Private Payments in Practice

Integrating Zcash from Scratch to Production

ETHBoulder Feb 2026

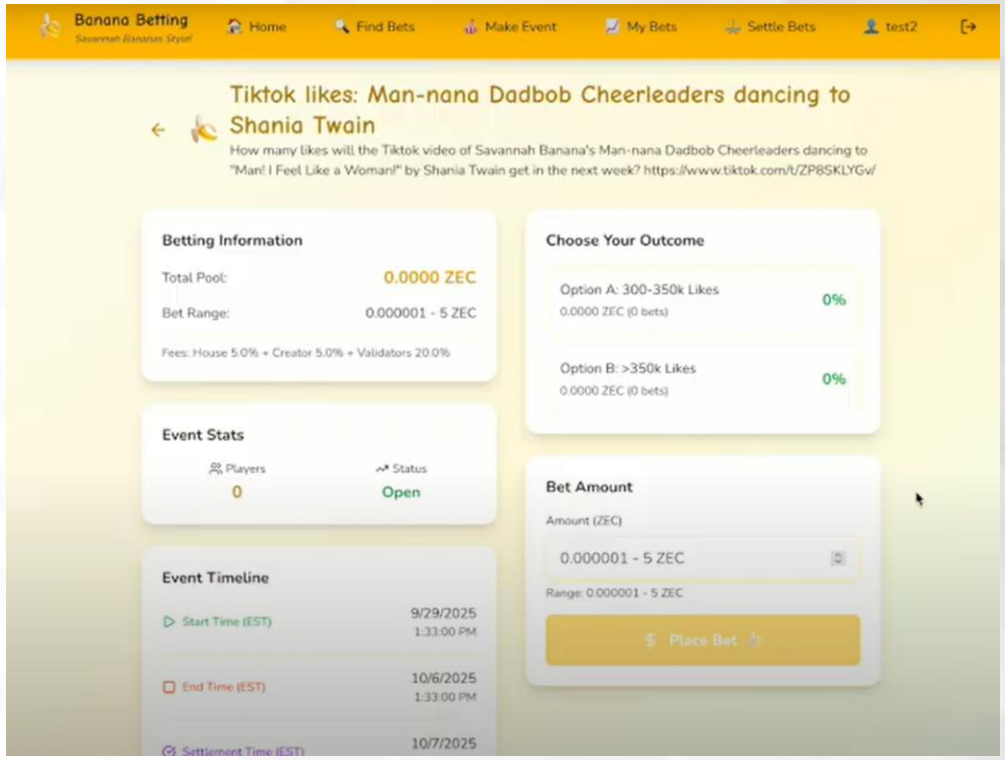
Mylo Bennett aka ReadyMouse

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



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

*RPC: Remote Procedure Call

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

ZEC-Based Development

I can't find a ZEC blockchain simulator

Testnet.zecfaucet → Wait, what wallet supports TAZ? Zingo

Tatum.io RPC → READ ONLY, not helpful for a private blockchain

No Smart Contracts, no programable money

Own Code + Own Node → Create wallets + receive addresses

Lack of smart contracts and mature development infrastructure for zcash

*RPC: Remote Procedure Call

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

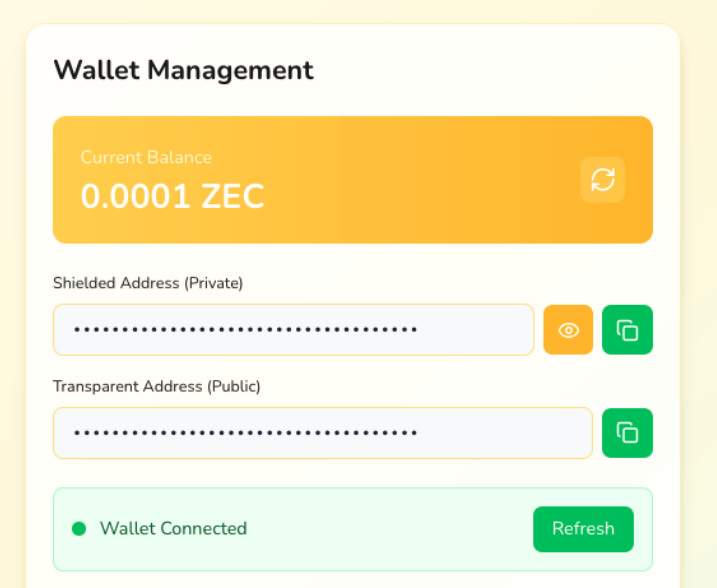
Lack of smart contracts and mature development infrastructure for zcash

*RPC: Remote Procedure Call

Own Code + Own Node

ZECHUB 2025 Hackathon 3rd Place Winner: Banana Betting

Frontend



Not covered in this talk

Backend

```python
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

```yaml
# Simple `zcashd` Docker Compose configuration.
#
# THIS DAEMON IS BEING DEPRECATED AND SHOULD NOT BE USED FOR NEW
# SERVERS. ONLY USE THIS IF YOU KNOW YOU HAVE A SPECIFIC NEED FOR
# THE OLDER IMPLEMENTATION OF A ZCASH FULL NODE.
#
# Instead of this configuration, you should use `compose.yaml`, the
# new default with the `zebrad` service configuration.
---
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
```

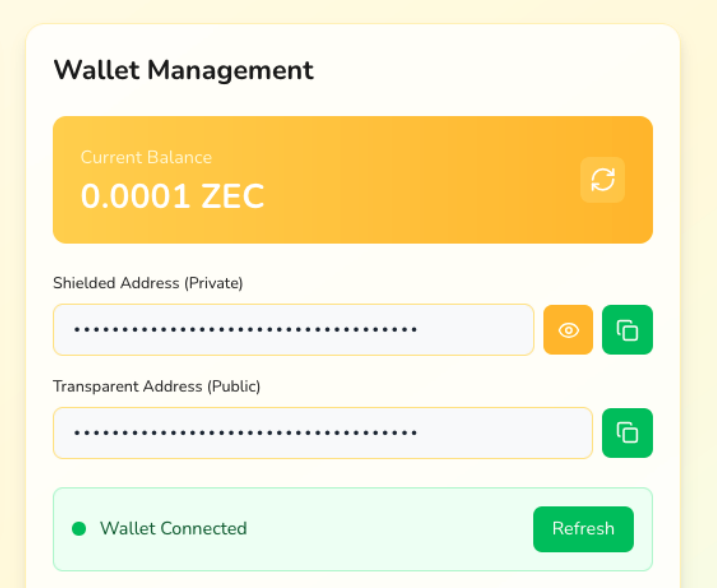
- Docker + server
- Interacts with blockchain
- Read/write capabilities

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

Own Code + Own Node

ZECHUB 2025 Hackathon 3rd Place Winner: Banana Betting

Frontend



Not covered in this talk

Backend

```python
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

```yaml
# Simple `zcashd` Docker Compose configuration.
#
# THIS DAEMON IS BEING DEPRECATED AND SHOULD NOT BE USED FOR NEW
# SERVERS. ONLY USE THIS IF YOU KNOW YOU HAVE A SPECIFIC NEED FOR
# THE OLDER IMPLEMENTATION OF A ZCASH FULL NODE.
#
# Instead of this configuration, you should use `compose.yaml`, the
# new default with the `zebrad` service configuration.
---
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
```

- 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 th
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
```

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)
)
```

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_listRecievedByAddress: get all transactions for an address

*RPC: Remote Procedure Call https://zcash.github.io/rpc/

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!!

zcash_sdk > zcash_utils.py

```python
import requests
from fastapi import Depends, FastAPI, HTTPException, status, Query
fro
def
fr
fr
fr
fr
fr
fr
mo
ap
or
]
ap
)
de
```

zcash_sdk > main_transactions.py

```python
from fastapi import Depends, FastAPI, HTTPException, status
from fastapi.middleware.cors import CORSMiddleware
from sqlalchemy.orm import Session
```

zcash_sdk > zcash_wallet.py

```python
import requests
from fastapi import Depends, FastAPI, HTTPException, status, Query
from ..zcash_mod import ZCASH_RPC_URL, ZCASH_RPC_USER, ZCASH_RPC_PASSWORD, DISABLE_ZCASH_NODE
from decimal import Decimal
import simplejson

# Mock balances for development (user_id -> balance)
_mock_user_balances = {}
_mock_pool_balance = 1000.0

def backupwallet(destination: str):
    try:
        # RPC request payload
        payload = {
            "jsonrpc": "1.0",
            "id": "backupwallet",
            "method": "backupwallet",
            "params": [destination]
        }

        # 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()
        print(validation_data)
        return validation_data['result']

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


def z_get_new_account():
    try:
```

