NWC- Nostr wallet Connect

lightning wallet integration with Nostr

BitDevs Abuja
Fairtrade Business Complex
28-06-2025

Shytypes

TOC

- Overview of NIP-47: Nostr Wallet Connect (NWC)
- Purpose: Standard protocol for connecting Nostr clients to remote Lightning wallets
- Key Roles:
  - Client: Nostr app wanting Lightning wallet access
  - Wallet Service: App/server with wallet APIs, always online
  - User: Person connecting their wallet via client
- Communication: E2E-encrypted direct messages over Nostr relays using specific event kinds
- Event Kinds:
  - 13194: Info event (wallet capabilities)
  - 23194: Request (e.g., pay, get balance)
  - 23195: Response (to requests)
  - 23196: Notification (wallet events)
- Connection Flow:
  - User gets a connection URI (with relay URL, secret, wallet pubkey)
  - Client uses URI to communicate securely with wallet service
- Main Commands Supported:
  - `pay_invoice`, `multi_pay_invoice`
  - `pay_keysend`, `multi_pay_keysend`
  - `make_invoice`, `lookup_invoice`
  - `list_transactions`, `get_balance`, `get_info`
- Notifications: `payment_received` & `payment_sent`, sent as encrypted events
- Security: Uses unique keys per connection, improves privacy and compartmentalization
- Use Cases: Non-custodial and custodial wallets, clients on any platform
- Example connection string and event structures included for implementers

NWC (Nostr Wallet Connect) - Technical Overview

- Overview of NIP-47: Nostr Wallet Connect (NWC)
  - Standard protocol for connecting Nostr clients to remote Lightning wallets.
  - Enables wallet actions from any Nostr-compatible app.
  - Supports both custodial and non-custodial wallet services.
- Purpose
  - Grants secure Lightning wallet access via encrypted Nostr messages.
  - Provides a unified protocol for wallet interaction.
- Key Roles
  - Client: Nostr app for wallet access (mobile, desktop, web).
  - Wallet Service: Server or device with wallet APIs, always online (LND...)
  - User: Person linking their wallet through the client.
- Communication
  - End-to-end encrypted (NIP04) direct messages over Nostr relays.
  - Relays see only metadata, not message content.

Event Kinds

- 13194: Info event (wallet capabilities).

```
{
  "id": "df467db0a9f9ec77ffe6f561811714ccaa2e26051c20f58f33c3d66d6c2b4d1c",
  "pubkey": "c04ccd5c82fc1ea3499b9c6a5c0a7ab627fbe00a0116110d4c750faeaecba1e2",
  "created_at": 1713883677,
  "kind": 13194,
  "tags": [
    [
      "notifications",
      "payment_received payment_sent"
    ]
  ],
  "content": "pay_invoice pay_keysend get_balance get_info make_invoice lookup_invoice
list_transactions multi_pay_invoice multi_pay_keysend sign_message notifications",
  "sig":
"31f57b369459b5306a5353aa9e03be7fbde169bc881c3233625605dd12f53548179def16b9fe1137e6465d7e4d5bb27ce81fd6
e75908c46b06269f4233c845d8"
}
```

- 23194: Request event (client → wallet, e.g., pay invoice).

```
{
  "id": "e.g. unique-event-id",
  "pubkey": "c04ccd5c82fc1ea3499b9c6a5c0a7ab627fbe00a0116110d4c750faeaecba1e2",
  "created_at": 1713883677,
  "kind": 23194,
  "tags": [],
  "content": "{\"method\":\"pay_invoice\",\"params\":{\"invoice\":\"lnbc1...\"}}",
  "sig":
"31f57b369459b5306a5353aa9e03be7fbde169bc881c3233625605dd12f53548179def16b9fe1137e6465d7e4d5bb27ce81fd6
e75908c46b06269f4233c845d8"
}
```

Event Kinds (continued)

- `23195`: Response event (wallet → client, e.g., result).

```
{
  "id": "e.g. unique-event-id",
  "pubkey": "c04ccd5c82fc1ea3499b9c6a5c0a7ab627fbe00a0116110d4c750faeaecba1e2",
  "created_at": 1713883677,
  "kind": 23195,
  "tags": [],
  "content": "{\"result\":{\"preimage\":\"abcdef...\"},\"error\":null}",
  "sig":
"31f57b369459b5306a5353aa9e03be7fbde169bc881c3233625605dd12f53548179def16b9fe1137e6465d7e4d5bb27ce81fd6
e75908c46b06269f4233c845d8"
}
```

