

# Private Payments in Practice

Integrating Zcash from  
Scratch to Production

ETHBoulder Feb 2026

Mylo Bennett aka ReadyMouse







# Motivation: Ethics in Practice

Maybe it's crazy, but I don't want the government (or anyone) tracking my purchases

## Banana Betting

Zcash-Native Prediction Market

The screenshot displays the Banana Betting website interface. At the top is a navigation bar with links: Home, Find Bets, Make Event, My Bets, Settle Bets, and a user profile for 'test2'. The main content area features a bet titled 'Tiktok likes: Man-nana Dadbob Cheerleaders dancing to Shania Twain'. Below the title, it asks 'How many likes will the Tiktok video of Savannah Banana's Man-nana Dadbob Cheerleaders dancing to "Man! I Feel Like a Woman!" by Shania Twain get in the next week?' and provides a TikTok link. The interface is divided into several sections: 'Betting Information' showing a total pool of 0.0000 ZEC and a bet range of 0.000001 - 5 ZEC; 'Choose Your Outcome' with two options (A: 300-350k Likes and B: >350k Likes), both at 0% odds; 'Event Stats' showing 0 players and an 'Open' status; and 'Event Timeline' with start, end, and settlement times. A 'Bet Amount' section at the bottom allows users to place a bet within the same range.

Full node, from scratch with Zcash's SDK

## KarenBot.Com

Customer Service Advocate



BTCPay + ZEC Plugin



# Comment on Electric Coin Company



In January 2026, the entire Electric Coin Company development team resigned following a governance dispute with Bootstrap, the nonprofit that oversees ECC. The core of the conflict centers on differing views between ECC leadership and the Bootstrap board over how to structure funding and commercialization, particularly around the Zashi wallet, within the constraints of nonprofit law. The former ECC team has since formed a new company and plans to continue building on Zcash.

The Zcash *protocol* is open source, permissionless,  
and completely unaffected

Today's focus is on the technology



# Mainstream money...



		Square
Privacy	Shielded TX	No
	Native ZEC	No
Control	Custodial	Yes
	Run Own Node	No
	Fees	1%
Complexity	Set Up	Low
	Readiness	Mature
Multi-coin		BTC only



# Now accepting private money...



		Square	Zcash SDK	Zgo	BTCPay	NOWPayments
Privacy	Shielded TX	No	Yes	Yes	Yes	No
	Native ZEC	No	Yes	Yes	Yes	Yes
Control	Custodial	Yes	No	No	No	Optional
	Run Own Node	No	Yes	No	Optional	No
	Fees	1%	-	\$22/mo	\$10-30/mo	0.5%
Complexity	Set Up	Low	High	Moderate	Moderate	Low
	Readiness	Mature	Mature	Early	Mature	Mature
Multi-coin		BTC only	No	No	XMR, DASH,...	XMR, DASH,...

No clear winner. It's all about trade-offs.



		Square	Zcash SDK	Zgo	BTCPay	NOWPayments
Privacy	Shielded TX	No	Yes	Yes	Yes	No
	Native ZEC	No	Yes	Yes	Yes	Yes
Control	Custodial	Yes	No	No	No	Optional
	Run Own Node	No	Yes	No	Optional	No
	Fees	1%	-	\$22/mo	\$10-30/mo	0.5%
Complexity	Set Up	Low	High	Moderate	Moderate	Low
	Readiness	Mature	Mature	Early	Mature	Mature
Multi-coin		BTC only	No	No	XMR, DASH,...	XMR, DASH,...



# Wallet-to-Wallet TX Development Pipeline



## ETH-Based

1. Develop w. Hardhat's blockchain simulator
2. Test on fake blockchain
3. Find a faucet for testnet tokens
4. Connect to testnet node
5. Verify smart contract works on testnet
6. Publish on mainnet





# Wallet-to-Wallet TX Development Pipeline



## ETH-Based

## ZEC-Based Development

1. Develop w. Hardhat's blockchain simulator  
→ I can't find a ZEC blockchain simulator
2. Test on fake blockchain
3. Find a faucet for testnet tokens → Testnet.zecfaucet → Wait, what wallet supports TAZ? Zingo
4. Connect to testnet node → Tatum.io RPC → READ ONLY, not helpful for a private blockchain
5. Verify smart contract works on testnet → No Smart Contracts, no programable money
6. Publish on mainnet → Own Code + Own Node → Create wallets + receive addresses

Lack of smart contracts and mature development infrastructure for zcash





# Wallet-to-Wallet TX Development Pipeline



## ETH-Based

## ZEC-Based Development

1. Develop w. Hardhat's blockchain simulator  
I can't find a ZEC blockchain simulator
2. Test on fake blockchain
3. Find a faucet for testnet tokens → Testnet.zecfaucet → Wait, what wallet supports TAZ?
4. Connect to testnet node → Tatum.io RPC → READ ONLY, not helpful for a private blockchain
5. Verify smart contract works on testnet → No Smart Contracts, no programable money
6. Publish on mainnet → Own Code + Own Node → Create wallets + receive addresses

Lack of smart contracts and mature development infrastructure for zcash



# Own Code + Own Node



ZECHUB 2025 Hackathon 3rd Place Winner: Banana Betting

## Frontend

### Wallet Management

Current Balance

0.0001 ZEC

Shielded Address (Private)

.....

Transparent Address (Public)

.....

● Wallet Connected

Refresh

Not covered in this talk

## Backend

```
def z_getaddressforaccount(account: int, receiver_type=None, diversifier_index: int=None):
    params = [account]
    if receiver_type is not None:
        params.append(receiver_type)
    if diversifier_index is not None:
        params.append(diversifier_index)
    try:
        # RPC request payload
        payload = {
            "jsonrpc": "1.0",
            "id": "z_getaddressforaccount",
            "method": "z_getaddressforaccount",
            "params": params
        }

        # Make the request to the Zcash node
        response = requests.post(ZCASH_RPC_URL, json=payload, auth=(ZCASH_RPC_USER, ZCASH_RPC_PASSWORD))

        # Handle Zcash node response
        if response.status_code != 200:
            print(response.json())
            raise HTTPException(status_code=500, detail="Failed to connect to Zcash node")

        # Parse response
        validation_data = response.json()
        return validation_data['result']['address']
    except Exception as e:
        raise HTTPException(status_code=500, detail=str(e))
```

- Generate addresses
- Create transactions
- Get balances

## Full Node

```
1 # Simple 'zcashd' Docker Compose configuration.
2 #
3 # THIS DAEMON IS BEING DEPRECATED AND SHOULD NOT BE USED FOR NEW
4 # SERVERS. ONLY USE THIS IF YOU KNOW YOU HAVE A SPECIFIC NEED FOR
5 # THE OLDER IMPLEMENTATION OF A ZCASH FULL NODE.
6 #
7 # Instead of this configuration, you should use 'compose.yaml', the
8 # new default with the 'zebrad' service configuration.
9 ---
10 services:
11     zcashd:
12         image: electriccoinco/zcashd:v6.3.0
13         restart: unless-stopped
14         configs:
15             - source: zcashd_conf
16               target: /etc/zcashd/zcashd.conf
17         volumes:
18             - ./data/zcashd-cache:/srv/zcashd/.zcache
19             - ./data/zcashd-params:/srv/zcashd/.zcache-params
20         ports:
21             - 0.0.0.0:8232:8232
22             - 0.0.0.0:8233:8233
23         command: >
24             -conf=/etc/zcashd/zcashd.conf
25
26 configs:
27     zcashd_conf:
28         file: ./configs/zcashd.conf
```

- Docker + server
- Interacts with blockchain
- Read/write capabilities

Backend: "Smart contracts" for your own wallet, but it's python



# Own Code + Own Node



ZECHUB 2025 Hackathon 3rd Place Winner: Banana Betting

## Frontend

### Wallet Management

Current Balance

0.0001 ZEC

Shielded Address (Private)

.....

Transparent Address (Public)

.....

● Wallet Connected

Refresh

Not covered in this talk

## Backend

```
def z_getaddressforaccount(account: int, receiver_type=None, diversifier_index: int=None):
    params = [account]
    if receiver_type is not None:
        params.append(receiver_type)
    if diversifier_index is not None:
        params.append(diversifier_index)
    try:
        # RPC request payload
        payload = {
            "jsonrpc": "1.0",
            "id": "z_getaddressforaccount",
            "method": "z_getaddressforaccount",
            "params": params
        }

        # Make the request to the Zcash node
        response = requests.post(ZCASH_RPC_URL, json=payload, auth=(ZCASH_RPC_USER, ZCASH_RPC_PASSWORD))

        # Handle Zcash node response
        if response.status_code != 200:
            print(response.json())
            raise HTTPException(status_code=500, detail="Failed to connect to Zcash node")

        # Parse response
        validation_data = response.json()
        return validation_data['result']['address']

    except Exception as e:
        raise HTTPException(status_code=500, detail=str(e))
```

- Generate addresses
- Create transactions
- Get balances

## Full Node

```
1 # Simple 'zcashd' Docker Compose configuration.
2 #
3 # THIS DAEMON IS BEING DEPRECATED AND SHOULD NOT BE USED FOR NEW
4 # SERVERS. ONLY USE THIS IF YOU KNOW YOU HAVE A SPECIFIC NEED FOR
5 # THE OLDER IMPLEMENTATION OF A ZCASH FULL NODE.
6 #
7 # Instead of this configuration, you should use 'compose.yaml', the
8 # new default with the 'zebrad' service configuration.
9 ---
10 services:
11     zcashd:
12         image: electriccoinco/zcashd:v6.3.0
13         restart: unless-stopped
14         configs:
15             - source: zcashd_conf
16               target: /etc/zcashd/zcashd.conf
17         volumes:
18             - ../data/zcashd-cache:/srv/zcashd/.zcache
19             - ../data/zcashd-params:/srv/zcashd/.zcache-params
20         ports:
21             - 0.0.0.0:8232:8232
22             - 0.0.0.0:8233:8233
23         command: >
24             -conf=/etc/zcashd/zcashd.conf
25
26 configs:
27     zcashd_conf:
28         file: ./configs/zcashd.conf
```

- Docker + server
- Interacts with blockchain
- Read/write capabilities

Backend: "Smart contracts" for your own wallet, but it's python



# Zcash Backend – To the Docs!



The documentation for the RPC call

## z\_getbalanceforaccount - Zcash 6.2.0 RPC

```
z_getbalanceforaccount account ( minconf asOfHeight )
```

Returns the account's spendable balance for each value pool ("transparent", "sapling", and "orchard").