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

RPC Function Call definitions:

- Zcash_wallet.py
- Zcash_utils.py

Example: main_transactions.py

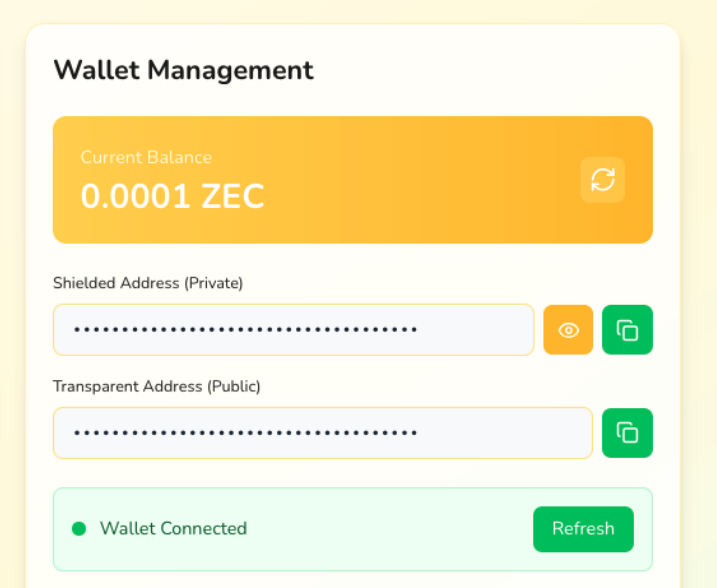
Now we have code to...

- 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

Own Code + Own Node

Frontend



Not covered in this talk

Backend

```python
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

```yaml
# Simple `zcashd` Docker Compose configuration.
#
# THIS DAEMON IS BEING DEPRECATED AND SHOULD NOT BE USED FOR NEW
# SERVERS. ONLY USE THIS IF YOU KNOW YOU HAVE A SPECIFIC NEED FOR
# THE OLDER IMPLEMENTATION OF A ZCASH FULL NODE.
#
# Instead of this configuration, you should use `compose.yaml`, the
# new default with the `zebrad` service configuration.
---
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
```

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

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

Zcash Node – To the Docs!

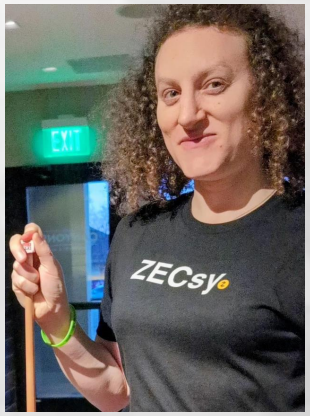
Node Requirements:

- 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

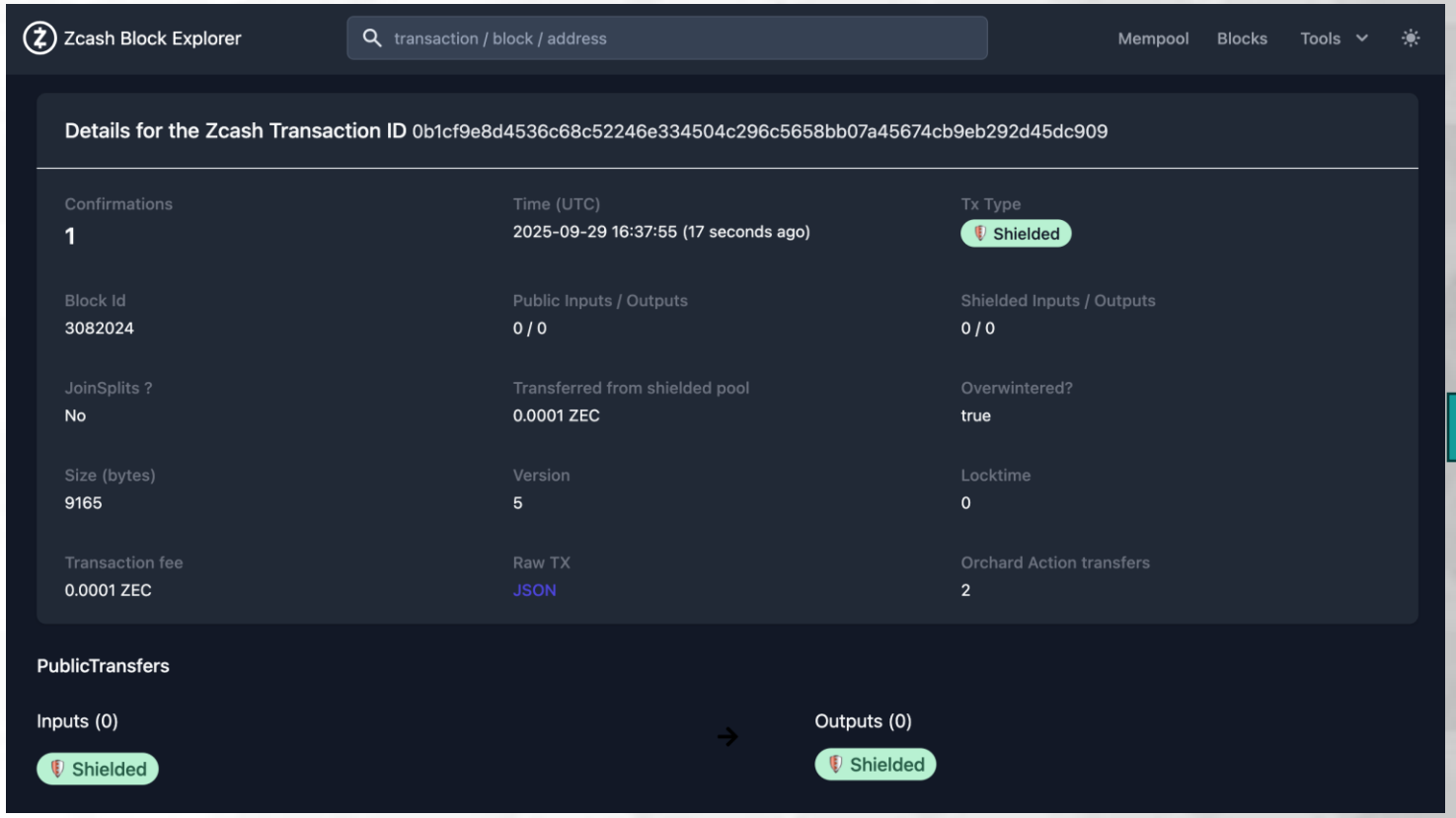
```
# Zcashd configuration file.
i-am-aware-zcashd-will-be-replaced-by-zebrad-and-zallet-in-2025=1
txindex=1
experimentalfeatures=1
lightwalletd=1
disablewallet=0
rpcbind=0.0.0.0
rpcallowip=0.0.0.0/0

rpcuser=rpcuser
rpcpassword=PASSWORD
```

Example Docker File for Node Instance

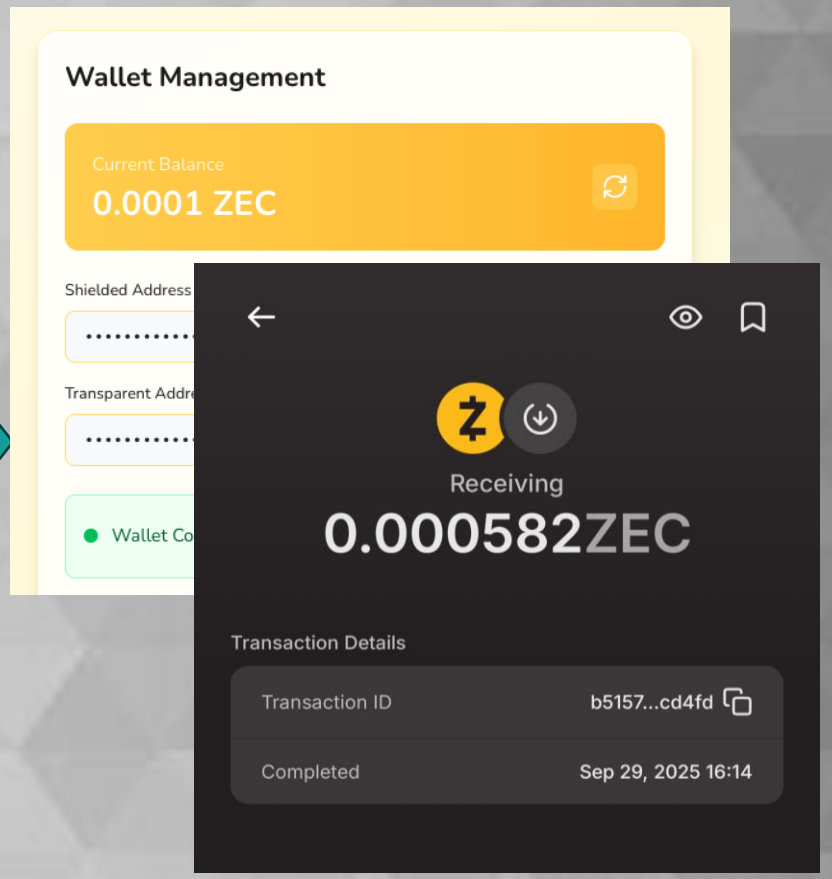
```
services:
  zcashd:
    image: electriccoinco/zcashd:v6.3.0
    restart: unless-stopped
    configs:
      - source: zcashd_conf
        target: /etc/zcashd/zcashd.conf
    volumes:
      - ./data/zcashd-cache:/srv/zcashd/.zcash
      - ./data/zcashd-params:/srv/zcashd/.zcash-params
    ports:
 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



