 **Medium. API key in the embed.** `embed.js` can take `data-api-key` and call `create-invoice` from the browser. Anyone who can view that page can copy the key. Keep invoice creation on the merchant server.
- 1 **Medium. Static QR invoices skip the API key.** If `orderId` starts with `STATIC-`, `create-invoice` does not require a key. Anyone who knows the public merchant id can open those invoices and bind a static USDC amount for that merchant.
- 1 **Low. Payment state is readable with the invoice id.** `GET /payment-status`, `GET /invoice`, and `GET /receipt` return the locked order id, USD amount, and status without a key. Paid invoices also return the payer wallet and transaction signature. Checkout needs this public read. The invoice id has to stay unguessable and out of public logs.
- 1 **Low. Two key comparisons.** `assertMerchantApiKey` uses a timing-safe compare. `loadBusiness` uses a plain inequality when a key is supplied. Those checks should match. Practical exploitation over the network is unlikely.

Webhook replay protection is only half done on the SlicePay side. We send a unique event id. The store must verify `X-SlicePay-Signature` and ignore a repeated event id.

How the controls fit together

Create. The merchant server calls `POST /api/gateway/create-invoice` with the API key, order id, and either `amountUsd` or `amount` plus currency. The server writes `amountUsd`. Static QR is the exception: `STATIC-` order ids skip the key.

Pay. The shopper opens the hosted checkout with `invoiceId` only. Quotes and `start-payment` use the stored USD amount.

Confirm. `GET /payment-status/:invoiceId` returns the locked amount and status. Full metadata requires the API key. `POST /invoice/:invoiceId/redeem` is the fulfillment lock and requires the key.

Notify. On the pending-to-paid transition, `invoice.paid` is posted with an HMAC over the canonical string and a unique event id.

Files reviewed

- `backend/src/services/gatewayInvoiceSecurity.js`
- `backend/src/services/gatewayFulfillment.js`
- `backend/src/services/webhookDispatchService.js`
- `backend/src/routes/gateway.js`
- `backend/src/utls/businessSettingsSecurity.js`
- `slice-pay-checkout-next-js/lib/gatewayCheckout.js`

Not in this recheck

Solana program logic, wallet key storage, admin auth, PAJ, employee QR tokens, nginx, MongoDB, and the host were not re-tested. No exploit attempt was run against production hosts.

Practical rules

- Keep the API key on the merchant server.
- Use `data-invoice-id` in the widget.
- Treat `STATIC-` invoice creation as a public action tied to the merchant id.

- Verify webhook signatures and reject reused event ids.
- Do not publish invoice ids in analytics, referrer logs, or support tickets if payment state should stay private.