Arguments:

1. account (numeric) The account number.
2. minconf (numeric, optional, default=1) Only include transactions confirmed at least this many times.
3. asOfHeight (numeric, optional, default=-1) Execute the query as if it were run when the blockchain was at the height specified by

### Major functions needed:

- Z\_get\_new\_account: make an account (wallet)
- Z\_getAddressForAccount: make receive address(es)
- Z\_send\_many: send transaction(s)
- Z\_getBalance
- Z\_listReceivedByAddress: get all transactions for an address

The implementation for the RPC call

```
payload = {  
  "jsonrpc": "1.0",  
  "id": "z_getbalanceforaccount",  
  "method": "z_getbalanceforaccount",  
  "params": [account or 0, minconf]  
}
```

*# Make the request to the Zcash node*

```
response = requests.post(  
  ZCASH_RPC_URL,  
  json=payload,  
  auth=(ZCASH_RPC_USER, ZCASH_RPC_PASSWORD)  
)
```





# Zcash Backend – Examples



Woah! That's so cool...I want to write the backend of my production system using Python someone wrote for a hackathon!!

<https://github.com/ReadyMouse/private-crypto-processors>

RPC Function Call definitions:

- Zcash\_wallet.py
- Zcash\_utils.py

Example: main\_transactions.py

Now we have code to...

- Make unique receive addresses for customers
- Check addresses balances for correct amount deposited
- Withdraw ZEC to outside wallet(s) using z\_sendmany

-> We need the node to test

```
zcash_sdk > zcash_utils.py
1 import requests
2 from fastapi import Depends, FastAPI, HTTPException, status, Query
3 from zcash_sdk import main_transactions.py
4
5 def
6     1 from fastapi import Depends, FastAPI, HTTPException, status
7     2 from fastapi.middleware.cors import CORSMiddleware
8     3 from sqlalchemy.orm import Session
9     4 fr zcash_sdk > zcash_wallet.py
10    5 fr
11    6 fr 1 import requests
12    7 fr 2 from fastapi import Depends, FastAPI, HTTPException, status, Query
13    8 fr 3 from ..zcash_mod import ZCASH_RPC_URL, ZCASH_RPC_USER, ZCASH_RPC_PASSWORD, DISABLE_ZCASH_NODE
14    9 fr 4 from decimal import Decimal
15   10 fr 5 import simplejson
16   11 fr 6
17   12 # Mock balances for development (user_id -> balance)
18   13 _mock_user_balances = {}
19   14 _mock_pool_balance = 1000.0
20   15
21   16 def backupwallet(destination: str):
22   17     try:
23   18         # RPC request payload
24   19         payload = {
25   20             "jsonrpc": "1.0",
26   21             "id": "backupwallet",
27   22             "method": "backupwallet",
28   23             "params": [destination]
29   24         }
30   25
31   26         # Make the request to the Zcash node
32   27         response = requests.post(ZCASH_RPC_URL, json=payload, auth=(ZCASH_RPC_USER, ZCASH_RPC_PASSWORD))
33   28
34   29         # Handle Zcash node response
35   30         if response.status_code != 200:
36   31             print(response.json())
37   32             raise HTTPException(status_code=500, detail="Failed to connect to Zcash node")
38   33
39   34         # Parse response
40   35         validation_data = response.json()
41   36         print(validation_data)
42   37         return validation_data['result']
43   38
44   39     except Exception as e:
45   40         raise HTTPException(status_code=500, detail=str(e))
46   41
47   42 def z_get_new_account():
48   43     try:
```



# Own Code + Own Node



## Frontend

### Wallet Management

Current Balance

0.0001 ZEC

Shielded Address (Private)

.....

Transparent Address (Public)

.....

● Wallet Connected

Refresh

Not covered in this talk

## Backend

```
def z_getaddressforaccount(account: int, receiver_type=None, diversifier_index: int=None):
    params = [account]
    if receiver_type is not None:
        params.append(receiver_type)
    if diversifier_index is not None:
        params.append(diversifier_index)
    try:
        # RPC request payload
        payload = {
            "jsonrpc": "1.0",
            "id": "z_getaddressforaccount",
            "method": "z_getaddressforaccount",
            "params": params
        }

        # Make the request to the Zcash node
        response = requests.post(ZCASH_RPC_URL, json=payload, auth=(ZCASH_RPC_USER, ZCASH_RPC_PASSWORD))

        # Handle Zcash node response
        if response.status_code != 200:
            print(response.json())
            raise HTTPException(status_code=500, detail="Failed to connect to Zcash node")

        # Parse response
        validation_data = response.json()
        return validation_data['result']['address']
    except Exception as e:
        raise HTTPException(status_code=500, detail=str(e))
```

- Generate addresses
- Create transactions
- Get balances

## Full Node

```
1 # Simple 'zcashd' Docker Compose configuration.
2 #
3 # THIS DAEMON IS BEING DEPRECATED AND SHOULD NOT BE USED FOR NEW
4 # SERVERS. ONLY USE THIS IF YOU KNOW YOU HAVE A SPECIFIC NEED FOR
5 # THE OLDER IMPLEMENTATION OF A ZCASH FULL NODE.
6 #
7 # Instead of this configuration, you should use 'compose.yaml', the
8 # new default with the 'zebrad' service configuration.
9 ---
10 services:
11     zcashd:
12         image: electriccoinco/zcashd:v6.3.0
13         restart: unless-stopped
14         configs:
15             - source: zcashd_conf
16               target: /etc/zcashd/zcashd.conf
17         volumes:
18             - ./data/zcashd-cache:/srv/zcashd/.zcash
19             - ./data/zcashd-params:/srv/zcashd/.zcash-params
20         ports:
21             - 0.0.0.0:8232:8232
22             - 0.0.0.0:8233:8233
23         command: >
24             -conf=/etc/zcashd/zcashd.conf
25
26 configs:
27     zcashd_conf:
28         file: ./configs/zcashd.conf
```

- Docker + server
- Interacts with blockchain
- Read/write capabilities

Full Node: Decentralized database of all user transactions (transparent + shielded)



# Zcash Node – To the Docs!



## Node Requirements:

- ZEC-accepting server provider
- 300-400GB Blockchain storage
- 4-6GB RAM
- ZECsy, DevOps girlfriend who can run your infrastructure





# Zcash Node – To the Docs!



## Node Requirements:

- ZEC-accepting server provider
- 300-400GB Blockchain storage
- 4-6GB RAM
- ZECsy, DevOps girlfriend who can run your infrastructure



## Documentation:

- <https://github.com/zecrocks/zcash-stack/>
- Workshop on hosting read-only, lightwallet nodes

## Process:

- Create an account (wallet)
- Save your seed phrase/keys
- Start making addresses to accept TXs

```
# Zcashd configuration file.
i-am-aware-zcashd-will-be-replaced-
by-zegrad-and-zallet-in-2025=1
txindex=1
experimentalfeatures=1
lightwalletd=1
disablewallet=0
rpcbind=0.0.0.0
rpccallowip=0.0.0.0/0

rpcuser=rpcuser
rpcpassword=PASSWORD
```

## Example Docker File for Node Instance

```
services:
  zcashd:
    image: electriccoinco/zcashd:v6.3.0
    restart: unless-stopped
    configs:
      - source: zcashd_conf
        target: /etc/zcashd/zcashd.conf
    volumes:
      - ./data/zcashd-cache:/srv/zcashd/.zcash
      - ./data/zcashd-params:/srv/zcashd/.zcash-params
    ports:
      - 0.0.0.0:8232:8232
      - 0.0.0.0:8233:8233
    command: >
      -conf=/etc/zcashd/zcashd.conf

configs:
  zcashd_conf:
    file: ./configs/zcashd.conf
```





# End to End Testing



Yay! Let's check the blockchain.