Zashi Deposit



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

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

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

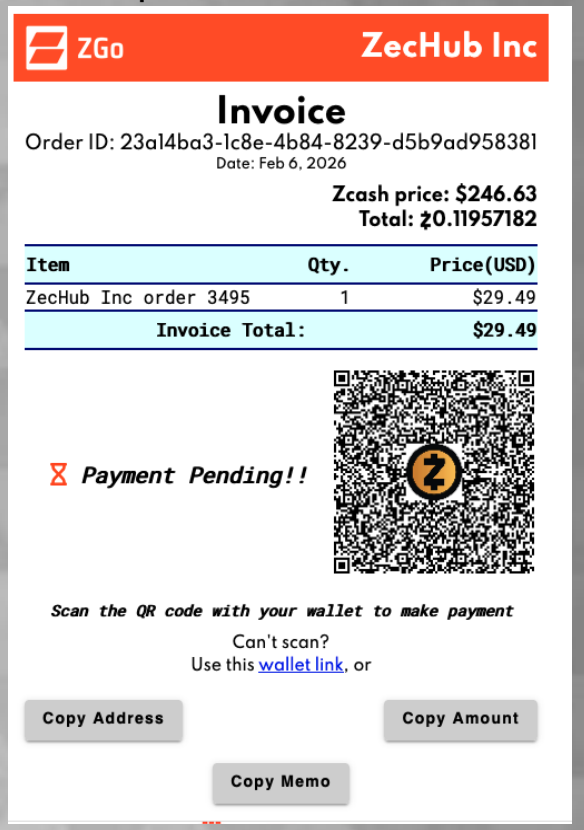
...integration options. No APIs.