- `23196`: Notification event (wallet → client, e.g., payment received).

```
{
  "id": "e.g. unique-event-id",
  "pubkey": "<user-pubkey>",
  "created_at": 1713883677,
  "kind": 23196,
  "tags": [],
  "content": "{\"notification\":{\"type\":\"payment_received\",\"amount\":10000}}",
  "sig": "e.g. signature"
}
```

Connection Flow (Client 🔌 Lightning Wallet Service)

- User gets a connection URI from wallet service.
- Client scans QR or pastes URI, stores connection details.
- Encryption/signing uses the connection's secret.
- Example URI:

```
nostr+walletconnect://<wallet-pubkey>?relay=<relay-url>&secret=<secret>&lud16=<ln-address>
```

- URL Breakdown:
  - `nostr+walletconnect://`: The Nostr wallet connect URI scheme
  - `<wallet-pubkey>`: Wallet's public key.
  - `<relay-url>`: Nostr relay URL for communication.
  - `<secret>`: Unique secret for, authentication, encryption/signing.
  - `<ln-address>`: The Lightning address for payments in LUD-16 format

> ⓘ Note
>
> * Each connection uses a unique secret per client.
> * Revocable, improves privacy, no master key exposure.
> * Any relay can be used; custodial services may offer their own relay.

Main Commands Supported

- pay_invoice: Pay a Lightning invoice.

```
{
  "method": "pay_invoice",
  "params": { "invoice": "lnbc50n1...", "amount": 123 }
}
```

- multi_pay_invoice: Pay multiple invoices in batch.

```
{
  "method": "multi_pay_invoice",
  "params": { "invoices": [ { "id":"abc", "invoice": "lnbc1...", "amount": 123 }, { "id":"def",
"invoice": "lnbc50n1..." } ] }
}
```

- pay_keysend: Keysend payment (no invoice).

```
{
  "method": "pay_keysend",
  "params": { "amount": 123, "pubkey": "03...", "preimage": "...", "tlv_records": [...] }
}
```

- multi_pay_keysend: Batch keysend payments.

```
{
  "method": "multi_pay_keysend",
  "params": { "keysends": [ { "id": "id1", "pubkey": "03...", "amount": 123 } ] }
}
```

- make_invoice: Create a Lightning invoice.

- Use Cases
  - Any Nostr client can connect to a Lightning wallet for payments, balance, invoices.
  - Works with both personal nodes and custodial services.
- Implementation Details
  - All request/response content is JSON, encrypted with NIP04.
  - Event tags define sender/receiver, support optional expiry.
  - Example info event:

```json
{
  "kind": 13194,
  "content": "pay_invoice get_balance notifications",
  "tags": [
    ["notifications", "payment_received payment_sent"]
  ]
}
```

Error Codes

- Standard errors: `RATE_LIMITED`, `INSUFFICIENT_BALANCE`, `UNAUTHORIZED`, `PAYMENT_FAILED`, etc.

```
{
  "error": { "code": "INSUFFICIENT_BALANCE", "message": "Not enough funds" }
}
```

Example pay invoice flow

1. User scans QR code from wallet service or follows `nostr+walletconnect://` link.
2. Client sends an event to wallet service (`kind 23194`) with a `pay_invoice` request. Payload is encrypted with secret from connection string.
3. Wallet service verifies key, decrypts payload, performs payment.
4. Wallet service sends response event (`kind 23195`) with payment result or error.

Example request

```
{
  "method": "pay_invoice",
  "params": { "invoice": "lnbc50n1..." }
}
```

Example response

```
{
  "result_type": "pay_invoice",
  "result": { "preimage": "0123456789abcdef..." },
  "error": null
}
```

Example error response

```
{
  "result_type": "pay_invoice",
  "result": null,
  "error": { "code": "PAYMENT_FAILED", "message": "Payment failed due to insufficient capacity." }
}
```

Thank you

Nostr wallet connect gives independent creators the power to monetize their content without relying on centralized platforms