Zcash Block Explorer

transaction / block / address

Mempool Blocks Tools

Details for the Zcash Transaction ID 0b1cf9e8d4536c68c52246e334504c296c5658bb07a45674cb9eb292d45dc909

Confirmations	Time (UTC)	Tx Type
1	2025-09-29 16:37:55 (17 seconds ago)	Shielded
Block Id	Public Inputs / Outputs	Shielded Inputs / Outputs
3082024	0 / 0	0 / 0
JoinSplits ?	Transferred from shielded pool	Overwintered?
No	0.0001 ZEC	true
Size (bytes)	Version	Locktime
9165	5	0
Transaction fee	Raw TX	Orchard Action transfers
0.0001 ZEC	JSON	2

PublicTransfers

Inputs (0) → Outputs (0)

Shielded

Zashi Deposit

Wallet Management

Current Balance

0.0001 ZEC

Shielded Address

Transparent Address

Wallet Connected

Receiving

0.000582ZEC

Transaction Details

Transaction ID: b5157...cd4fd

Completed: Sep 29, 2025 16:14

Confirmation it occurred, no details.

ZEC-native, private, non-custodial payment processing but with high technical overhead



		Square	Zcash SDK	Zgo	BTCPay	NOWPayments
Privacy	Shielded TX	No	Yes	Yes	Yes	No
	Native ZEC	No	Yes	Yes	Yes	Yes
Control	Custodial	Yes	No	No	No	Optional
	Run Own Node	No	Yes	No	Optional	No
	Fees	1%	-	\$22/mo	\$10-30/mo	0.5%
Complexity	Set Up	Low	High	Moderate	Moderate	Low
	Readiness	Mature	Mature	Early	Mature	Mature
Multi-coin		BTC only	No	No	XMR, DASH,...	XMR, DASH,...

ZEC-native, private, non-custodial payment processing but with high technical overhead



		Square	Zcash SDK	Zgo	BTCPay	NOWPayments
Privacy	Shielded TX	No	Yes	Yes	Yes	No
	Native ZEC	No	Yes	Yes	Yes	Yes
Control	Custodial	Yes	No	No	No	Optional
	Run Own Node	No	Yes	No	Optional	No
	Fees	1%	-	\$22/mo	\$10-30/mo	0.5%
Complexity	Set Up	Low	High	Moderate	Moderate	Low
	Readiness	Mature	Mature	Early	Mature	Mature
Multi-coin		BTC only	No	No	XMR, DASH,...	XMR, DASH,...



# Zgo



Starting off, let's check out their website...

Screenshot of Zgo home page

**ZGo**  
The Zcash Register

Home News Features Guides FAQ Info Contact Archive

## Electronic payment software for the 21st century.

Helping you receive payments in a safe, private, and independent way.

Latest News

- [Unified Addresses are here!](#) - October 30, 2023
- [ZGo's Security Hardening](#) - June 27, 2023

Launch App ▶



Like cash, but digital

An electronic payment that goes directly from your customer to you, no third-parties involved.





# Zgo



Starting off, let's check out their website...

Screenshot of Zgo home page



**\$1 / day**  
**\$6 / week**  
**\$22 / month**

...integration options. No APIs.

## Xero Accounting

- Designed for full accounting
- Manually create invoices
- View keys optional

## WooCommerce

- Requires Wordpress
- Requires view keys

## Zgo App

- Not a Point-of-Sale system
- Set-up and tracking



# Zgo



Starting off, let's check out their website...

Screenshot of Zgo home page



**ZGo**  
The Zcash Register

Home News Features Guides FAQ Info Contact Archive

Electronic payment software for the 21st century.

Helping you receive payments in a safe, private, and independent way.

Latest News

- [Unified Addresses are here!](#) - October 30, 2023
- [ZGo's Security Hardening](#) - June 27, 2023

[Launch App](#)



Like cash, but digital

An electronic payment that goes directly from your customer to you, no third-parties involved.

**\$1 / day**  
**\$6 / week**  
**\$22 / month**

...integration options. No APIs.

## Xero Accounting

- Designed for full accounting
- Manually create invoices
- View keys optional

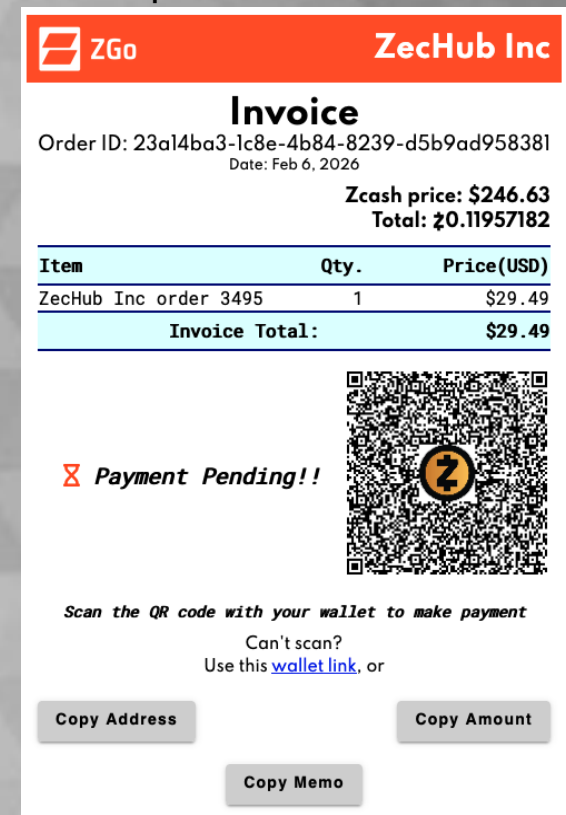
## WooCommerce

- Requires WordPress
- Requires view keys

## Zgo App

- Not a Point-of-Sale system
- Set-up and tracking

Example from Zechub.store



**ZGo** **ZecHub Inc**


**Invoice**

Order ID: 23a14ba3-1c8e-4b84-8239-d5b9ad958381  
Date: Feb 6, 2026

**Zcash price: \$246.63**  
**Total: ₳0.11957182**

Item	Qty.	Price(USD)
ZecHub Inc order 3495	1	\$29.49
<b>Invoice Total:</b>		<b>\$29.49</b>

**Payment Pending!!**



Scan the QR code with your wallet to make payment

Can't scan?  
Use this [wallet link](#), or

[Copy Address](#) [Copy Amount](#) [Copy Memo](#)

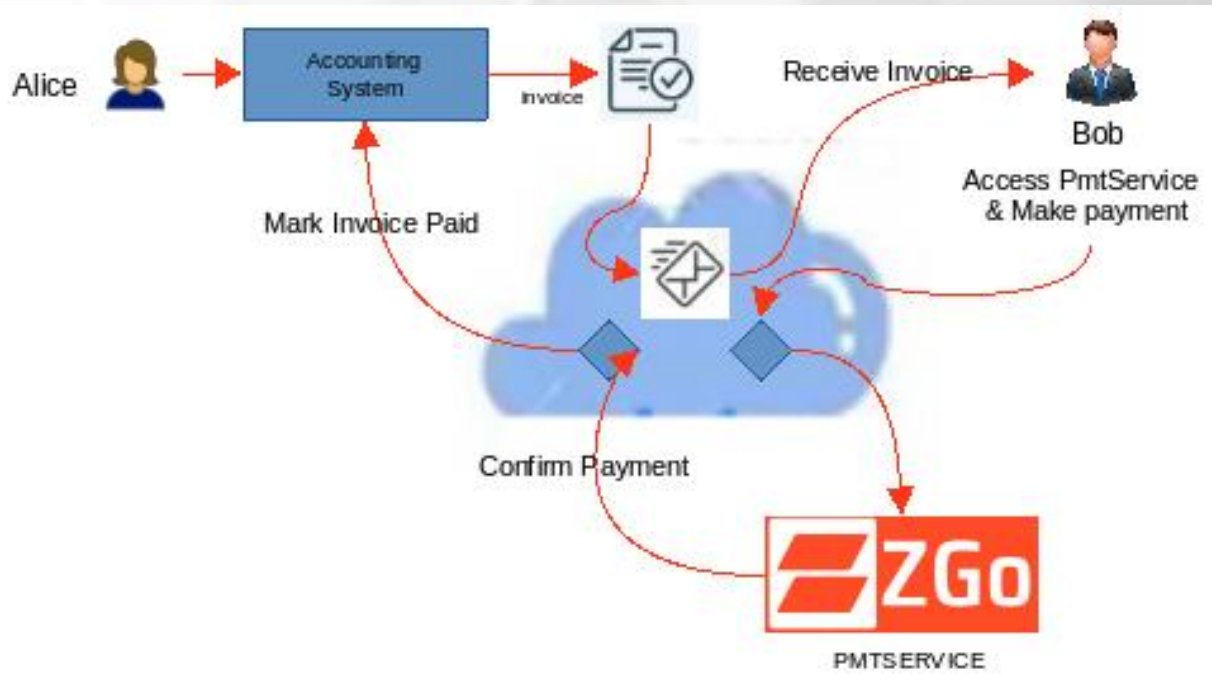
No API-token based access, requires multiple party integrations



# Zgo + Xero Accounting + Ywallet



Screenshot from Zgo.cash



## Xero Accounting

- Full accounting software
- Manually create invoices
- View keys optional
- Need a Ywallet for the business



## View Keys Optional

- Zgo would need READ ONLY access to your shielded transactions to confirm receipt and mark invoice as paid
- Zashi doesn't support view keys currently
- Need the older Ywallet to access view keys
- Alice could manually confirm invoice received and mark as paid, but then why bother with zgo?