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

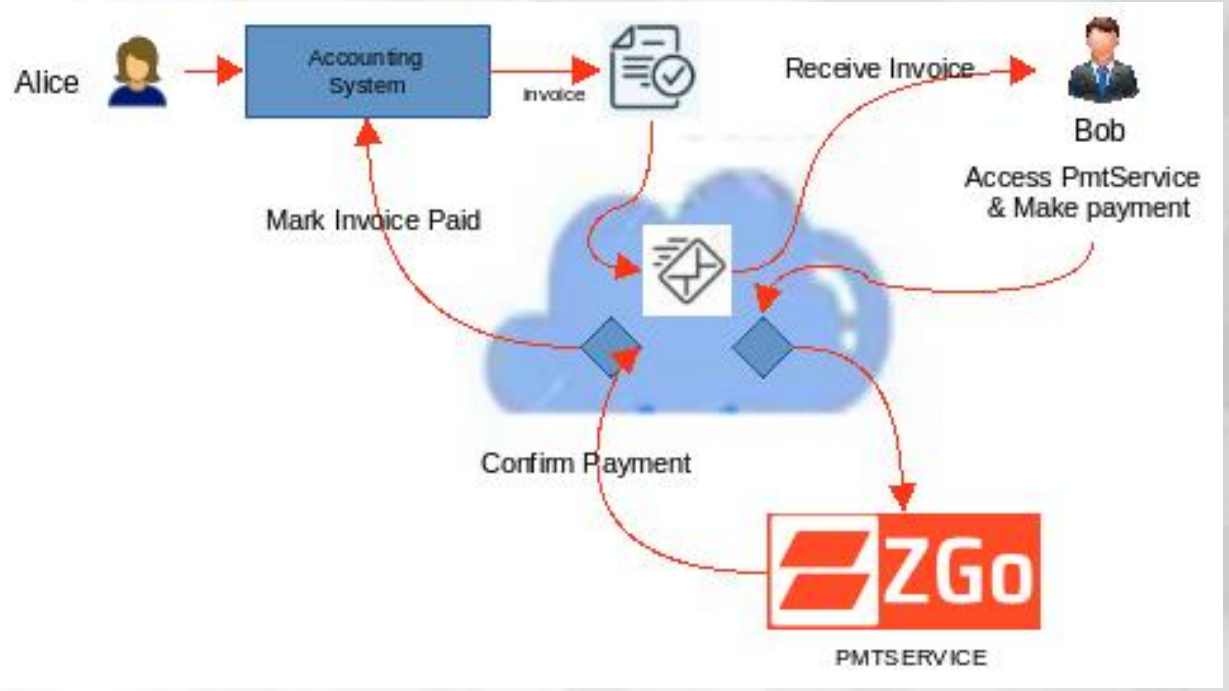
Example from Zechub.store



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

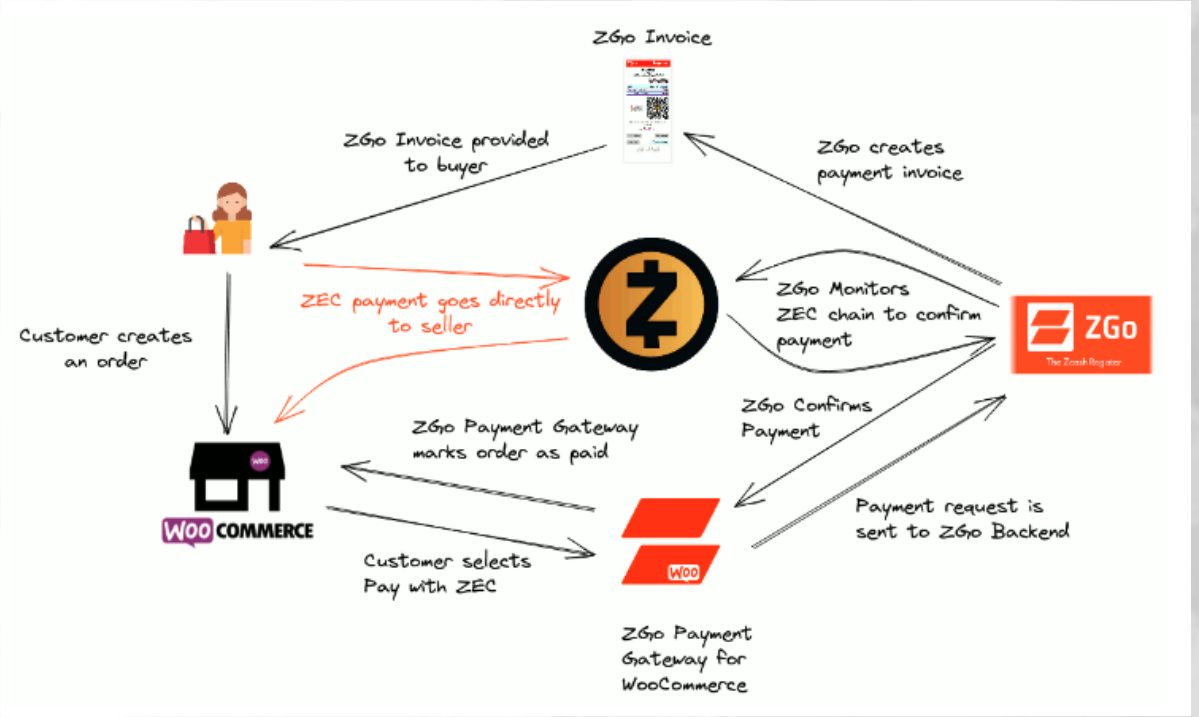
View Keys Optional

- 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

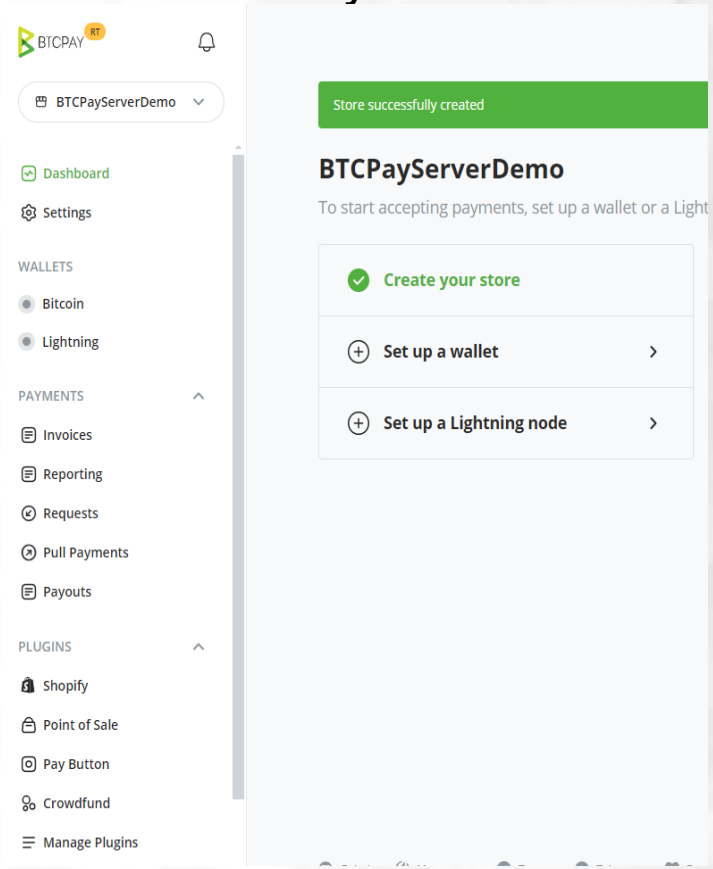
		Square	Zcash SDK	Zgo	BTCPay	NOWPayments
Privacy	Shielded TX	No	Yes	Yes	Yes	No
	Native ZEC	No	Yes	Yes	Yes	Yes
Control	Custodial	Yes	No	No	No	Optional
	Run Own Node	No	Yes	No	Optional	No
	Fees	1%	-	$22/mo	$10-30/mo	0.5%
Complexity	Set Up	Low	High	Moderate	Moderate	Low
	Readiness	Mature	Mature	Early	Mature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