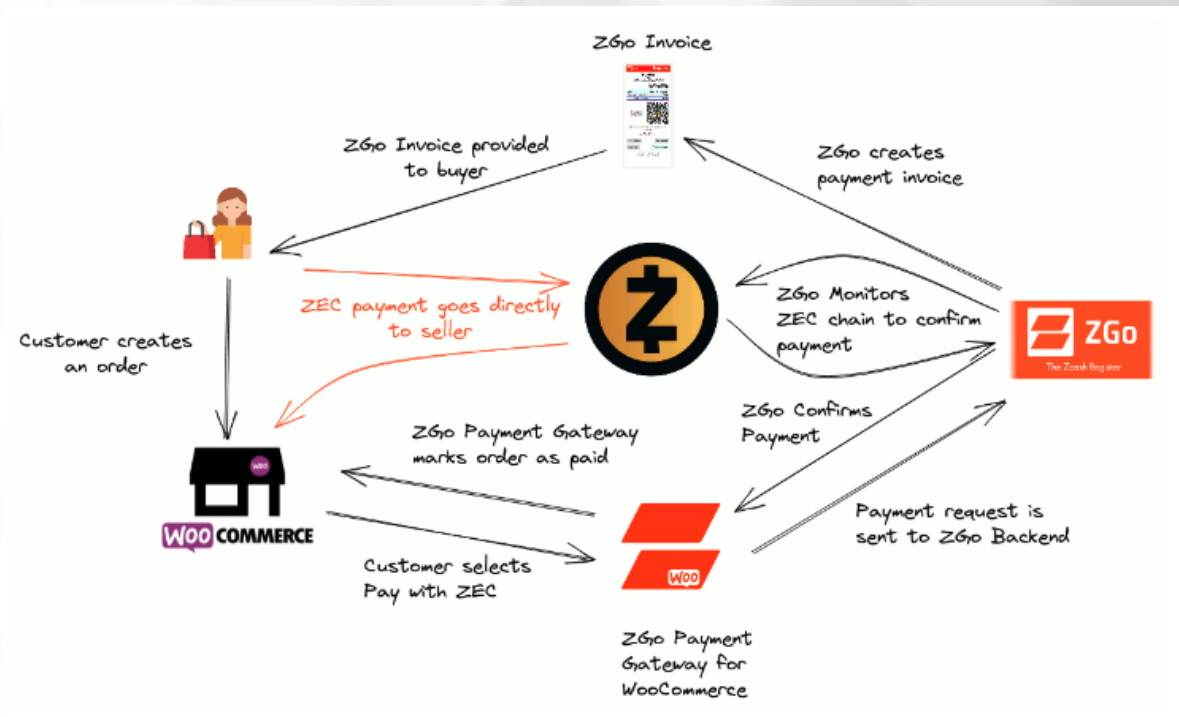
Accounting software designed for manual invoices but automatic confirmation



# Zgo + WooCommerce + WordPress + Ywallet



Screenshot from Zgo.cash



## Woo Commerce

- Requires WordPress
- Automatically create invoices
- Need a Ywallet for the business (desktop app available)

## Non-Rotating Receive Addresses

- On store set-up with Zgo, Alice sends a test message to Zgo with a code in the memo (no signing transactions like ETH)
- This sets ONE receive address
- View keys for this receive address
- Relies on the memo field to match invoice + payment
- **!! Bookkeeping nightmare if user erases memo with unique invoice note and sends mismatched amount**

Simple option for WordPress-based sites looking for basic services





		Square	Zcash SDK	Zgo	BTCPay	NOWPayments
Privacy	Shielded TX	No	Yes	Yes	Yes	No
	Native ZEC	No	Yes	Yes	Yes	Yes
Control	Custodial	Yes	No	No	No	Optional
	Run Own Node	No	Yes	No	Optional	No
	Fees	1%	-	\$22/mo	\$10-30/mo	0.5%
Complexity	Set Up	Low	High	Moderate	Moderate	Low
	Readiness	Mature	Mature	Early	Mature	Mature
Multi-coin		BTC only	No	No	XMR, DASH,...	XMR, DASH,...

Simple storefront for Wordpress sites or low volume businesses using Xero accounting



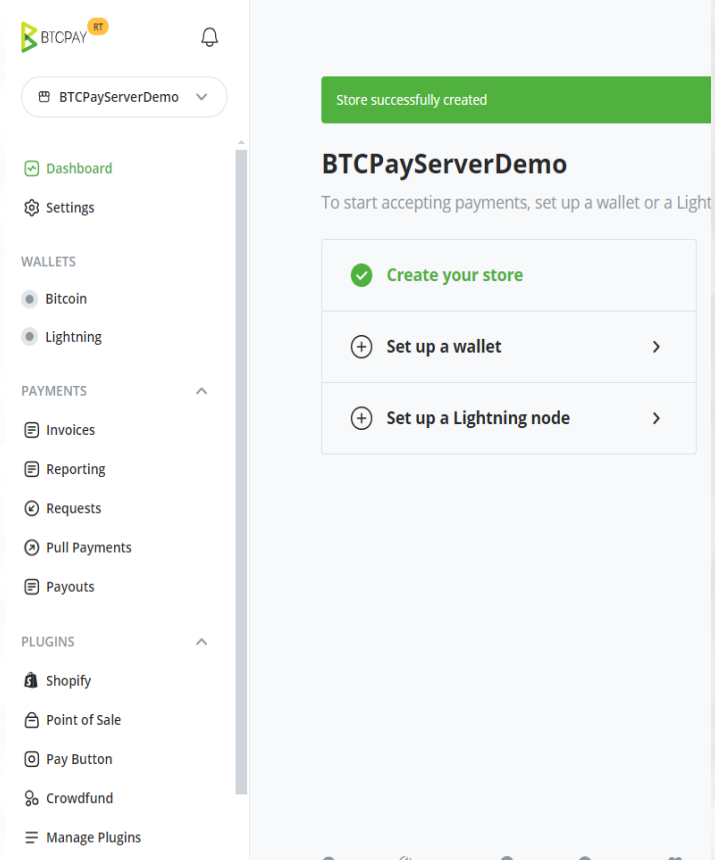
		Square	Zcash SDK	Zgo	BTCPay	NOWPayments
Privacy	Shielded TX	No	Yes	Yes	Yes	No
	Native ZEC	No	Yes	Yes	Yes	Yes
Control	Custodial	Yes	No	No	No	Optional
	Run Own Node	No	Yes	No	Yes	No
	Fees	1%	-	\$22/mo	\$10-30/mo	0.5%
Complexity	Set Up	Low	High	Moderate	Moderate	Low
	Readiness	Mature	Mature	Early	Mature	Mature
Multi-coin		BTC only	No	No	XMR, DASH,...	XMR, DASH,...



# Major System Components

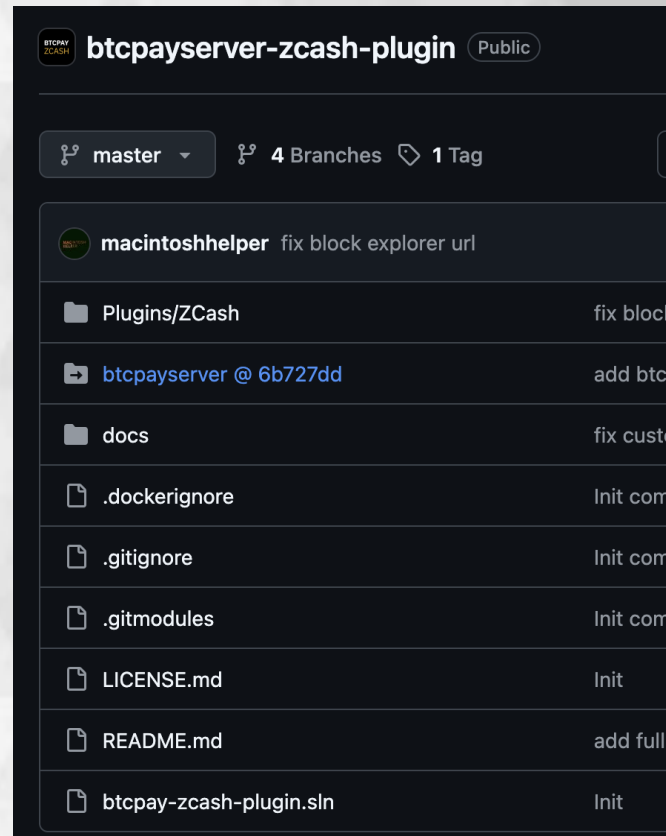


## BTCPay Server



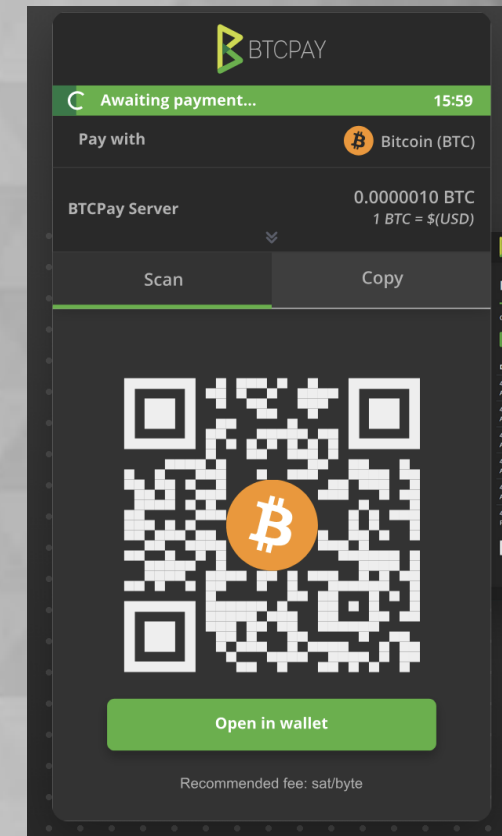
+

## Zcash Plugin



=>

## Operational Front

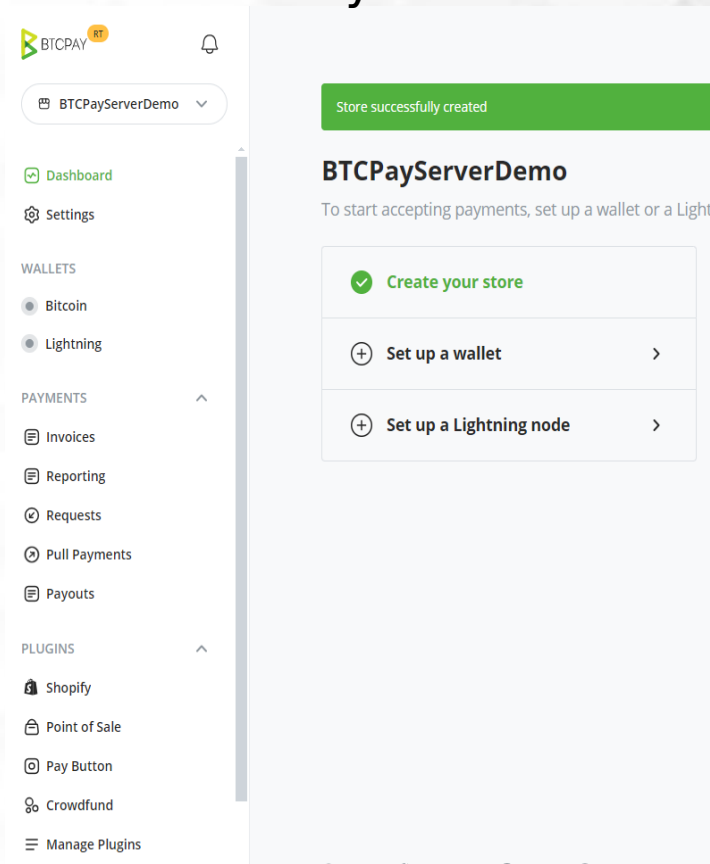




# Major System Components

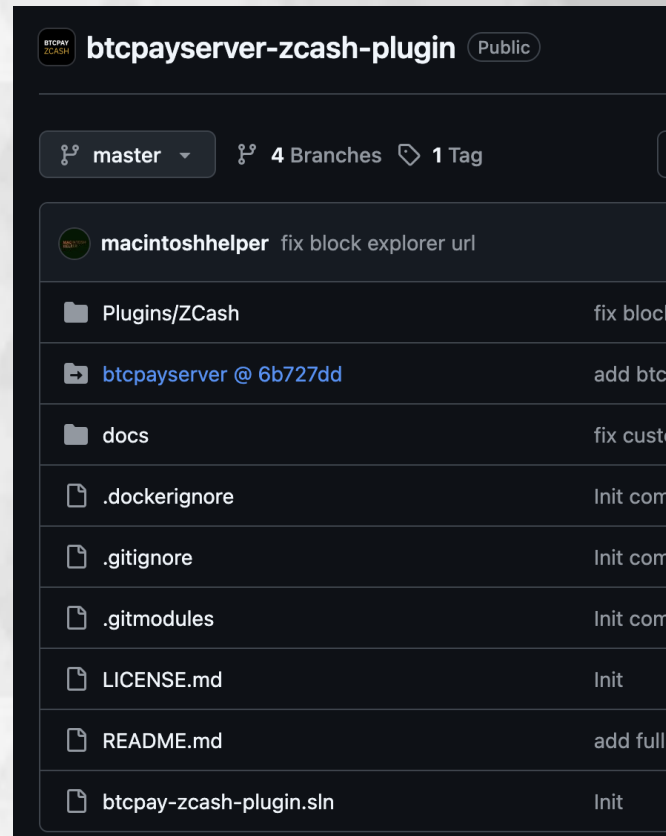


## BTCPay Server



+

## Zcash Plugin



=>

## Operational Elements

### Hosted BTCPay Server

- 1-Click VPS/Cloud options
- Self-Hosted, Self-Custody
- Enables multi-coin support

### Full ZCash Node

- Enables own custodial zcash wallet
- Self-Hosted
- + Zingo/Ywallet view keys

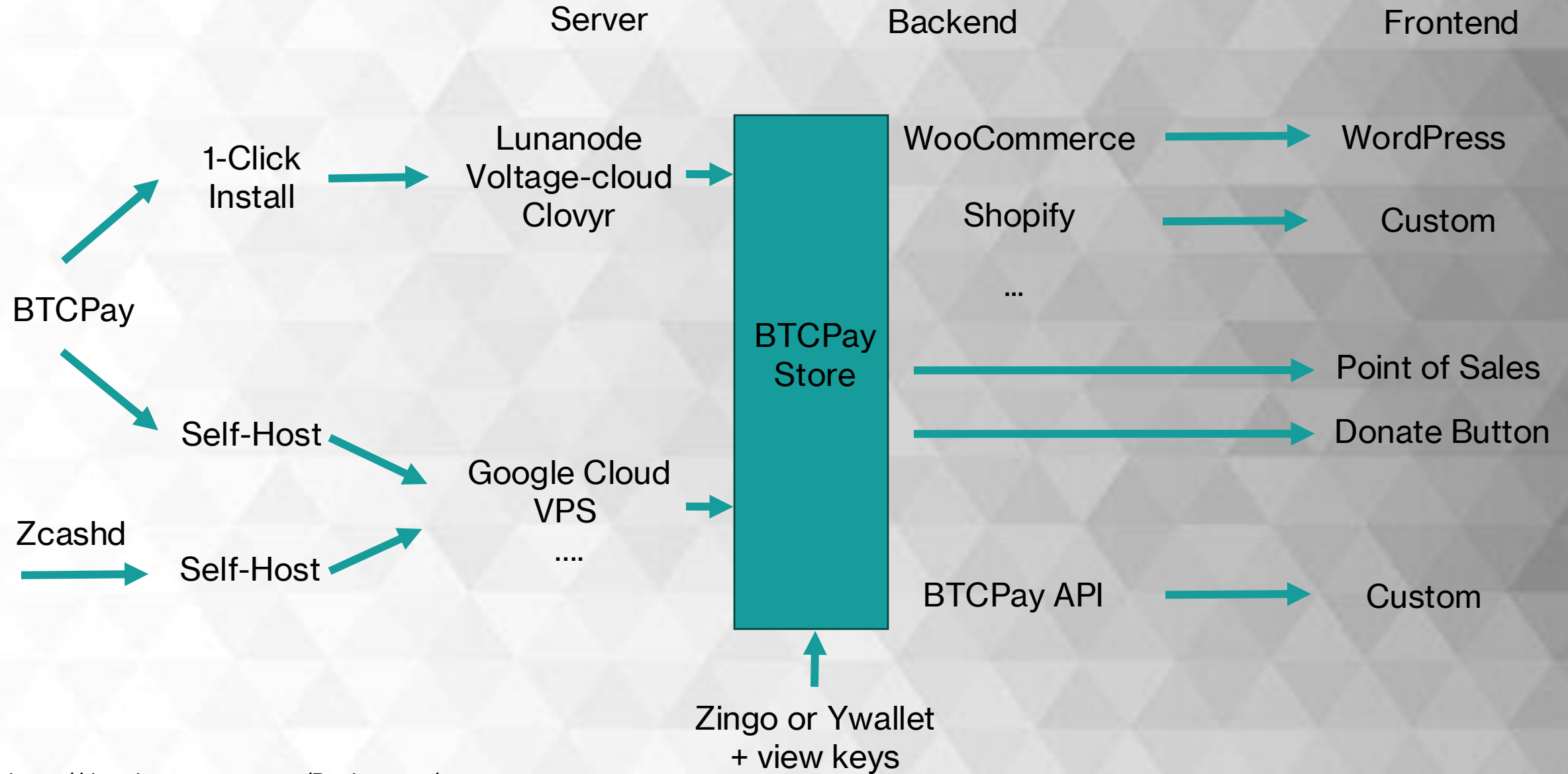
### Frontend:

- Custom Site w. API
- WooCommerce, Zapier, Shopify, etc
- Plus 24+ other integration partners
- Point of Sales App
- Embeddable HTML pay button