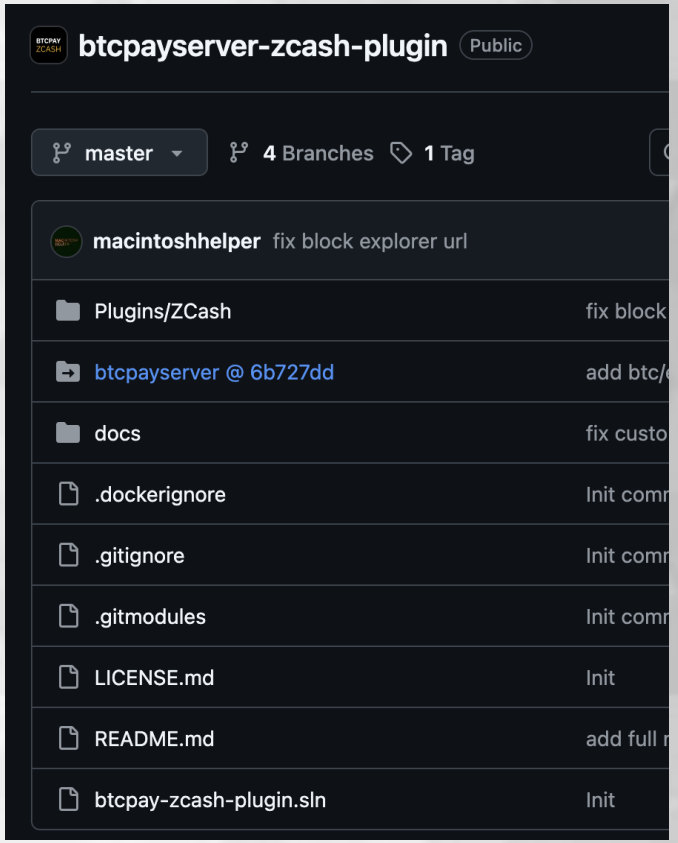
Major System Components

BTCPay Server



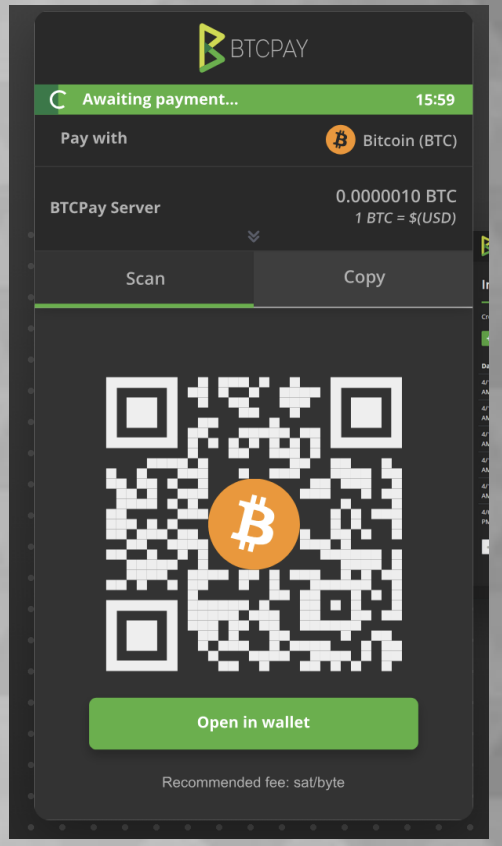
+

Zcash Plugin



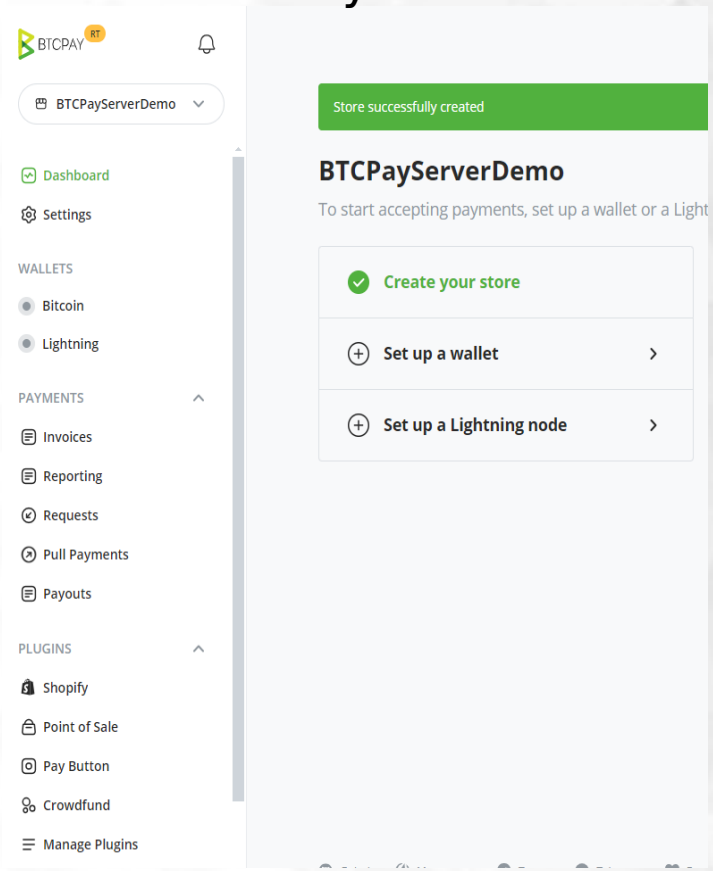
=>

Operational Front



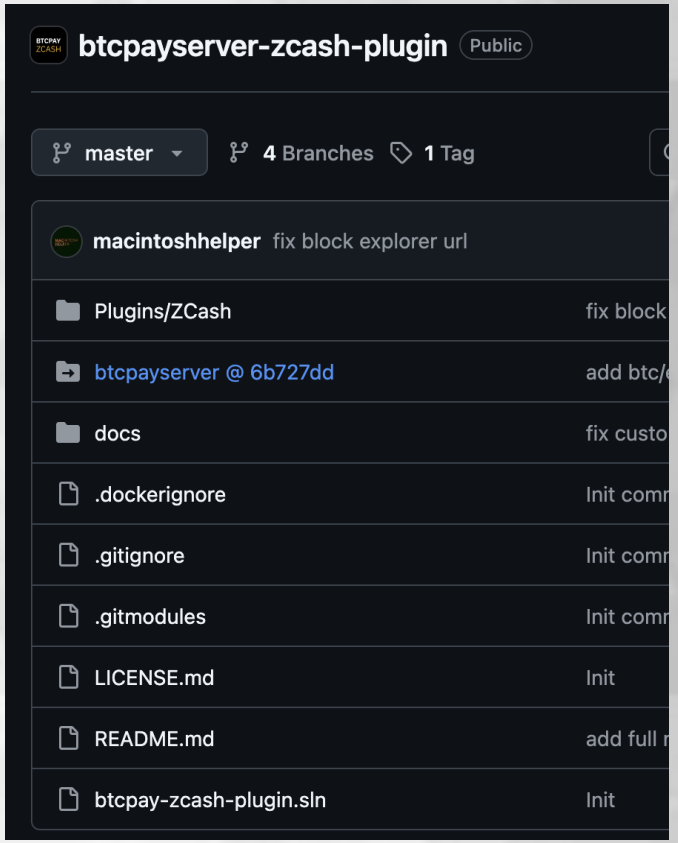
Major System Components

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

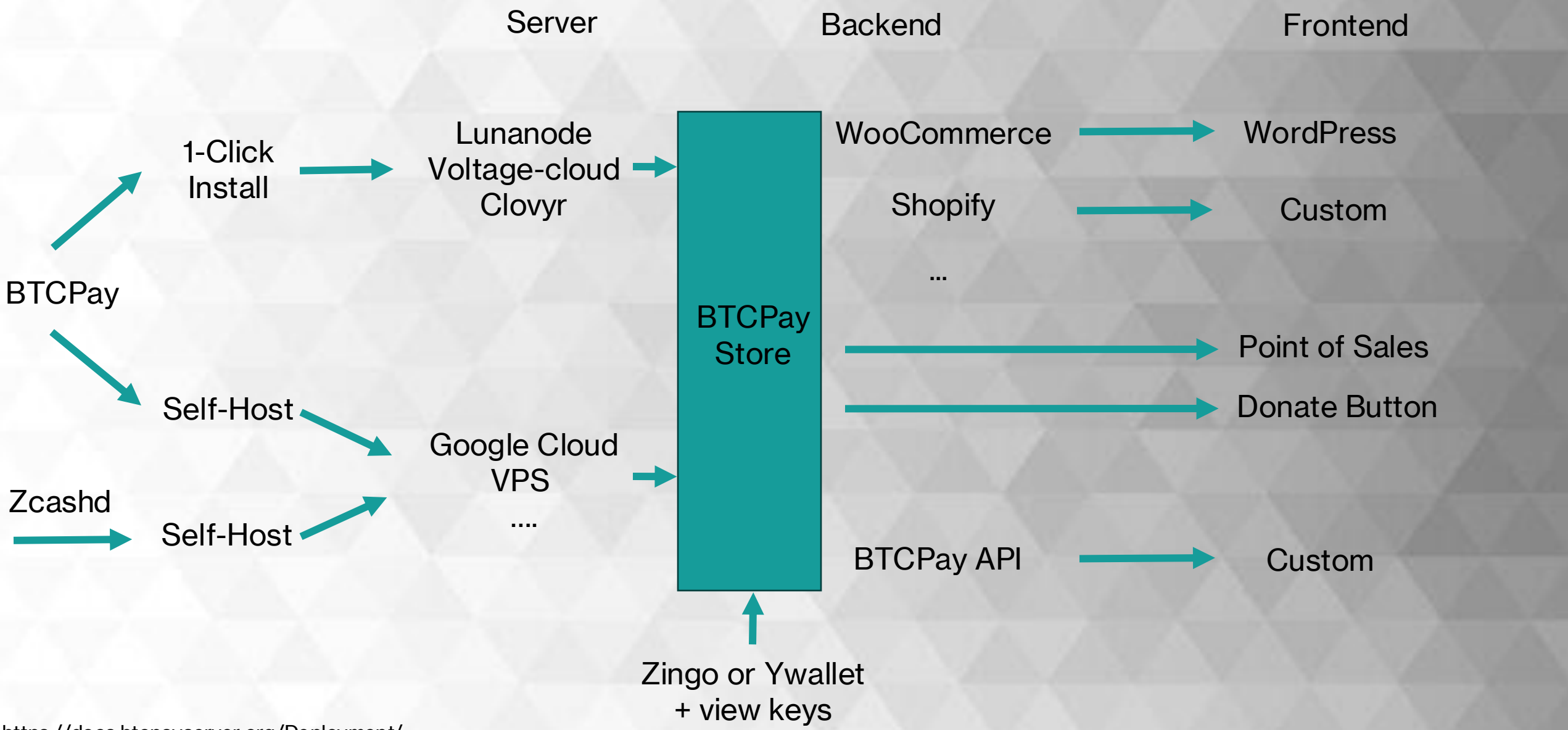
- Enables own custodial zcash wallet
- Self-Hosted
- + Zingo/Ywallet view keys

Frontend:

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

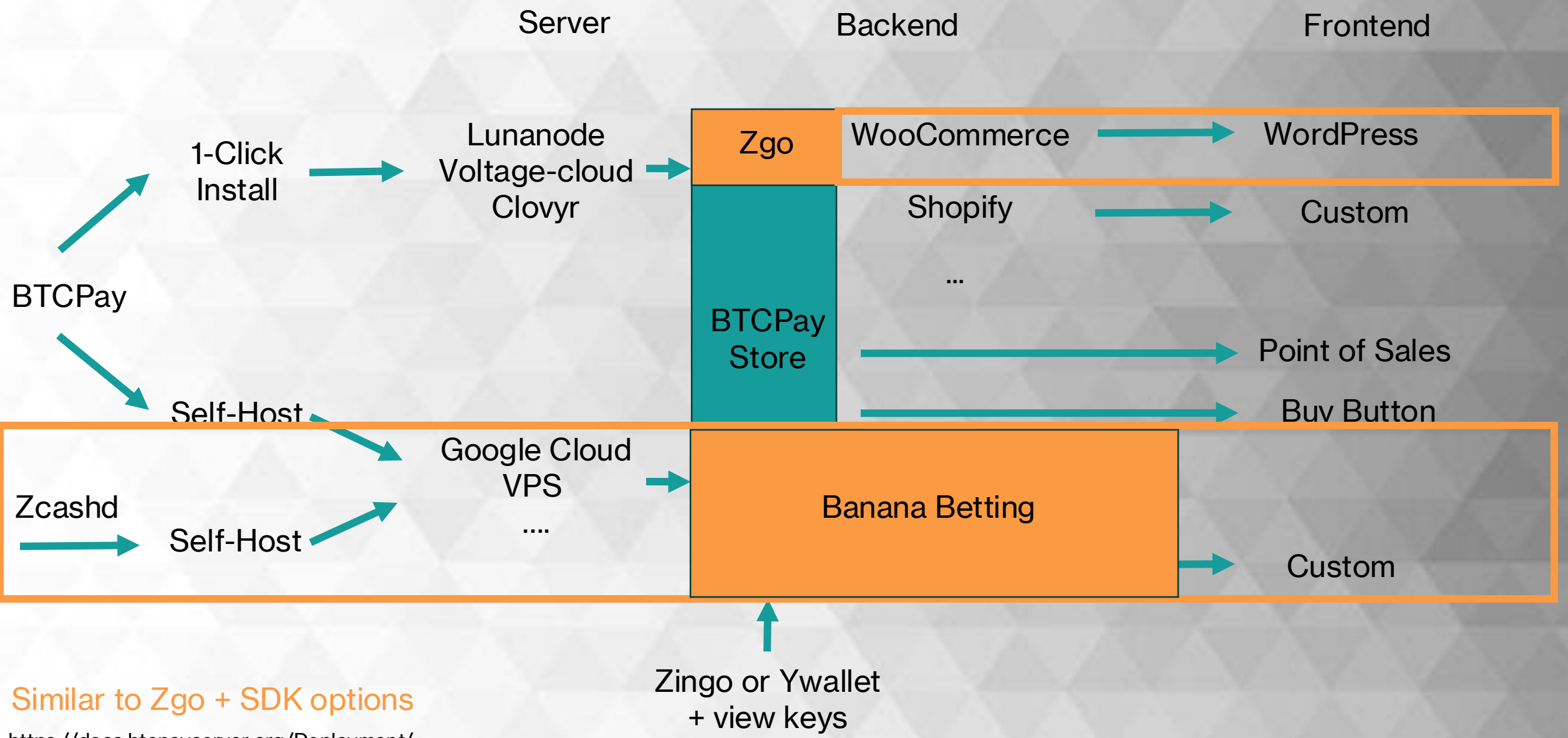
Zcash Community Grant funded Zcash Plugin, enables ZEC payments via BTCPay