Zcash Community Grant funded Zcash Plugin, enables ZEC payments via BTCPay



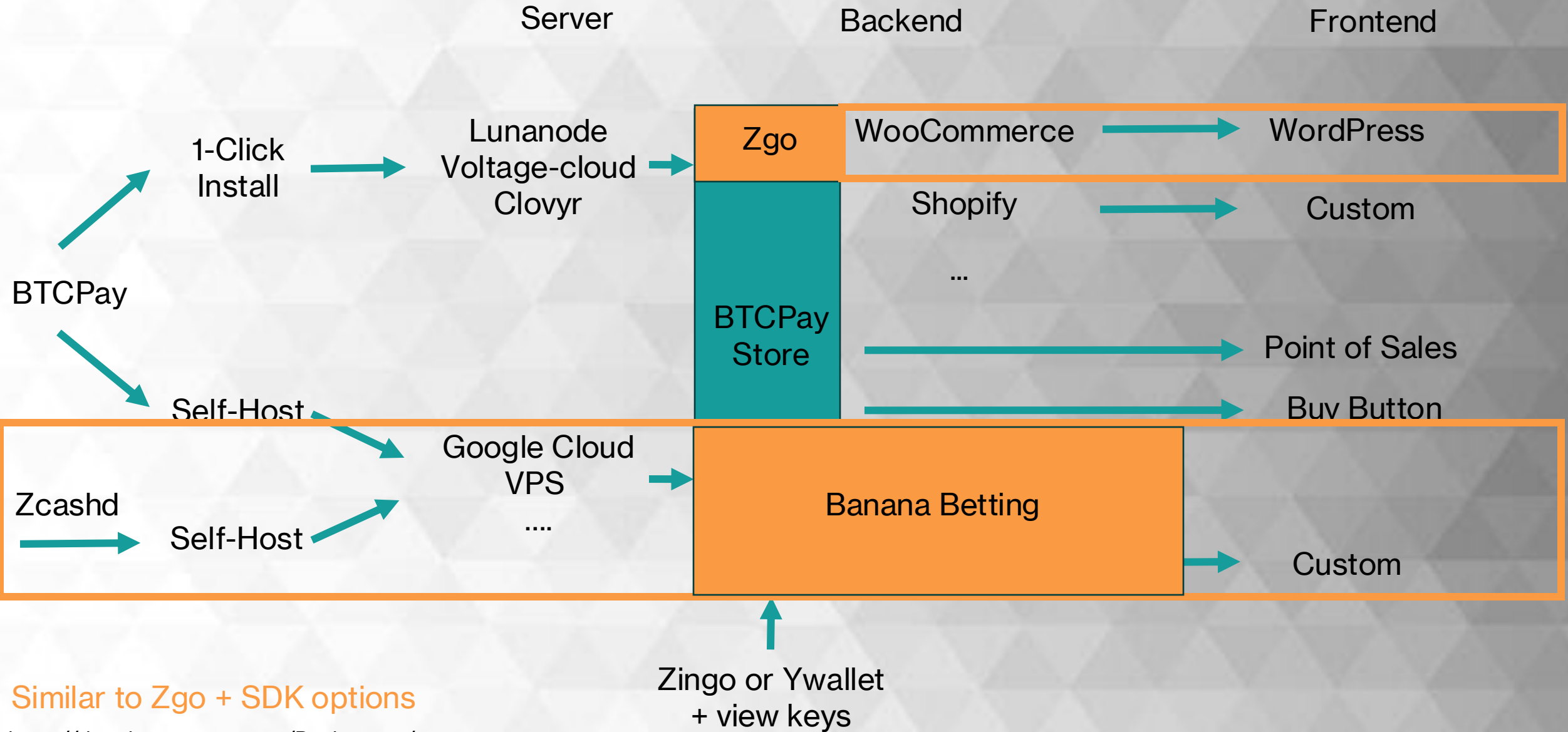
# Deployment Options





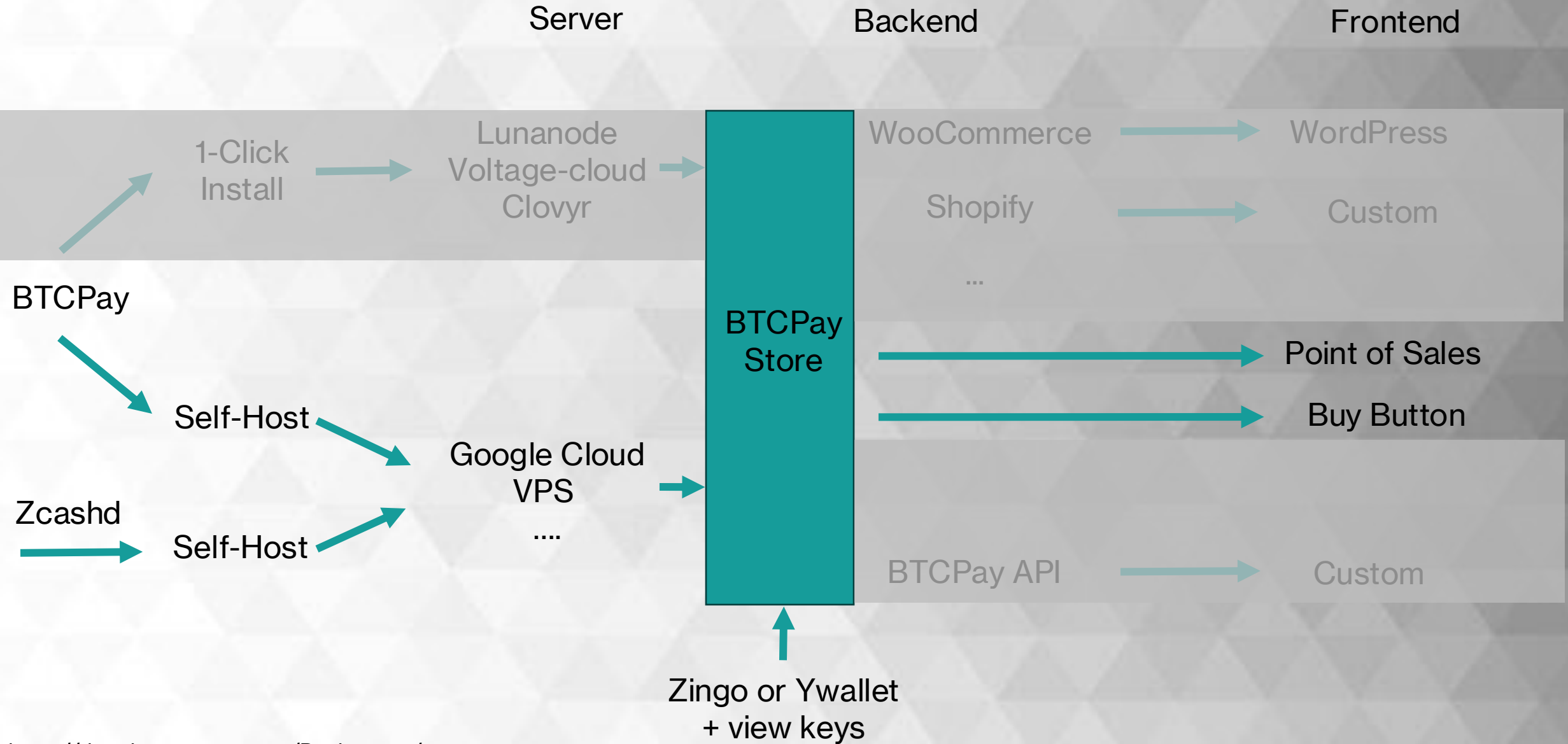


# Deployment Options





# Deployment Options





# From-Scratch Backend Implementation

# Clone the docker fragment repository

```
git clone https://github.com/btcpay-zcash/btcpayserver-docker
```

**Copy the code!**

# Add ZEC plugin

```
git checkout feat/zec
```

**Add ZEC plugin**

# Run btcpay-setup.sh with the right parameters

```
export BTCPAY_HOST="btcpay.example.com"
```

```
export NBITCOIN_NETWORK="mainnet"
```

```
export BTCPAYGEN_CRYPT01="zec"
```

```
export BTCPAYGEN_CRYPT02="btc"
```

```
export BTCPAYGEN_CRYPT02="..."
```

```
export BTCPAYGEN_ADDITIONAL_FRAGMENTS=""
```

```
export BTCPAYGEN_REVERSEPROXY="nginx" . ./btcpay-setup.sh -l
```

**Multi-coin Support**

# Full Node

-> \*\*The first part of the talk on running Zcash nodes\*\*

Requires CLI familiarity and networking skills



# BTCPay GUI Setup 1 of 2



BTCPAY RT

BTCPayServerDemo

Dashboard

Settings

WALLETS

- Bitcoin
- Lightning

PAYMENTS

- Invoices
- Reporting
- Requests
- Pull Payments
- Payouts

PLUGINS

- Shopify
- Point of Sale
- Pay Button
- Crowdfund
- Manage Plugins

Store successfully created

### BTCPayServerDemo

To start accepting payments, set up a wallet or a Lightning node.

- ✓ Create your store
- + Set up a wallet >
- + Set up a Lightning node >

Create your store!

#### Installed Plugins

**BTCPay Server: ZCash support plugin**  
by btc-pay-zcash  
1.0.2.0

This plugin extends BTCPay Server to enable users to receive payments via ZCash.