Deployment Options



https://docs.btcpayserver.org/Deployment/

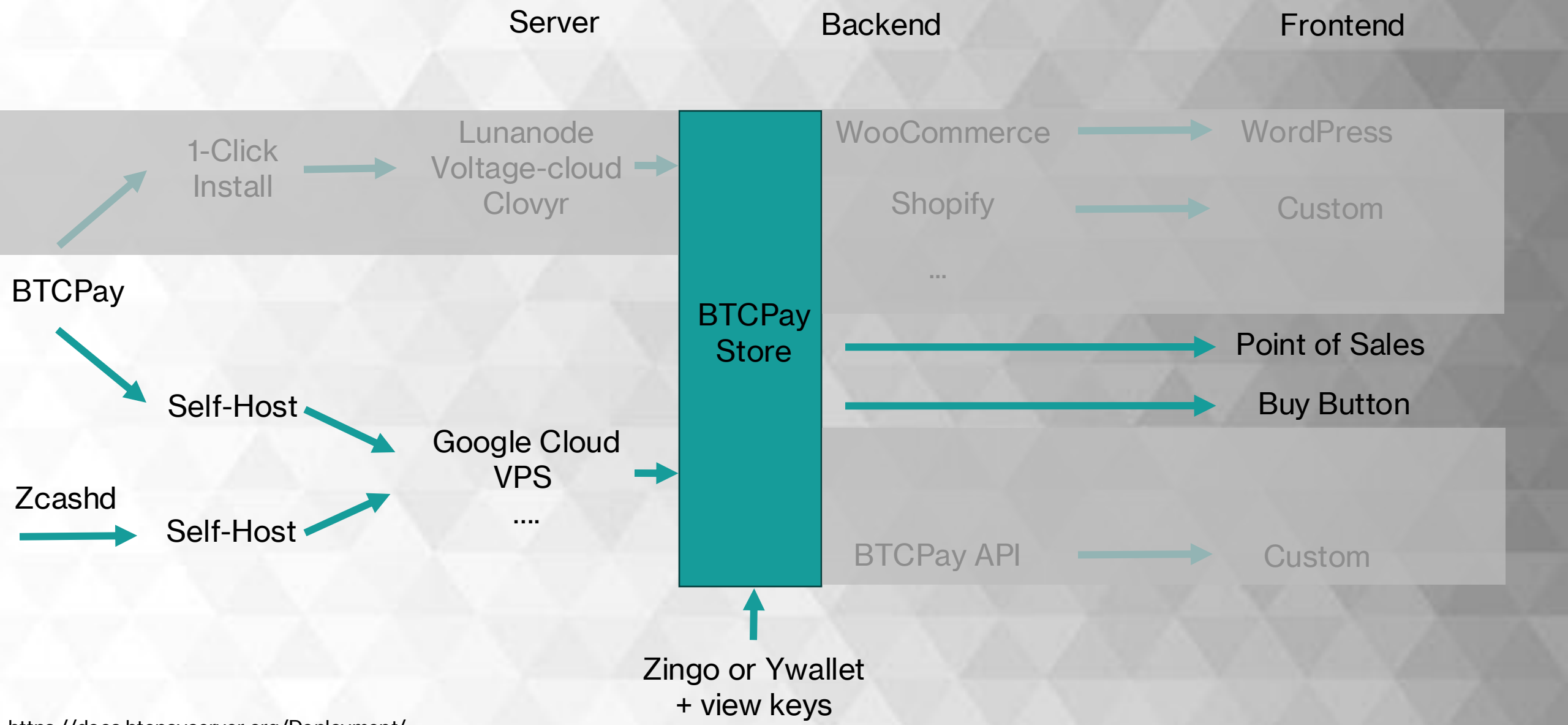
Deployment Options



Similar to Zgo + SDK options

https://docs.btcpayserver.org/Deployment/

Deployment Options



https://docs.btcpayserver.org/Deployment/

From-Scratch Backend Implementation

```
# Clone the docker fragment repository
git clone https://github.com/btcpay-zcash/btcpayserver-docker

# Add ZEC plugin
git checkout feat/zec

# Run btcpay-setup.sh with the right parameters
export BTCPAY_HOST="btcpay.example.com"
export NBITCOIN_NETWORK="mainnet"
export BTCPAYGEN_CRYPTO1="zec"
export BTCPAYGEN_CRYPTO2="btc"
export BTCPAYGEN_CRYPTO2="..."
export BTCPAYGEN_ADDITIONAL_FRAGMENTS=""
export BTCPAYGEN_REVERSEPROXY="nginx" . ./btcpay-setup.sh -l

# Full Node
-> **The first part of the talk on running Zcash nodes**
```

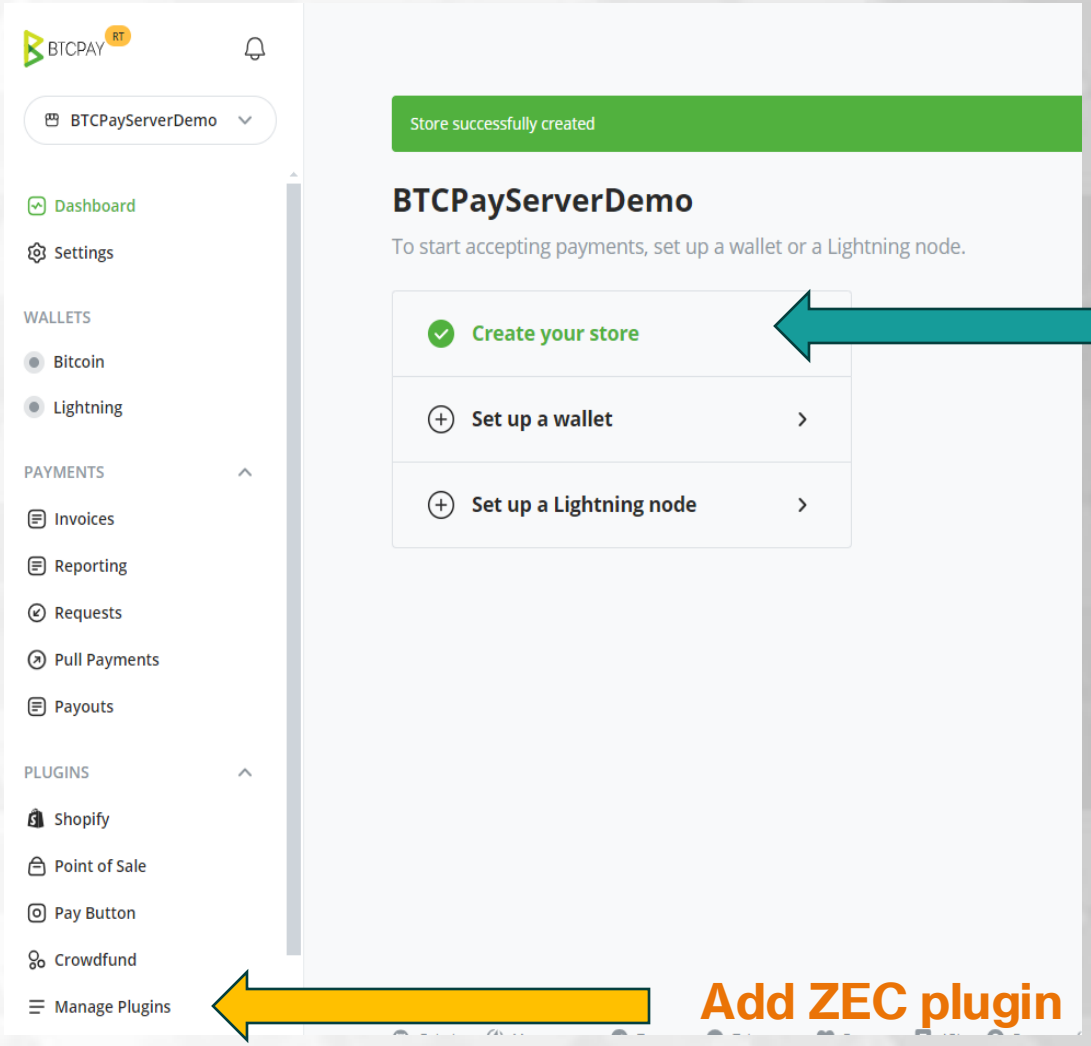
Copy the code!

Add ZEC plugin

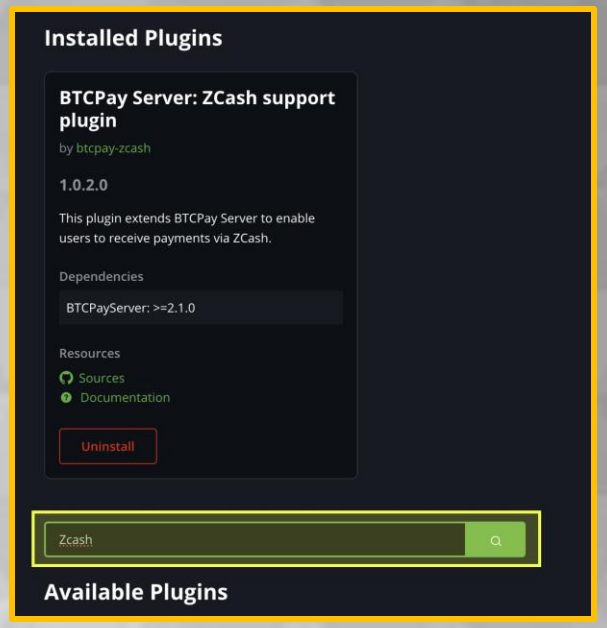
Multi-coin Support

Requires CLI familiarity and networking skills

BTCPay GUI Setup 1 of 2



Create your store!