Dependencies

BTCPayServer: >=2.1.0

Resources

- Sources
- Documentation

Uninstall

Zcash

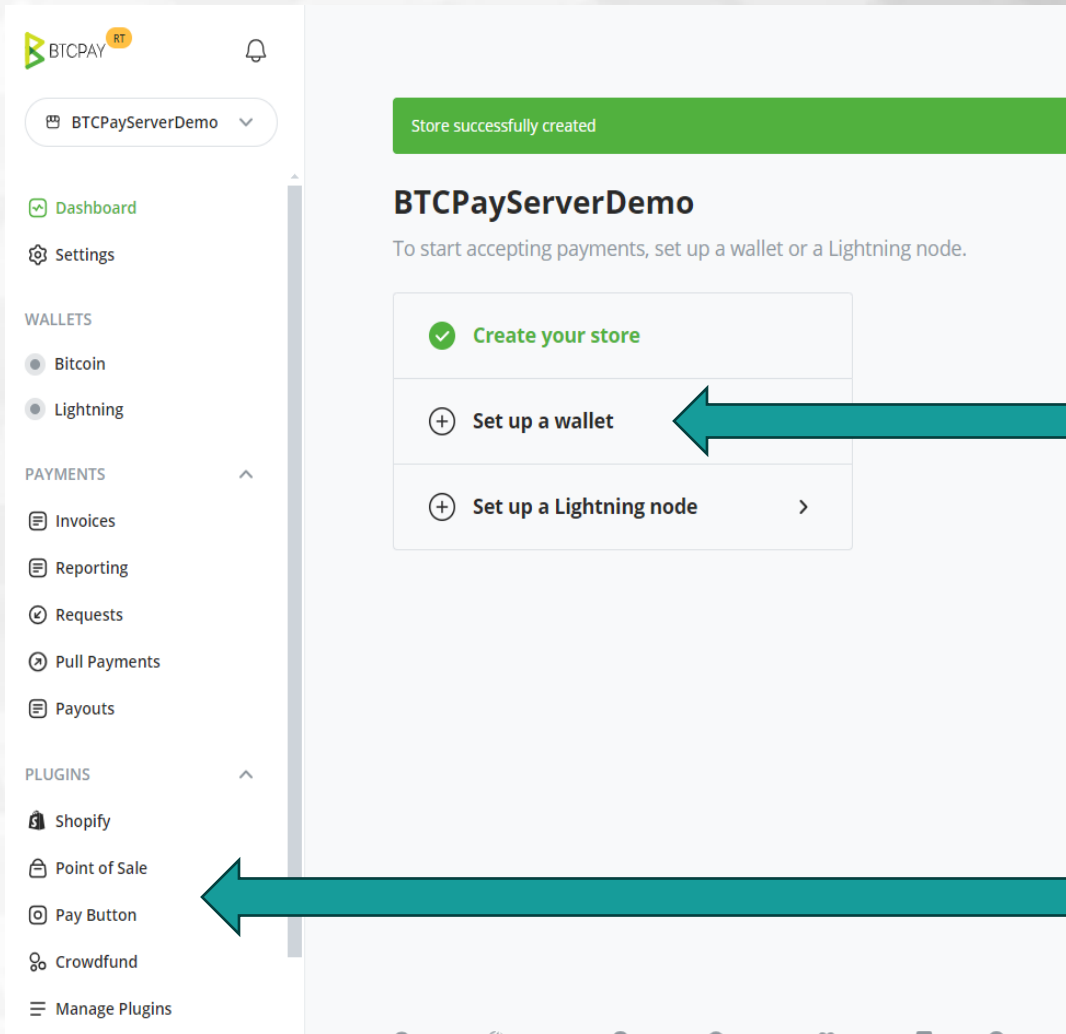
Available Plugins

Graphic interface for store creation and ZEC plugin





# BTC Pay GUI Setup 2 of 2



**Set up your wallets**

**Point of Sales  
Pay Button integrations**

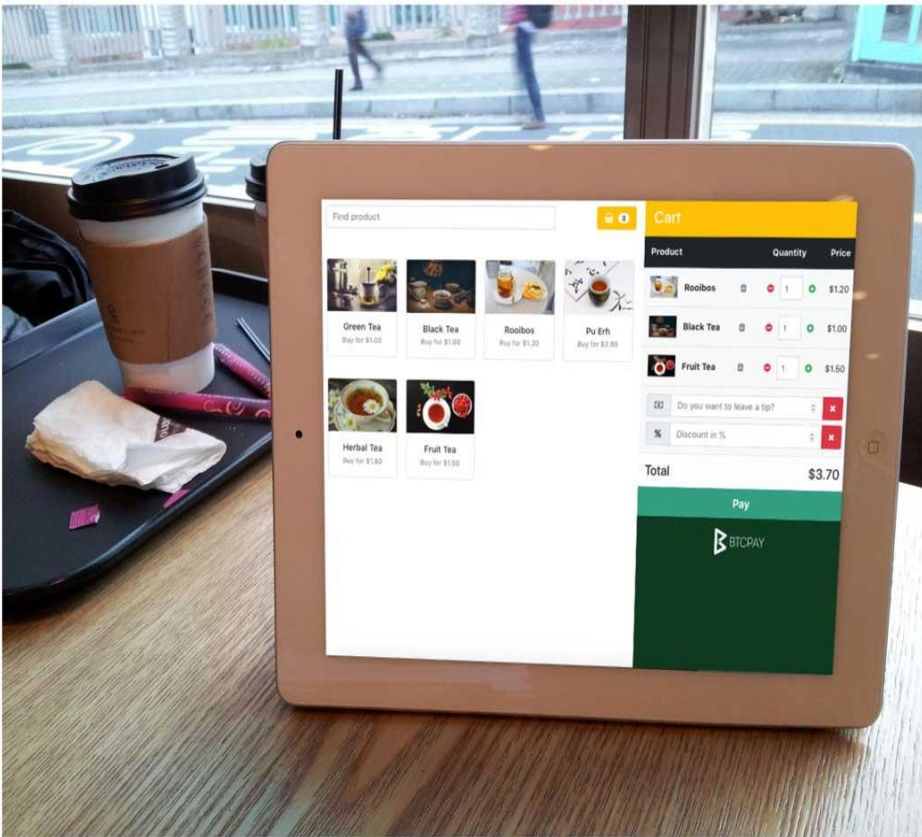
**BTC Pay includes plugins for Point of Sales App and HTML site integrations**