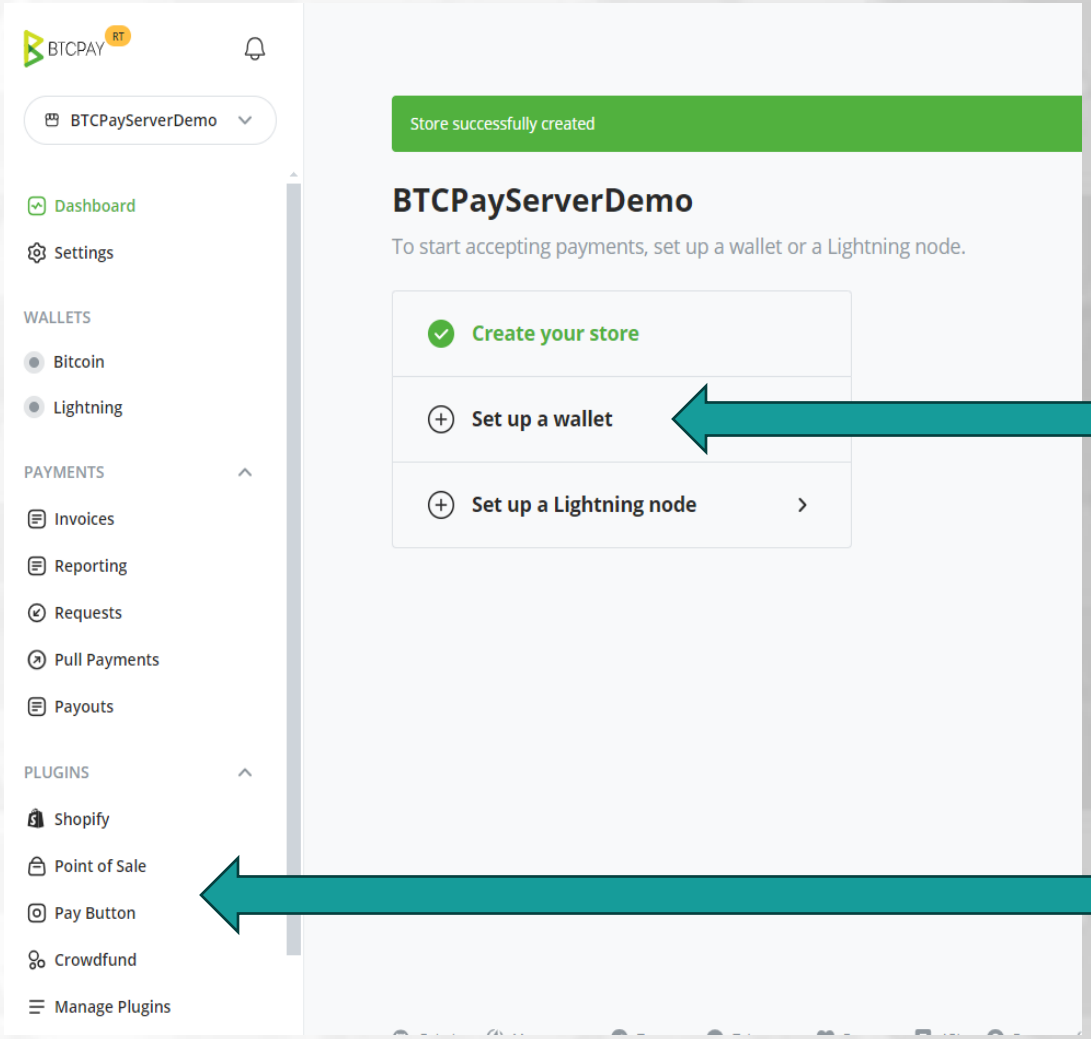


Add ZEC plugin

Graphic interface for store creation and ZEC plugin

https://docs.btcpayserver.org/Apps/#payment-button

BTCPay GUI Setup 2 of 2



Set up your wallets

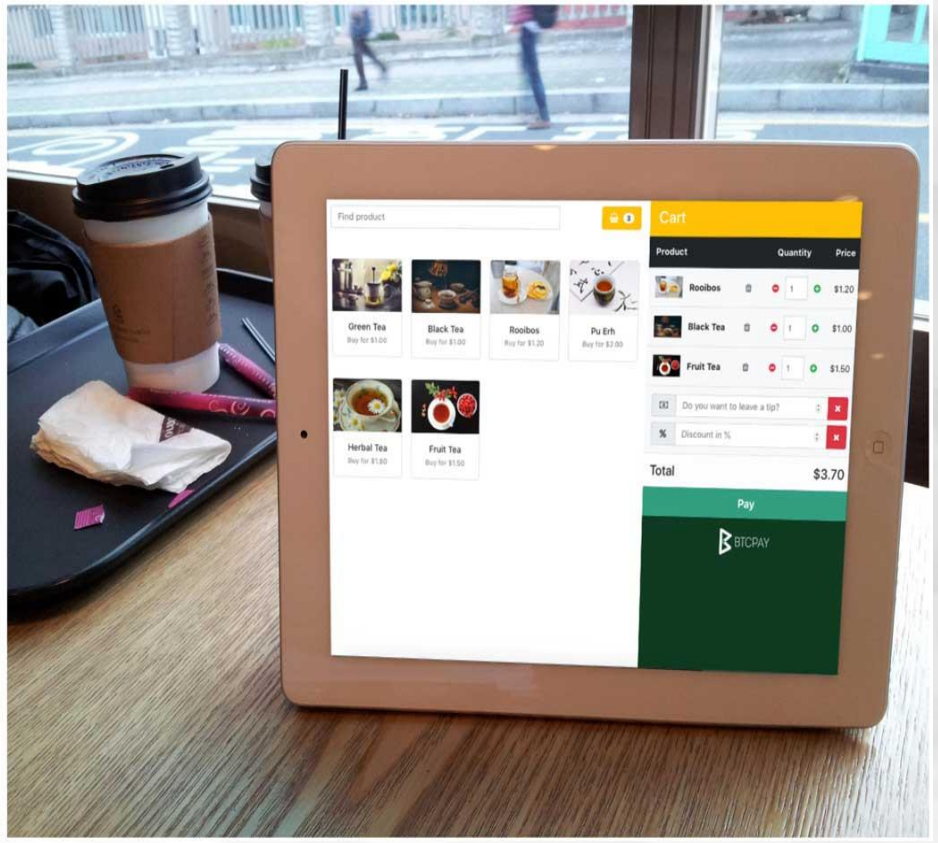
Point of Sales
Pay Button integrations

BTCPay includes plugins for Point of Sales App and HTML site integrations

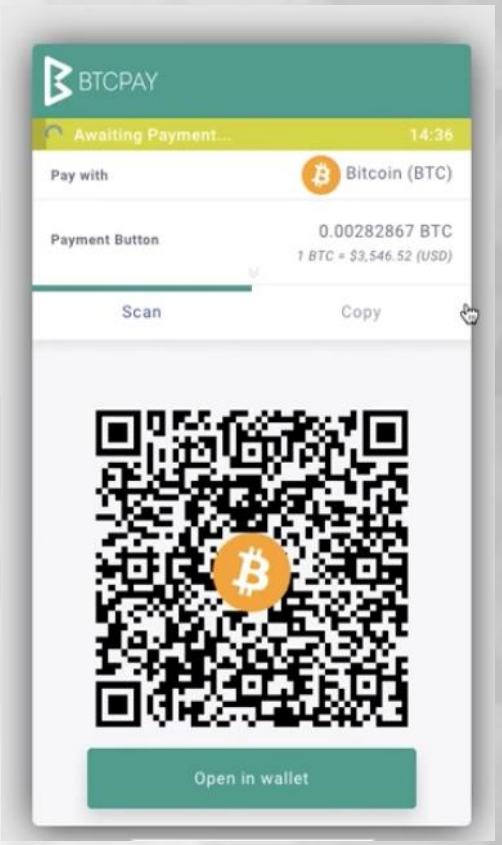
https://docs.btcpayserver.org/Apps/#payment-button

Frontend Implementation

BTCPay Point of Sales



Embeddable HTML

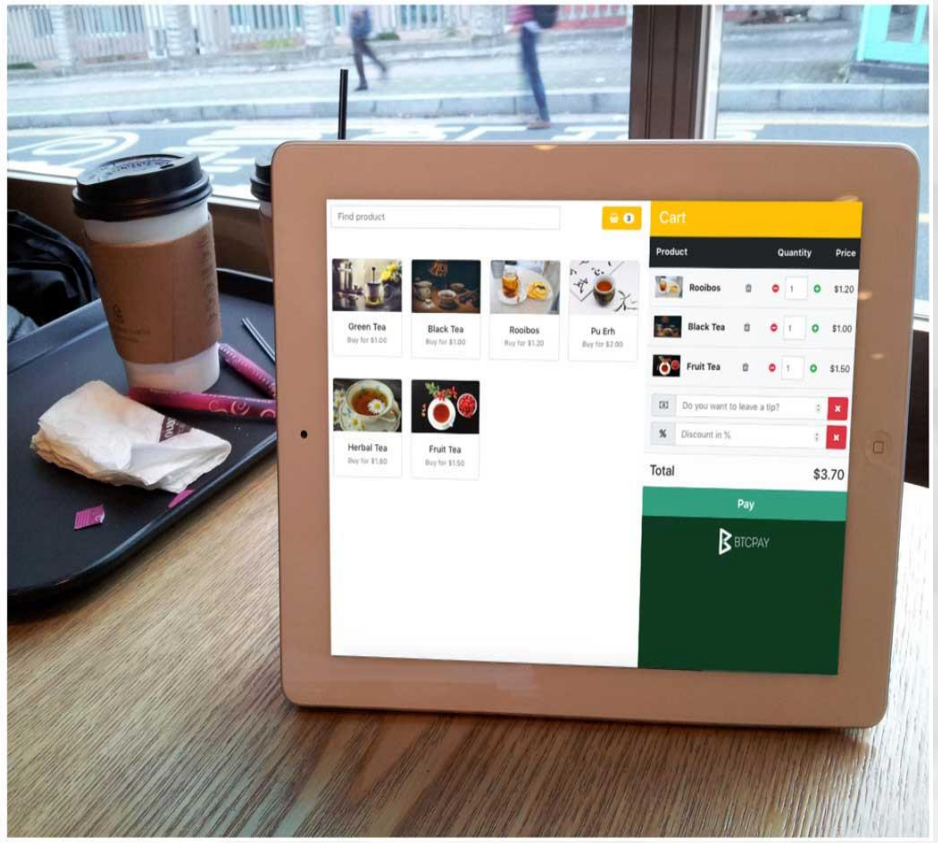


Now accepting shielded ZEC

https://docs.btcpayserver.org/Apps/#payment-button

Frontend Implementation

BTCPay Point of Sales



Embeddable HTML

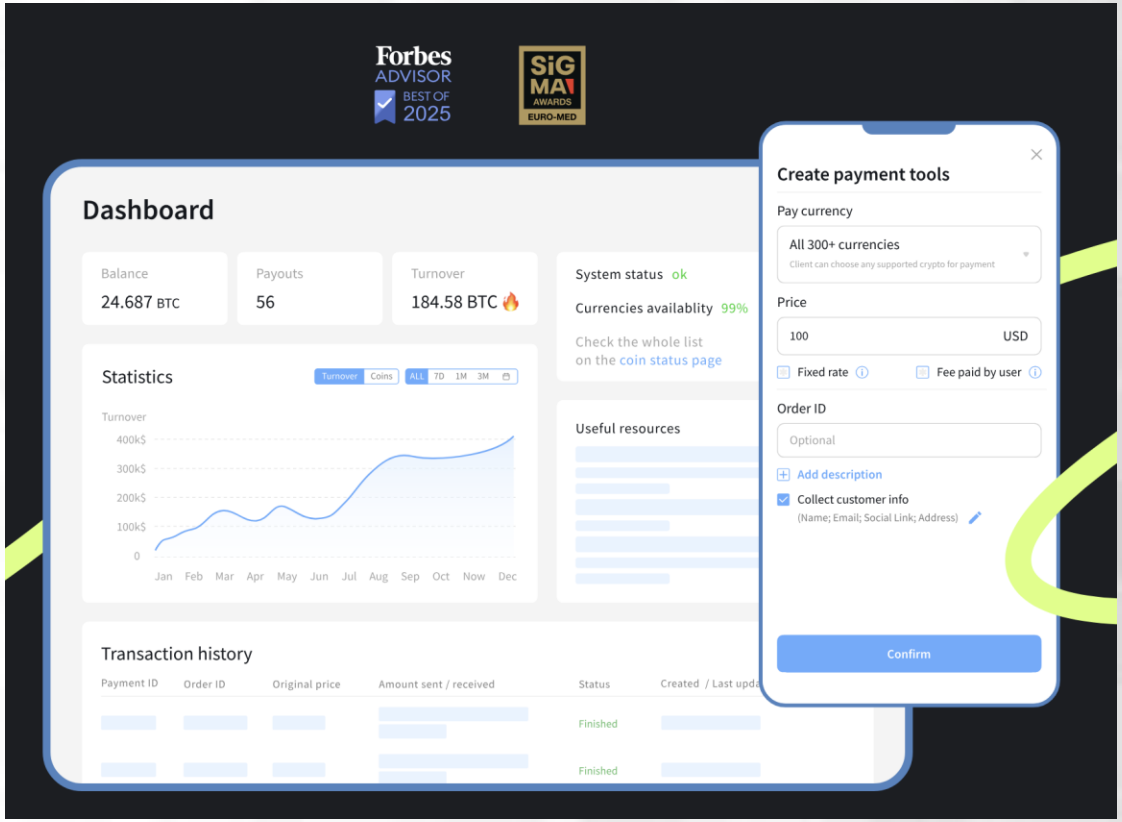
Ready for Business

Now accepting shielded ZEC

https://docs.btcpayserver.org/Apps/#payment-button

		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 processer, 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 shieled 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	https://github.com/ReadyMouse/BananaBetting
ZEC RPC Nodes	https://docs.tatum.io/reference/rpc-zcash
Pay Processor for Non-profits	https://www.every.org/crypto
Zcashd Node Docker Files	https://github.com/zecrocks/zcash-stack/blob/main/docker/compose.zcashd.yaml
Zcash Community Grants	https://zcashcommunitygrants.org/
BTCPay Zcash Plugin	https://github.com/btcpay-zcash/btcpayserver-zcash-plugin/tree/master