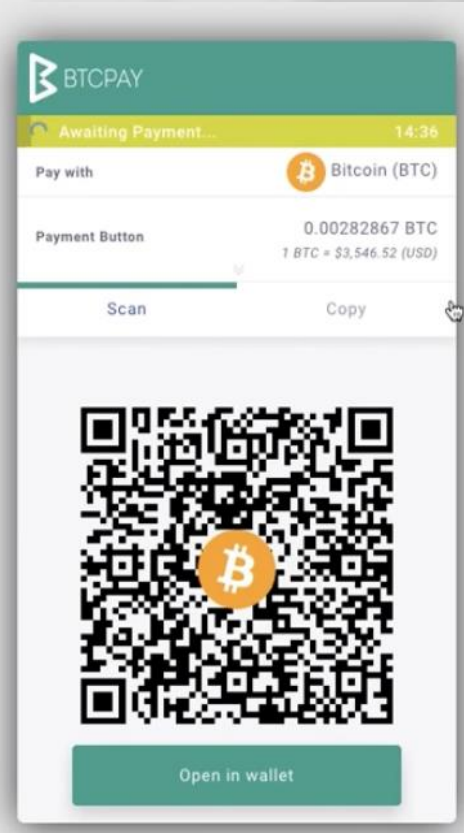
# Frontend Implementation



## BTCPay Point of Sales



## Embeddable HTML



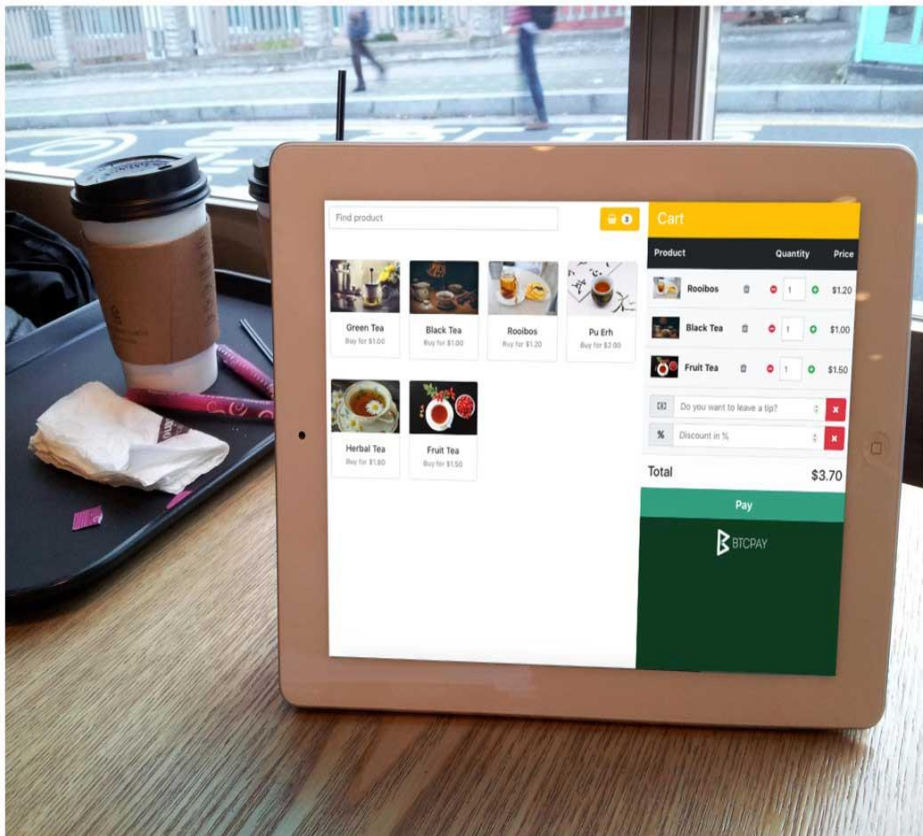
Now accepting shielded ZEC



# Frontend Implementation



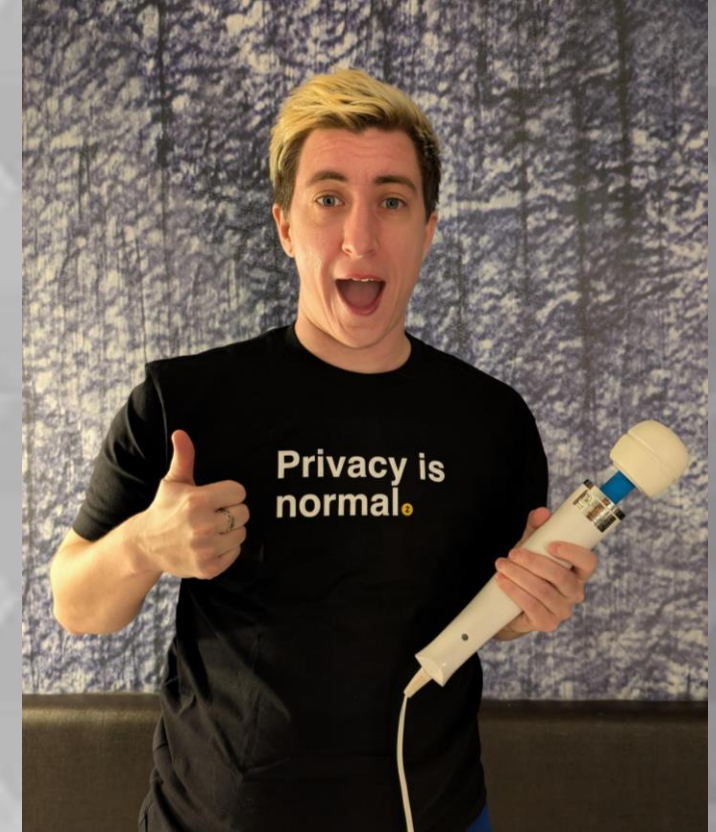
## BTCPay Point of Sales



## Embeddable HTML



## Ready for Business



Now accepting shielded ZEC



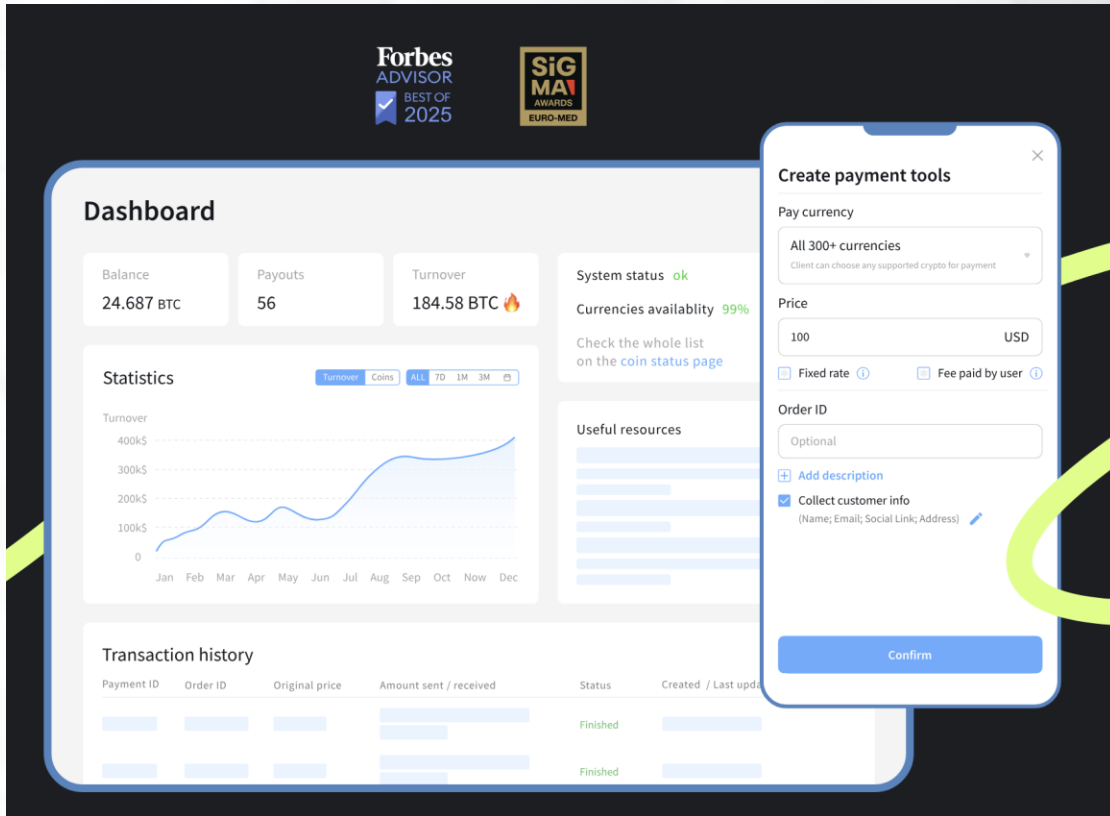


		Square	Zcash SDK	Zgo	BTCPay	NOWPayments
Privacy	Shielded TX	No	Yes	Yes	Yes	No
	Native ZEC	No	Yes	Yes	Yes	Yes
Control	Custodial	Yes	No	No	No	Optional
	Run Own Node	No	Yes	No	Yes	No
	Fees	1%	-	\$22/mo	\$10-30/mo	0.5%
Complexity	Set Up	Low	High	Moderate	Moderate	Low
	Readiness	Mature	Mature	Early	Mature	Mature
Multi-coin		BTC only	No	No	XMR, DASH,...	XMR, DASH,...





# NOWpayments



## Accept Zcash Payments

NOWPayments is a crypto payment gateway that lets you accept ZCash payments and donations in an easy and convenient way. Try this ZCash payment gateway NOW!

- ✓ Accept anything: 300+ cryptocurrencies supported
- ✓ Let your customers pay with anything: Auto coin conversion
- ✓ Explore many tools: API, plugins, invoices, donation tools and customized solutions
- ✓ Non-custodial: you are in control
- ✓ Save on fees: cheapest on the crypto market

**+ Fiat off-ramping**  
**+ Currency Exchange**

**- Only transparent ZEC**

Fully featured PoS payment processor, including fiat off-ramping, but only transparent ZEC



# Summary



		Square	Zcash SDK	Zgo	BTCPay	NOWPayments
Privacy	Shielded TX	No	Yes	Yes	Yes	No
	Native ZEC	No	Yes	Yes	Yes	Yes
Control	Custodial	Yes	No	No	No	Optional
	Run Own Node	No	Yes	No	Optional	No
	Fees	1%	-	\$22/mo	\$10-30/mo	0.5%
Complexity	Set Up	Low	High	Moderate	Moderate	Low
	Readiness	Mature	Mature	Early	Mature	Mature
Multi-coin		BTC only	No	No	XMR, DASH,...	XMR, DASH,...

No clear winner. It's all about trade-offs.



# The Future



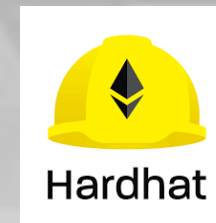
## What's Missing...

- Square + NOWPayments: Convince them to support shielded ZEC
- SDK:
  - Terraform modules for zcash full nodes (mainnet + testnet)
  - Maintained Python packages, JavaScript library, etc for major function calls
    - Some partial projects exist
- Zgo:
  - Upgrade the UX
  - Add other integration partners
- BTCPay:
  - Rotating shielded address for Zcash plugin
  - Terraform module for BTCPay + Zcash node combo
- Development:
  - Hardhat-like blockchain simulator for local development



HashiCorp

**Terraform**



Hardhat

Wanna build one of these? Apply for a grant: <https://zcashcommunitygrants.org>



# Resources



Code Examples + This Slide Deck	<a href="https://github.com/ReadyMouse/BananaBetting">https://github.com/ReadyMouse/BananaBetting</a>
ZEC RPC Nodes	<a href="https://docs.tatum.io/reference/rpc-zcash">https://docs.tatum.io/reference/rpc-zcash</a>
Pay Processor for Non-profits	<a href="https://www.every.org/crypto">https://www.every.org/crypto</a>
Zcashd Node Docker Files	<a href="https://github.com/zecrocks/zcash-stack/blob/main/docker/compose.zcashd.yaml">https://github.com/zecrocks/zcash-stack/blob/main/docker/compose.zcashd.yaml</a>
Zcash Community Grants	<a href="https://zcashcommunitygrants.org/">https://zcashcommunitygrants.org/</a>
BTCPay Zcash Plugin	<a href="https://github.com/btcpay-zcash/btcpayserver-zcash-plugin/tree/master">https://github.com/btcpay-zcash/btcpayserver-zcash-plugin/tree/master</a>